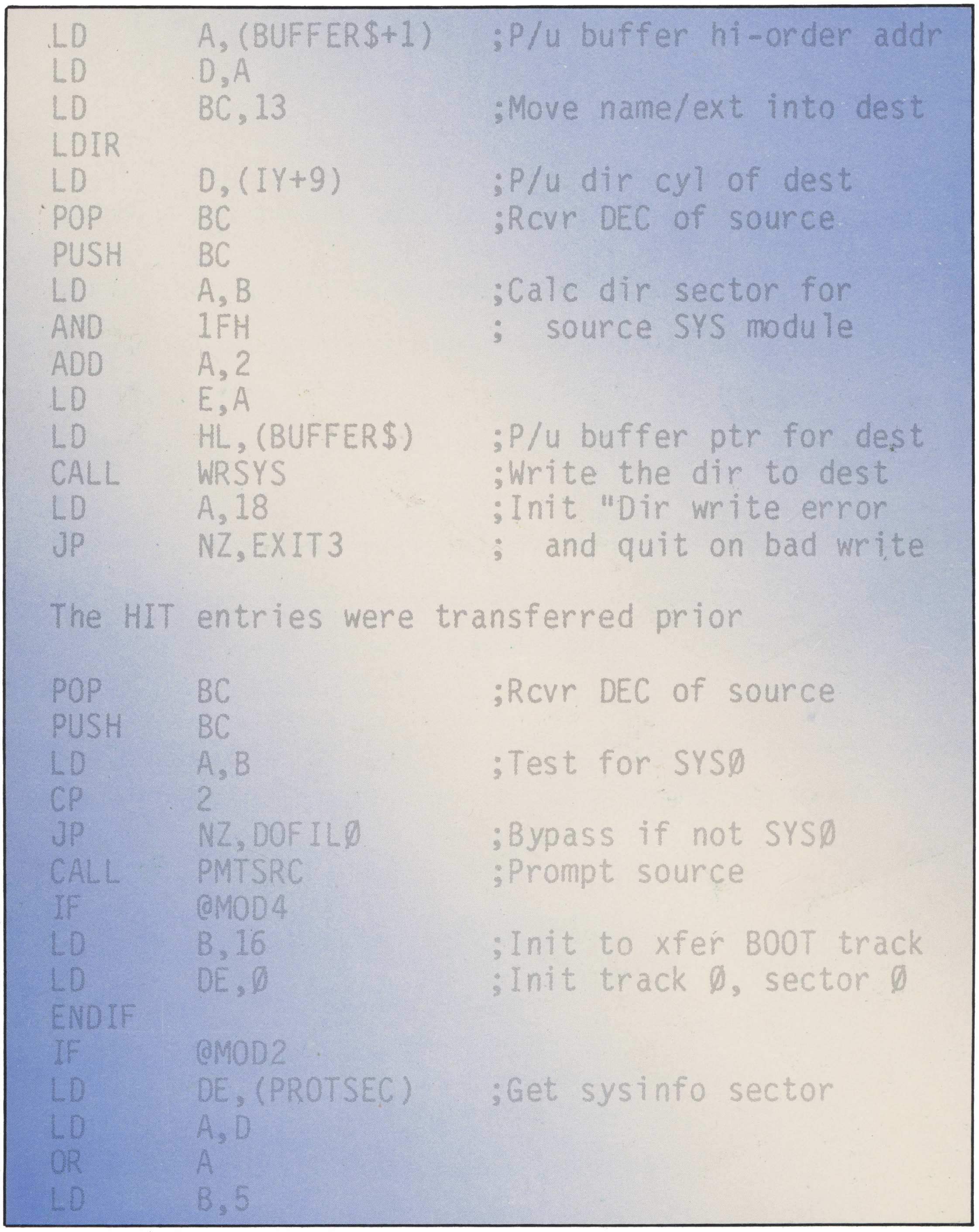
$99

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 SYS0
CP      2
JP      NZ,DOFIL0       ;Bypass if not SYS0
CALL    PMTSRC          ;Prompt source
IF      @MOD4
LD      B,16            ;Init to xfer BOOT track
LD      DE,0            ;Init track 0, sector 0
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

Introduction

BOOT/SYS

SYS0

Appendix A

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-80 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 SYS0/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, LDOS60/EQU and SYS0/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 0 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
```

```
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     00011111B       ;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
```

```
02D8 82        01230          ADD     A,D
02D9 10F8      01240 MLTCA1   DJNZ    MLTCA
02DB C1        01250          POP     BC
02DC C9        01260          RET
               01270 ;
               01280 ;        Initialize the CRTC
               01290 ;
               01300 INITCRTC
02DD 01880F    01310          LD      BC,15<8!88H     ;Count, CRTC address reg
02E0 21FD02    01320          LD      HL,CRTCTAB
02E3 7E        01330 $A1      LD      A,(HL)
02E4 ED41      01340                                  ;Pass reg # to CRTC
02E6 D389      01350                                  ;Pass value to CRTC reg
02E8 2B        01360          DEC     HL              ;Backup to next value
02E9 05        01370          DEC     B               ;To next lower reg
02EA F2E302    01380          JP      P,$A1
02ED C9        01390          RET
02EE 63        01400          DB      99              ;Horiz total MD
02EF 50        01410          DB      80              ;Horiz displayed MD
02F0 56        01420          DB      86              ;Horiz sync position MD
02F1 08        01430          DB      8               ;Horiz sync width
02F2 18        01440          DB      24              ;Vertical total
02F3 00        01450          DB      0               ;Vertical total adjust
02F4 18        01460          DB      24              ;Vertical displayed
02F5 18        01470          DB      24              ;Vertical sync position
02F6 00        01480          DB      0               ;Interlace mode
02F7 09        01490          DB      9               ;Maximum scan line addr
02F8 65        01500          DB      65H             ;Cursor start
02F9 09        01510          DB      9               ;Cursor end
02FA 00        01520          DB      0               ;Start address (H)
02FB 00        01530          DB      0               ;Start address (L)
02FC 00        01540          DB      0               ;Cursor (H)
02FD 00        01550 CRTCTAB  DB      0               ;Cursor (L)
02FE 00        01560          DC      -$&0FFH,0
     00
               01570 ;
               01580 ;        System BOOT entry point, loaded by ROM
               01590 ;
0300           01600 CORE$    DEFL    $
4300           01610          ORG     4300H
4300 00        01620 BOOT     NOP
4301 FE14      01630          CP      14H             ;Directory track location
4302           01640 DIRTRK   EQU     $-1
4303 F3        01650          DI
4304 3E86      01660                                  ;Bring up the RAM
4306 D384      01670
4308 327800    01680          LD      (OPREG$),A
430B 2100F8    01690          LD      HL,CRTBGN$      ;Clear video RAM
430E 1101F8    01700          LD      DE,CRTBGN$+1
4311 017F07    01710          LD      BC,CRTSIZE-1
4314 3620      01720          LD      (HL),' '
4316 EDB0      01730          LDIR
4318 21CD43    01740          LD      HL,NMIRET       ;Set NMI vector
431B 226700    01750          LD      (NMIVECT+1),HL
431E 3EC3      01760          LD      A,0C3H
4320 326600    01770          LD      (NMIVECT),A
4323 3EC9      01780          LD      A,0C9H          ;Stuff return for ints
4325 323800    01790          LD      (38H),A
               01800 ;
```

```
                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
```

```
438F 1193FB    02390         LD      DE,80*11+CRTBGN$+35     ;Middle of screen
4392 EDB0      02400         LDIR
4394 18FE      02410 HALTS   JR      HALTS           ;Wait for RESET
               02420 ;
4396 01F481    02430 READ                            ;Set DDEN, DS1, d.s. port
4399 ED41      02440                                 ;Select it
439B 0D        02450         DEC     C               ;Point C to data reg
439C 3E18      02460         LD      A,18H           ;Seek command (6 ms)
439D           02470 BOOTST$ EQU     $-1             ;Set for boot step rate
               02480         IFNE    BOOTST$,439DH
               02490         ERR     'Boot step out of position'
               02500         ENDIF
439E ED51      02510                                 ;Set desired track
43A0 CDD943    02520         CALL    FDCMD           ;Pass command & delay
43A3 DBF0      02530 SEEK1   IN      A,(0F0H)        ;Get status
43A5 CB47      02540         BIT     0,A             ;Busy?
43A7 20FA      02550         JR      NZ,SEEK1
43A9 7B        02560         LD      A,E             ;Set sector register
43AA D3F2      02570
43AC 3E81      02580         LD      A,81H           ;Set DDEN & DS1
43AE D3F4      02590
43B0 D5        02600         PUSH    DE
43B1 1102C1    02610         LD      DE,81H!40H<8!2  ;D=DS1 + DDEN + WSGEN
               02620                                 ;E=Mask to see DRQ
43B4 3E80      02630         LD      A,80H           ;FDC READ command
43B6 CDD943    02640         CALL    FDCMD           ;Pass to ctrlr & set B=0
43B9 3EC0      02650         LD      A,0C0H          ;Enable INTRQ & timeout
43BB D3E4      02660
43BD DBF0      02670 READLP1                         ;Grab status
43BF A3        02680         AND     E               ;Test bit 1
43C0 28FB      02690         JR      Z,READLP1
43C2 EDA2      02700         INI
43C4 7A        02710         LD      A,D             ;Set DDEN & DS1 & WSGEN
43C5 D3F4      02720 READLP2                         ;Continue to select
43C7 EDA2      02730         INI                     ;  while inputting
43C9 20FA      02740         JR      NZ,READLP2
43CB 18FE      02750         JR      $               ;Wait for NMI
43CD D1        02760 NMIRET  POP     DE              ;Pop interrupt ret
43CE D1        02770         POP     DE              ;Restore DE
43CF AF        02780         XOR     A               ;Disable INTRQ & timeout
43D0 D3E4      02790
43D2 3E81      02800         LD      A,81H           ;Reselect drive
43D4 D3F4      02810
43D6 DBF0      02820                                 ;Get status
43D8 C9        02830         RET
43D9 D3F0      02840 FDCMD                           ;Give cmd to ctrlr
43DB 0618      02850         LD      B,24            ;Time delay
43DD 10FE      02860         DJNZ    $
43DF C9        02870         RET
43E0 44        02880 DISKERR DB      'Disk error'
     69 73 6B 20 65 72 72 6F
     72
43EA 4E        02890 NOSYS   DB      'No system '
     6F 20 73 79 73 74 65 6D
     20
000A           02900 ERRLEN  EQU     $-NOSYS         ;Length of error msg
43F4 00        02910         DC      -$&0FFH,0
     00 00 00 00 00 00 00 00
     00 00 00
```

Bootstrap Loader

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

```
               02970 ;
001C           02980 HOME    EQU     1CH
001F           02990 CLRFRM  EQU     1FH
               03000 ;
               03010 ;       Log out routine - display & log
               03020 ;
0500 CD2D05    03030 @LOGOT  CALL    @DSPLY
               03040 ;
               03050 ;       Job log loger routine
               03060 ;
0503 3A3002    03070 @LOGER  LD      A,(JLDCB$)      ;If NIL, don't do
0506 EE08      03080         XOR     8               ;  anything
0508 E608      03090         AND     8
050A C8        03100         RET     Z
050B E5        03110         PUSH    HL              ;Save pointer to command
050C 211D05    03120         LD      HL,LOGBUF       ;Get time string into buf
050F E5        03130         PUSH    HL
0510 CD8D07    03140         CALL    @TIME
0513 E1        03150         POP     HL
0514 113002    03160         LD      DE,JLDCB$       ;Log the time
0517 CD3005    03170         CALL    @MSG
051A E1        03180         POP     HL              ;Log the command
051B 1813      03190         JR      @MSG
051D 68        03200 LOGBUF  DB      'hh:mm:ss  ',3
     68 3A 6D 6D 3A 73 73 20
     20 03
               03210 ;
               03220 ;       Line print routine
               03230 ;
0528 111802    03240 @PRINT  LD      DE,PRDCB$       ;Printer DCB
052B 1803      03250         JR      @MSG
               03260 ;
               03270 ;       Line display routine
               03280 ;
052D 111002    03290 @DSPLY  LD      DE,DODCB$       ;Video DCB
               03300 ;
               03310 ;       Device message routine
               03320 ;
               03330 *MOD
0530 E5        03340 @MSG    PUSH    HL              ;Save pointer to message
0531 7E        03350 $B1     LD      A,(HL)          ;P/u a message character
0532 FE03      03360         CP      3               ;Exit on ETX
0534 280D      03370         JR      Z,$B3
0536 FE0D      03380         CP      CR              ;Exit & put on ENTER
0538 2806      03390         JR      Z,$B2
053A C44506    03400         CALL    NZ,@PUT         ;Else put the char
053D 23        03410         INC     HL              ;  & loop on no error
053E 28F1      03420         JR      Z,$B1           ;  else fall thru & exit
0540 CC4506    03430 $B2     CALL    Z,@PUT
0543 E1        03440 $B3     POP     HL
0544 C9        03450         RET
               03460 ;
               03470 ;       Clear screen routine
               03480 ;
0545 3E1C      03490 @CLS    LD      A,HOME          ;Cursor home to 0,0
0547 CD4D05    03500         CALL    DSPBYT
054A C0        03510         RET     NZ              ;Return on error
054B 3E1F      03520         LD      A,CLRFRM        ;Clear to end of frame
054D D5        03530 DSPBYT  PUSH    DE
054E CD4206    03540         CALL    @DSP
```

Device I/O handling

```
0551 D1         03550           POP     DE
0552 C9         03560           RET
                03570 ;
                03580 ;         Check and Clear <BREAK> bit SVC
                03590 ;
                03600 @CKBRKC
0553 E5         03610           PUSH    HL              ;Save registers
0554 217400     03620           LD      HL,KFLAG$       ;Point to KFLAG$
0557 CB46       03630           BIT     0,(HL)          ;Check break bit
0559 281A       03640           JR      Z,NOBRK         ;  and ret if none
055B F5         03650           PUSH    AF              ;Save flags
055C C5         03660           PUSH    BC
055D D5         03670           PUSH    DE
055E CB86       03680 BRKTEST   RES     0,(HL)          ;Reset the break bit
0560 01000B     03690           LD      BC,0B00H        ;Wait more than 1/30
0563 CD8203     03700           CALL    PAUSE@          ;  of a second
0566 CB46       03710           BIT     0,(HL)          ;Test the bit again
0568 20F4       03720           JR      NZ,BRKTEST      ;Loop until gone
056A 110802     03730           LD      DE,KIDCB$       ;Point at keyboard &
056D 3E03       03740           LD      A,03            ;  clear buffer with
056F CD2306     03750           CALL    @CTL            ;  control 3 call
0572 D1         03760           POP     DE
0573 C1         03770           POP     BC              ;Recover registers
0574 F1         03780           POP     AF              ;Recover FLAGS
0575 E1         03790 NOBRK     POP     HL
0576 C9         03800           RET
                03810 ;
                03820 ;         Keyboard line input routine
                03830 ;
                03840 *MOD
                03850 ;
                03860 ;         Backspace to beginning of line
                03870 ;
0577 CDDB05     03880 $C4       CALL    $C6             ;Backspace
057A 2B         03890           DEC     HL              ;Get the char prior
057B 7E         03900           LD      A,(HL)          ;  to the current
057C 23         03910           INC     HL
057D FE0A       03920           CP      0AH             ;Return if line feed
057F C8         03930           RET     Z
0580 78         03940 $C5       LD      A,B             ;Check for empty buffer
0581 B9         03950           CP      C
0582 20F3       03960           JR      NZ,$C4          ;Loop if not
0584 C9         03970           RET                     ;  else return
0585 E5         03980 @KEYIN    PUSH    HL              ;Save buffer pointer
0586 48         03990           LD      C,B             ;Set C = buffer size
0587 112806     04000 $C1       LD      DE,@KEY         ;Init for standard input
058A 3A7C00     04010           LD      A,(SFLAG$)      ;If JCL is active,
058D E620       04020           AND     20H             ;  then use the JCL input
058F 2802       04030           JR      Z,$C0           ;Must loop here in case
0591 1E30       04040           LD      E,@JCL&0FFH     ;  JCL exits with //STOP
0593 ED539805   04050 $C0       LD      ($C1A+1),DE
0597 CD0000     04060 $C1A      CALL    $-$             ;Get a key
059A 203D       04070           JR      NZ,$C3B         ;Back on error
059C FE80       04080           CP      80H             ;Break?
059E 2875       04090           JR      Z,$C10
05A0 FE20       04100           CP      20H             ;Go if not a control
05A2 3020       04110           JR      NC,$C2
05A4 FE0D       04120           CP      0DH             ;Carriage return?
05A6 286E       04130           JR      Z,$C11
```

Device I/O handling

```
05A8 FE1F      04140         CP      1FH             ;Clear?
05AA 2825      04150         JR      Z,$C3
05AC 118705    04160         LD      DE,$C1          ;Set return address
05AF D5        04170         PUSH    DE
05B0 FE08      04180         CP      08H             ;Backspace?
05B2 2827      04190         JR      Z,$C6
05B4 FE18      04200         CP      18H             ;Backspace to BOL?
05B6 28C8      04210         JR      Z,$C5
05B8 FE09      04220         CP      09H             ;Tab?
05BA 283C      04230         JR      Z,$C8
05BC FE12      04240         CP      'R'&1FH         ;CTL-R?
05BE 282A      04250         JR      Z,$C7
05C0 FE0A      04260         CP      0AH             ;Line feed?
05C2 C0        04270         RET     NZ              ;Ret if none above
05C3 D1        04280         POP     DE              ;Pop the return
05C4 77        04290 $C2     LD      (HL),A          ;Stuff the char
05C5 78        04300         LD      A,B             ;Check on buffer full
05C6 B7        04310         OR      A
05C7 28BE      04320         JR      Z,$C1           ;Loop if so
05C9 7E        04330         LD      A,(HL)          ;  else get char
05CA 23        04340         INC     HL              ;  & bump pointer
05CB 05        04350         DEC     B               ;Count down
05CC CD4206    04360         CALL    @DSP            ;Display entry
05CF 1806      04370         JR      $C3A            ;  then loop
               04380 ;
               04390 ;       Clear the screen invoked
               04400 ;
05D1 CD4505    04410 $C3     CALL    @CLS
05D4 41        04420         LD      B,C             ;Reset to start of
05D5 E1        04430         POP     HL              ;  line & start of
05D6 E5        04440         PUSH    HL              ;  buffer
05D7 28AE      04450 $C3A    JR      Z,$C1
05D9 183B      04460 $C3B    JR      $C11
               04470 ;
               04480 ;       Backspace key entry
               04490 ;
05DB 78        04500 $C6     LD      A,B             ;If buffer is empty,
05DC B9        04510         CP      C               ;  return
05DD C8        04520         RET     Z
05DE 2B        04530         DEC     HL              ;  else do the backspace
05DF 7E        04540         LD      A,(HL)
05E0 FE0A      04550         CP      0AH             ;Last char a linefeed?
05E2 23        04560         INC     HL
05E3 C8        04570         RET     Z               ;Return if so
05E4 2B        04580         DEC     HL
05E5 04        04590         INC     B               ;Add back one char
05E6 3E08      04600         LD      A,8             ;Backspace the cursor
05E8 1858      04610         JR      @DSP
               04620 ;
               04630 ;       Test if repeat last command
               04640 ;
05EA 3A6C00    04650 $C7     LD      A,(CFLAG$)      ;Test if SYS1 KEYIN bit
05ED E604      04660         AND     4               ;  is set (bit 2)
05EF C8        04670         RET     Z               ;Ignore CTL if not
05F0 78        04680         LD      A,B             ;If not at 1st position,
05F1 B9        04690         CP      C               ;  don't permit it
05F2 C0        04700         RET     NZ
05F3 E1        04710         POP     HL              ;Pop return to KEY
05F4 E1        04720         POP     HL              ;Point to command buffer
```

```
05F5 C32D05    04730          JP      @DSPLY          ;Display the old command
               04740 ;
               04750 ;        Tab entered
               04760 ;
05F8 E5        04770 $C8      PUSH    HL              ;Get pos on line
05F9 CDF10D    04780          CALL    ADDR_2_ROWCOL   ;Get row,col in HL
05FC 7D        04790          LD      A,L             ;Xfer column to A
05FD E1        04800          POP     HL
05FE E607      04810          AND     7
0600 ED44      04820          NEG                     ;Negate and add tab
0602 C608      04830          ADD     A,8
0604 5F        04840          LD      E,A             ;Reg E has tab length
0605 78        04850 $C9      LD      A,B             ;Check on buffer full
0606 B7        04860          OR      A
0607 C8        04870          RET     Z
0608 3E20      04880          LD      A,' '           ;Put spaces until
060A 77        04890          LD      (HL),A          ;  tab expanded
060B 23        04900          INC     HL
060C CD4D05    04910          CALL    DSPBYT
060F C0        04920          RET     NZ
0610 05        04930          DEC     B               ;Dec buffer remaining
0611 1D        04940          DEC     E               ;Dec tab count
0612 C8        04950          RET     Z
0613 18F0      04960          JR      $C9
               04970 ;
               04980 ;        Exit KEYIN routine
               04990 ;
0615 37        05000 $C10     SCF                     ;BREAK exit with CF
0616 F5        05010 $C11     PUSH    AF              ;Save flag
0617 3E0D      05020          LD      A,0DH           ;Stuff CR at end
0619 77        05030          LD      (HL),A
061A CD4206    05040          CALL    @DSP            ;  & display it
061D 79        05050          LD      A,C             ;Calculate # of chars
061E 90        05060          SUB     B               ;  entered
061F 47        05070          LD      B,A
0620 F1        05080          POP     AF              ;Rcvr flag
0621 E1        05090          POP     HL              ;Restore buffer ptr
0622 C9        05100          RET
               05110 ;
               05120 ;        Byte I/O device handler
               05130 ;                C => character if PUT or CTL
               05140 ;                DE => Device control block
               05150 ;
               05160 *MOD
0623 C5        05170 @CTL     PUSH    BC
0624 0604      05180          LD      B,4             ;Bit 2, CTL
0626 1820      05190          JR      IOBGN
0628 CD3506    05200 @KEY     CALL    @KBD            ;Scan the keyboard
062B C8        05210          RET     Z               ;Ret if key available
062C B7        05220          OR      A               ;Return if error
062D 28F9      05230          JR      Z,@KEY
062F C9        05240          RET
0630 110302    05250 @JCL     LD      DE,JCLCB$       ;JCL file FCB
0633 1803      05260          JR      @GET
0635 110802    05270 @KBD     LD      DE,KIDCB$       ;Keyboard DCB
0638 C5        05280 @GET     PUSH    BC
0639 0601      05290          LD      B,1             ;Bit 0, GET
063B 180B      05300          JR      IOBGN
063D 111802    05310 @PRT     LD      DE,PRDCB$       ;Printer DCB
```

Device I/O handling

```
0640 1803     05320           JR      @PUT
0642 111002   05330 @DSP      LD      DE,DODCB$        ;Video DCB
0645 C5       05340 @PUT      PUSH    BC
0646 0602     05350           LD      B,2              ;Bit 1, PUT
0648 DDE5     05360 IOBGN     PUSH    IX               ;Save the registers
064A E5       05370           PUSH    HL
064B D5       05380           PUSH    DE               ;Xfer DCB to IX
064C DDE1     05390           POP     IX
064E D5       05400           PUSH    DE
064F 4F       05410           LD      C,A              ;Xfer the I/O char
0650 218006   05420           LD      HL,@RSTREG       ;Restore register routine
0653 3A0202   05430           LD      A,(LBANK$)       ;If bank 0 is not
0656 B7       05440           OR      A                ;  resident, need to
0657 280E     05450           JR      Z,$D0            ;  get it resident!
              05460 ;
              05470 ;         Some other bank is resident - invoke bank 0
              05480 ;
0659 C5       05490           PUSH    BC               ;Save reg again
065A AF       05500           XOR     A                ;Prepare for bank-0
065B 47       05510           LD      B,A
065C 4F       05520           LD      C,A
065D CD7708   05530           CALL    @BANK            ;Invoke bank-0
0660 60       05540           LD      H,B              ;Get old bank data
0661 69       05550           LD      L,C              ;  into reg HL
0662 C1       05560           POP     BC               ;Rcvr BC
0663 E5       05570           PUSH    HL               ;Bank data to stack
0664 217906   05580           LD      HL,RSTBNK        ;Set return address
0667 E5       05590 $D0       PUSH    HL               ;  to restore registers
0668 1A       05600           LD      A,(DE)           ;P/u DCB type byte
0669 B7       05610           OR      A
066A C8       05620           RET     Z                ;Back if nothing
066B FE08     05630           CP      8                ;Ck on GET/PUT/CTL
066D 301A     05640           JR      NC,@CHNIO        ;Branch if special
066F DD6E01   05650           LD      L,(IX+1)         ;  else p/u the vector
0672 DD6602   05660           LD      H,(IX+2)
0675 78       05670 $D1       LD      A,B              ;Xfer I/O code
0676 FE02     05680           CP      2                ;Set flags state
0678 E9       05690           JP      (HL)
0679 C1       05700 RSTBNK    POP     BC               ;Get old bank data
067A F5       05710           PUSH    AF               ;Can't affect AF
067B 79       05720           LD      A,C              ;Request to A
067C CD7708   05730           CALL    @BANK            ;Bring back original bank
067F F1       05740           POP     AF
0680 D1       05750 @RSTREG   POP     DE               ;Restore regs
0681 E1       05760           POP     HL
0682 DDE1     05770           POP     IX
0684 C1       05780           POP     BC
0685 C9       05790           RET
              05800 ;
0686 E5       05810 $D2       PUSH    HL
0687 DDE1     05820           POP     IX
0689 DD6E01   05830 @CHNIO    LD      L,(IX+1)         ;P/u vector address
068C DD6602   05840           LD      H,(IX+2)
068F DD7E00   05850 $D3       LD      A,(IX+0)         ;P/u the DCB type
0692 B7       05860           OR      A                ;File Control Block?
0693 FA0013   05870           JP      M,@BYTEIO
0696 CB5F     05880           BIT     3,A              ;Test NIL bit 2nd
0698 2024     05890           JR      NZ,$D5
069A CB67     05900           BIT     4,A              ;Routed?
```

Device I/O handling

```
069C 20E8     05910          JR      NZ,$D2          ;Go get routed DCB
069E CB6F     05920          BIT     5,A             ;If not linked, then
06A0 28D3     05930          JR      Z,$D1           ;  must be filtered
06A2 E5       05940          PUSH    HL              ;Point to the link DCB
06A3 DDE1     05950          POP     IX
06A5 DD7003   05960          LD      (IX+3),B        ;Save the direction
06A8 DDE5     05970          PUSH    IX
06AA CD8906   05980          CALL    @CHNIO          ;I/O to 1st device
06AD DDE1     05990          POP     IX
06AF DD4603   06000          LD      B,(IX+3)        ;P/u the direction
06B2 200C     06010          JR      NZ,$D6          ;Go on NZ flag
              06020 ;
              06030 ;        Z-flag on return - check input/output
              06040 ;
06B4 CB40     06050          BIT     0,B             ;If input & got char,
06B6 DD6E04   06060 $D4      LD      L,(IX+4)        ;  p/u the linked DCB
06B9 DD6605   06070          LD      H,(IX+5)
06BC 28C8     06080          JR      Z,$D2
06BE BF       06090 $D5      CP      A
06BF C9       06100          RET
              06110 ;
              06120 ;        1st link got NZ condition - if input, get link
              06130 ;
06C0 CB40     06140 $D6      BIT     0,B             ;Was it input/output?
06C2 2803     06150          JR      Z,$D7           ;Output is error
06C4 B7       06160          OR      A               ;If A=0, then no input
06C5 28EF     06170          JR      Z,$D4
06C7 B7       06180 $D7      OR      A
06C8 C9       06190          RET
06C9          01530 *GET     MULDIV:3                ;16-bit MULT & DIV
              06200 ;MULDIV/ASM - 16 x 8 multiplication & division
06C9          06210          SUBTTL  '<16 X 8 multiply/divide>'
```

16 X 8 multiply/divide

```
                06230 *MOD
                06240 ;
                06250 ;       Multiply HL by A - SVC 91
                06260 ;       HL => multiplicand
                06270 ;       A  => multiplier
                06280 ;       HLA <= 24-bit result
                06290 ;       DE destroyed
                06300 ;
06C9 C5         06310 @MUL16  PUSH    BC              ;Save reg BC
06CA EB         06320         EX      DE,HL           ;Multiplicand to DE
06CB 4F         06330         LD      C,A             ;  & multiplier to C
06CC 210000     06340         LD      HL,0            ;Init value to zero
06CF 7D         06350         LD      A,L             ;  in regs HLA
06D0 0608       06360         LD      B,8             ;Init for 8-bit mult
06D2 29         06370 $E1     ADD     HL,HL           ;Shift to next place
06D3 17         06380         RLA                     ;Use A for bits 16-23
06D4 CB01       06390         RLC     C               ;Multiply this bit?
06D6 3003       06400         JR      NC,$E2          ;Go if not
06D8 19         06410         ADD     HL,DE           ;Else add multiplicand
06D9 CE00       06420         ADC     A,0             ;  & any overflow to 16
06DB 10F5       06430 $E2     DJNZ    $E1             ;Loop for 8 bits
06DD 4F         06440         LD      C,A             ;Tempy save
06DE 7D         06450         LD      A,L             ;Xfer low-order to A
06DF 6C         06460         LD      L,H             ;Xfer mid-order to L
06E0 61         06470         LD      H,C             ;Xfer hi-order to H
06E1 C1         06480         POP     BC
06E2 C9         06490         RET
                06500 ;
                06510 ;       Divide HL by A - SVC 94
                06520 ;       HL => dividend
                06530 ;       A  => divisor
                06540 ;       HL <= resulting quotient
                06550 ;       A  <= remainder
                06560 ;
                06570 *MOD
06E3 D5         06580 @DIV16  PUSH    DE              ;Save this reg pair
06E4 57         06590         LD      D,A             ;Xfer divisor to D
06E5 1E10       06600         LD      E,16            ;Init for 16-bits
06E7 AF         06610         XOR     A
06E8 29         06620 $F1     ADD     HL,HL           ;Rotate dividend
06E9 17         06630         RLA                     ;  & subtract divisor if
06EA 3803       06640         JR      C,$F2           ;  carry into bit-16
06EC BA         06650         CP      D               ;Compare divisor
06ED 3802       06660         JR      C,$F3           ;Go if no subtract
06EF 92         06670 $F2     SUB     D               ;  else subtract divisor
06F0 2C         06680         INC     L               ;Set lo-order
06F1 1D         06690 $F3     DEC     E               ;Count down one bit
06F2 20F4       06700         JR      NZ,$F1          ;Loop for 16-bits
06F4 D1         06710         POP     DE
06F5 C9         06720         RET
                06730 ;
                06740 ;       @HEXDEC - SVC 97
                06750 ;       Routine to convert 16-bit hexadecimal to decimal
                06760 ;       HL => value
                06770 ;       DE => buffer pointer of 5-character buffer
                06780 ;       HL <= destroyed (always set to zero)
                06790 ;       DE <= Buffer+5
                06800 ;       BC <= destroyed
                06810 ;       Z  <= set
                06820 ;
```

16 X 8 multiply/divide

```
              06830 *MOD
06F6 0605     06840 @HEXDEC LD      B,5              ;Length max
06F8 3E20     06850         LD      A,' '            ;Load blank
06FA 12       06860 HEXDEC1 LD      (DE),A           ;To string
06FB 13       06870         INC     DE               ;Bump pointer
06FC 10FC     06880         DJNZ    HEXDEC1          ;Go for length
06FE D5       06890         PUSH    DE               ;Save end +1
06FF 1B       06900         DEC     DE               ;Adjust back
0700 3E0A     06910 HEXDEC2 LD      A,10             ;Base to convert to
0702 CDE306   06920         CALL    @DIV16           ;HL+A  =  HL/A
0705 C630     06930         ADD     A,'0'            ;Add ASCII to result
0707 12       06940         LD      (DE),A           ;  to user string
0708 1B       06950         DEC     DE               ;Move back
              06960 ;
              06970 ;       Check if done
              06980 ;
0709 7C       06990         LD      A,H              ;Get subtotal remainder
070A B5       07000         OR      L                ;Done?
070B 20F3     07010         JR      NZ,HEXDEC2       ;Go till completed
070D D1       07020         POP     DE               ;Restore end+1
070E C9       07030         RET                      ;Return Z
              07040 ;
070F          01540 *GET    CLOCKS:3                 ;Hardware task stuff
              07050 ;CLOCKS/ASM - LS-DOS 6.2
070F          07060         SUBTTL  '<Heartbeat & Bank handling>'
```

```
               07080 *MOD
               07090 ;
               07100 ;       Model IV time clock & blinking cursor
               07110 ;
070F 3C        07120 TIMETBL DB      60,60,24,30     ;Sec/min, min/hr, hr/day
     3C 18 1E
               07130 TIMTSK$
0713 3A970B    07140         LD      A,(CRSAVE)      ;If cursor not on,
0716 B7        07150         OR      A               ;  then don't blink
0717 217F00    07160         LD      HL,VFLAG$       ;Point to video flag
071A 2829      07170         JR      Z,$H2
               07180                                 ;Check if blinking
071C CB7E      07190         BIT     7,(HL)          ;Check system INHIBIT
071E CBBE      07200         RES     7,(HL)          ;Allow blink next time
0720 2023      07210         JR      NZ,$H2
0722 34        07220         INC     (HL)            ;Increment the counter
0723 CB5E      07230         BIT     3,(HL)          ;  & see if to 8
0725 281E      07240         JR      Z,$H2           ;Not this time
0727 CB9E      07250         RES     3,(HL)          ;Reset counter
0729 CB76      07260         BIT     6,(HL)          ;Check if SOLID cursor
072B 2802      07270         JR      Z,NOSOLID       ;If not, then blink
072D CBEE      07280         SET     5,(HL)          ;Force SOLID mode
072F CD1708    07290 NOSOLID CALL    ENADIS_DO_RAM   ;Bring up the video RAM
0732 7E        07300         LD      A,(HL)          ;Grab the toggle bit
0733 EE20      07310         XOR     20H             ;  and flip it
0735 77        07320         LD      (HL),A
0736 E620      07330         AND     20H             ;Was it on?
0738 ED5B950B  07340         LD      DE,(CURSOR)     ;Get the cursor pos
073C 3A970B    07350         LD      A,(CRSAVE)      ;  and char under cursor
073F 2003      07360         JR      NZ,$H1          ;Put character if flip on
0741 3A980B    07370         LD      A,(CRSCHAR)     ;  else put the cursor
0744 12        07380 $H1     LD      (DE),A          ;Put the char
0745 DD210F07  07390 $H2     LD      IX,TIMETBL      ;Point to data area
0749 DD3503    07400         DEC     (IX+3)          ;Count down by 30
074C C0        07410         RET     NZ              ;Back if not one second
               07420         IF      @HZ50
               07430         LD      (IX+3),25       ;Set for 50 hertz
               07440 HERTZ$  EQU     $-1
               07450         ELSE                    ;  else use 60 hertz
074D DD36031E  07460         LD      (IX+3),30       ;Reset for one second
0750           07470 HERTZ$  EQU     $-1
               07480         ENDIF
0751 CB66      07490         BIT     4,(HL)          ;Is clock on? (VFLAG$)
0753 2804      07500         JR      Z,$H3           ;Go if off
0755 118707    07510         LD      DE,CLOCK        ;Set to display clock
0758 D5        07520         PUSH    DE
0759 0603      07530 $H3     LD      B,3
075B 212D00    07540         LD      HL,TIME$
075E 110F07    07550         LD      DE,TIMETBL      ;Pt to max sec, min, hr
0761 34        07560 TIMER1  INC     (HL)            ;Bump time parm
0762 1A        07570         LD      A,(DE)
0763 96        07580         SUB     (HL)
0764 C0        07590         RET     NZ              ;Ret if not max
0765 77        07600         LD      (HL),A          ;  else set to 0
0766 2C        07610         INC     L               ;Pt to next parm
0767 1C        07620         INC     E
0768 10F7      07630         DJNZ    TIMER1          ;Loop thru 3 parms
               07640 ;
               07650 ;       Update date at midnight
               07660 ;
```

```
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
```

```
07B1 B307      08260          DW      $+2
07B3 2AAF07    08270          LD      HL,(PCSAVE$)
07B6 EB        08280          EX      DE,HL             ;Program counter to DE
07B7 CD1708    08290          CALL    ENADIS_DO_RAM     ;Bring up the video
07BA 213EF8    08300          LD      HL,CRTBGN$+62     ;Crt trace adr
               08310 ;
               08320 ;        Hexadecimal display routine
               08330 ;
07BD 7A        08340 @HEX16   LD      A,D               ;Convert reg D to
07BE CDC207    08350          CALL    @HEX8             ;  two hex digits
07C1 7B        08360          LD      A,E               ;Convert reg E to
07C2 F5        08370 @HEX8    PUSH    AF                ;  two hex digits
07C3 1F        08380          RRA                       ;Do left nybble first
07C4 1F        08390          RRA
07C5 1F        08400          RRA
07C6 1F        08410          RRA
07C7 CDCB07    08420          CALL    HXD1              ;Bits 0-3 stuffed in hex
07CA F1        08430          POP     AF                ;Reget the byte
07CB E60F      08440 HXD1     AND     0FH               ;  & use right nybble
07CD C690      08450          ADD     A,90H             ;Convert nybble to hex
07CF 27        08460          DAA
07D0 CE40      08470          ADC     A,40H
07D2 27        08480          DAA
07D3 77        08490          LD      (HL),A            ;Stuff in (HL)
07D4 23        08500          INC     HL
07D5 C9        08510          RET
               08520 ;
               08530 ;        Scan for PAUSE or BREAK & set KFLAG$
               08540 ;
F480           08550 SHIFT
               08560          IF      @USA
F401           08570 KB1
               08580          ENDIF
               08590          IF      @GERMAN
               08600 KB1
               08610          ENDIF
               08620          IF      @FRENCH
               08630 KB1
               08640          ENDIF
F440           08650 KB7
07D6 CD1708    08660 KCK@     CALL    ENADIS_DO_RAM     ;Bring up the keyboard
07D9 217400    08670          LD      HL,KFLAG$         ;Hang onto flag
07DC 3A80F4    08680          LD      A,(SHIFT)         ;P/u SHIFT row & ignore
07DF E607      08690          AND     7                 ;  CTRL key pressed
07E1 2F        08700          CPL
07E2 CB57      08710          BIT     2,A
07E4 C8        08720          RET     Z                 ;Back if CTRL
               08730 ;
               08740 ;        Set carry flag if a SHIFT key is down
               08750 ;
07E5 C601      08760          ADD     A,1               ;Set CF if no SHIFT
07E7 3F        08770          CCF                       ;Set CF if SHIFT
07E8 300B      08780          JR      NC,KCK1           ;No pause if no SHIFT
07EA 3A01F4    08790          LD      A,(KB1)           ;Test for "@"
               08800          IF      @USA
07ED CB47      08810          BIT     0,A
               08820          ENDIF
               08830          IF      @INTL
               08840          BIT     4,A               ;Foriegn keyboard
```

```
                08850          ENDIF
07EF 280B       08860          JR      Z,KCK1A          ;Bypass if no "@"
07F1 CBCE       08870          SET     1,(HL)           ;Turn on pause bit
07F3 1807       08880          JR      KCK1A
                08890 ;
                08900 ;        Inhibit test of unshifted BREAK if nested ENA_DO
                08910 ;
07F5 3A3508     08920 KCK1     LD      A,(OPREG_SV_PTR) ;If not at highest level
07F8 D66F       08930          SUB     OPREG_SV_AREA+1&0FFH ;  then don't allow
07FA 2009       08940          JR      NZ,KCK1B         ;  tasker BREAK handler
07FC 3A40F4     08950 KCK1A    LD      A,(KB7)          ;Check on BREAK & ENTER
07FF CB47       08960          BIT     0,A              ;Check on ENTER
0801 2802       08970          JR      Z,KCK1B          ;Go if not
0803 CBD6       08980          SET     2,(HL)           ;  else note set
0805 CB57       08990 KCK1B    BIT     2,A              ;Is <BREAK> depressed?
0807 F5         09000          PUSH    AF
0808 280B       09010          JR      Z,KCK2           ;Go if not
080A 3809       09020          JR      C,KCK2           ;Ignore if shifted
080C 3A7C00     09030          LD      A,(SFLAG$)       ;Permit break bit only
080F CB67       09040          BIT     4,A              ;  if BREAK enabled?
0811 2002       09050          JR      NZ,KCK2
0813 CBC6       09060          SET     0,(HL)           ;Turn on BREAK bit
0815 F1         09070 KCK2     POP     AF               ;C=shift, NZ=break
0816 C9         09080          RET
                09090 ;
                09100 ;        Routine to enable video RAM & change stack if necessary
                09110 ;
                09120 *MOD
                09130 ENADIS_DO_RAM
0817 F3         09140          DI                       ;Can't while we test stack
0818 226B08     09150          LD      (HLSAV),HL       ;Save HL but not on stack
081B F5         09160          PUSH    AF               ;Save AF
081C E1         09170          POP     HL
081D 226608     09180          LD      (AFSAV),HL
0820 21030C     09190          LD      HL,0C03H         ;Can't exceed X'F3FC'
0823 39         09200          ADD     HL,SP
0824 3009       09210          JR      NC,$I1
                09220 ;
                09230 ;        Switch to the system stack
                09240 ;
0826 E1         09250          POP     HL               ;Transfer RET address
0827 ED736308   09260          LD      (SPSAV),SP       ;Save stack pointer
082B 316003     09270          LD      SP,STACK$-20H    ;Keep room at top
082E E5         09280          PUSH    HL               ;Put RET back
082F 214608     09290 $I1      LD      HL,DIS_DO_RAM    ;Stack return to disable
0832 E3         09300          EX      (SP),HL          ;  video RAM below RET
0833 E5         09310          PUSH    HL
0834 216E08     09320          LD      HL,OPREG_SV_AREA
0835            09330 OPREG_SV_PTR     EQU     $-2
0837 23         09340          INC     HL               ;Get next save location
0838 3A7800     09350          LD      A,(OPREG$)       ;P/u port mask
083B 3002       09360          JR      NC,$I2           ;Bypass if NC (no stack switch)
083D E67F       09370          AND     7FH              ;Strip bit 7 to use as flag
083F 77         09380 $I2      LD      (HL),A           ;Save current state
0840 E6FC       09390          AND     0FCH             ;Strip SEL1 & SEL0
0842 F682       09400          OR      82H              ;Set SEL1,0 = (1,0) & NZ condx
0844 1812       09410          JR      DOOPREG          ;Set new assignment
                09420 ;
                09430 ;        Routine to disable video RAM
```

Heartbeat & Bank handling

```
                09440 ;
                09450 DIS_DO_RAM
0846 F3         09460          DI                       ;Interrupts off
0847 226B08     09470          LD      (HLSAV),HL       ;Save off of stack
084A F5         09480          PUSH    AF
084B E1         09490          POP     HL               ;Save AF
084C 226608     09500          LD      (AFSAV),HL
084F 2A3508     09510          LD      HL,(OPREG_SV_PTR)
0852 7E         09520          LD      A,(HL)           ;P/u previous state
0853 CB7F       09530          BIT     7,A              ;Test if we switch stack
0855 CBFF       09540          SET     7,A              ;Make sure PAGE is set
0857 2B         09550          DEC     HL
                09560 ;
0858 223508     09570 DOOPREG LD       (OPREG_SV_PTR),HL
085B 327800     09580          LD      (OPREG$),A       ;Restore port image
085E D384       09590                                   ;  and the port
0860 2003       09600          JR      NZ,$I3
                09610 ;
                09620 ;        Switch back to the old stack
                09630 ;
0862 310000     09640          LD      SP,$-$           ;Get the old stack
0863            09650 SPSAV    EQU     $-2
0865 210000     09660 $I3      LD      HL,$-$
0866            09670 AFSAV    EQU     $-2
0868 E5         09680          PUSH    HL               ;Restore AF
0869 F1         09690          POP     AF
086A 210000     09700          LD      HL,$-$           ;Restore HL
086B            09710 HLSAV    EQU     $-2
086D FB         09720          EI                       ;Interrupts back on
086E C9         09730          RET
086E            09740 OPREG_SV_AREA    EQU     $-1
086F 00         09750          DB      0,0,0,0,0,0,0,0
     00 00 00 00 00 00 00
                09760 ;
                09770 ;        Bank selection SVC handler
                09780 ;        HL=> transfer address for function B=0
                09790 ;        C => Bank request <0-2>; Set bit 7 to transfer
                09800 ;        B => Request function
                09810 ;                0 => Select bank C
                09820 ;                1 => Reset in-use bit of bank C
                09830 ;                2 => Test in-use bit of bank C
                09840 ;                3 => Set in-use bit of bank C
                09850 ;
                09860 *MOD
                09870 @BANK
0877 E67F       09880          AND     7FH              ;Strip possible bit 7
0879 FE03       09890          CP      2+1              ;Bank out of range?
087B D2ED0D     09900          JP      NC,PERR
087E 05         09910          DEC     B                ;Check option
087F FAB308     09920          JP      M,$J3            ;Go if bank select
0882 0E86       09930          LD      C,86H            ;Set for reset BUR$
0884 2819       09940          JR      Z,$J1            ;Go if function 1
0886 0E46       09950          LD      C,46H            ;Set for test BUR$
0888 05         09960          DEC     B
0889 2814       09970          JR      Z,$J1            ;Go if function 2
088B 05         09980          DEC     B
088C 2809       09990          JR      Z,$J0            ;Go on set BUR$
088E 05         10000          DEC     B
088F C2ED0D     10010 PERRX    JP      NZ,PERR          ;SVC parameter error
```

Heartbeat & Bank handling

```
0892 3A0202   10020       LD     A,(LBANK$)      ;P/u current bank
0895 BF       10030       CP     A
0896 C9       10040       RET
0897 47       10050 $J0   LD     B,A             ;Save the bank requested
0898 CD9F08   10060       CALL   $J1             ;Test if in use already
089B C0       10070       RET    NZ              ;Back if error
089C 78       10080       LD     A,B             ;Reget the request #
089D 0EC6     10090       LD     C,0C6H          ;Set for set BUR$
089F E607     10100 $J1   AND    7               ;Strip to bank 0-7
08A1 07       10110       RLCA                   ;Shift <0-2> to <3-5>
08A2 07       10120       RLCA
08A3 07       10130       RLCA
08A4 B1       10140       OR     C               ;Merge the code type
08A5 32B008   10150       LD     ($J2+1),A       ;Change the OP code
08A8 AF       10160       XOR    A               ;Init Z-flag
08A9 3E08     10170       LD     A,8             ;Init "Device not avail
08AB E5       10180       PUSH   HL              ;Don't alter HL
08AC 210002   10190       LD     HL,BUR$         ;Point to bank-used-RAM
08AF CB46     10200 $J2   BIT    0,(HL)          ;*** Modified instruction
08B1 E1       10210       POP    HL
08B2 C9       10220       RET
08B3 E5       10230 $J3   PUSH   HL              ;Ck if stack is in upper
08B4 210580   10240       LD     HL,8005H        ;  bank area
08B7 39       10250       ADD    HL,SP
08B8 E1       10260       POP    HL
08B9 DAED0D   10270       JP     C,PERR          ;Error if > X'7FFE'
08BC FE01     10280       CP     1               ;Change <0, 1, 2, 3>
08BE 17       10290       RLA                    ;  to <1, 2, 4, 6>
08BF 47       10300       LD     B,A             ;  & save for later
08C0 3A0102   10310       LD     A,(BAR$)        ;P/u Bank Avail Ram
08C3 A0       10320       AND    B               ;Is the bank installed?
08C4 20C9     10330       JR     NZ,PERRX        ;Error if not in machine
08C6 78       10340       LD     A,B             ;Get the requested bank
08C7 1F       10350       RRA                    ;Change <1, 2, 4> to
08C8 3F       10360       CCF                    ;  <0, 2, 3> {CF on 0
08C9 CE00     10370       ADC    A,0             ;  switched to 2 & 4}
08CB 07       10380       RLCA                   ;Shift bits 0-1
08CC 07       10390       RLCA                   ;  to 4-5 (MBIT0,1)
08CD 07       10400       RLCA
08CE 07       10410       RLCA
08CF 47       10420       LD     B,A             ;Save bit mask
08D0 3A7800   10430       LD     A,(OPREG$)      ;P/u current memory
08D3 E68F     10440       AND    08FH            ;  configuration &
08D5 B0       10450       OR     B               ;  mask off old &
08D6 327800   10460       LD     (OPREG$),A      ;  merge the new
08D9 D384     10470                              ;Switch the hardware
08DB 3A0202   10480       LD     A,(LBANK$)      ;Get old bank #
08DE 47       10490       LD     B,A             ;  & save it
08DF 79       10500       LD     A,C             ;P/u new bank #
08E0 E67F     10510       AND    7FH             ;Strip any bit-7
08E2 320202   10520       LD     (LBANK$),A      ;  & save new bank #
08E5 A9       10530       XOR    C               ;Keep bit-7
08E6 B0       10540       OR     B               ;Merge in new bank #
08E7 4F       10550       LD     C,A             ;  & replace into C
08E8 CB79     10560       BIT    7,C             ;Transfer to new bank?
08EA 0600     10570       LD     B,0             ;Init for invoke later
08EC C8       10580       RET    Z               ;No if bit-7 = 0
08ED E3       10590       EX     (SP),HL         ;Exchange RET with new
08EE BF       10600       CP     A               ;  transfer & go to it
```

Heartbeat & Bank handling

```
08EF C9         10610          RET
08F0            01550 @$SYS    EQU     $               ;Pointer for @GTMOD
                01560          IF      @USA
08F0            01570 *GET     KIDVR:3                 ;Keyboard driver
                10620 ;KIDVR/ASM - LS-DOS 6.2
08F0            10630          SUBTTL  '<Keyboard Driver>'
```

```
                10650 *MOD
                10660 ;
000A            10670 LF       EQU      10
000D            10680 CR       EQU      13
F401            10690 KB0                               ;Row 0 RAM address
F440            10700 KB6                               ;Row 6 RAM address
F480            10710 SHIFT                             ;Row 7 RAM address
                10720 ;
08F0 1831       10730 KIDVR    JR       KIBGN           ;Branch around linkage
08F2 870B       10740          DW       KILAST          ;Last byte used
08F4 03         10750          DB       3,'$KI'
     24 4B 49
08F8 0802       10760          DW       KIDCB$          ;Pointer to DCB
08FA 0000       10770          DW       0               ;Spare
08FC 00         10780 KIDATA$ DB        0               ;Last key entered
08FD 00         10790          DB       0               ;Repeat time check
0002            10800 RPTINIT EQU       $-KIDATA$
08FE 16         10810          DB       22              ;22 * 33.3ms = .733 sec
0003            10820 RPTRATE EQU       $-KIDATA$
08FF 02         10830          DB       2               ;2 x RTC rate
0004            10840 KBROW0   EQU      $-KIDATA$
0900 FF         10850          DB       -1,-1,-1,-1     ;Image of rows 0-3
     FF FF FF
0008            10860 KBROW4   EQU      $-KIDATA$
0904 FF         10870          DB       -1,-1           ;Image of rows 4-5
     FF
000A            10880 KBROW6   EQU      $-KIDATA$
0906 FF         10890          DB       -1,-1           ;Image of rows 6-7
     FF
                10900 ;
                10910 ;        Conversion table for keyboard row 7/8
                10920 ;
0908 0D         10930 KBTBL    DB       CR,1DH,1FH,1FH  ;<ENTER> <CLEAR>
     1D 1F 1F
090C 80         10940          DB       80H,0,0BH,1BH   ;<BREAK> <UPARW>
     00 0B 1B
0910 0A         10950          DB       LF,1AH,8,18H    ;<DNARW> <LTARW>
     1A 08 18
0914 09         10960          DB       9,19H,20H,20H   ;<RTARW> <SPACE>
     19 20 20
0918 81         10970          DB       81H,91H,82H,92H ;<F1> <F2>
     91 82 92
091C 83         10980          DB       83H,93H         ;<F3>
     93
                10990 ;
                11000 ;        Table to generate 5B-5F, 7B-7F
                11010 ;
091E 2C         11020 SPCLTB   DB       ',/.;',CR
     2F 2E 3B 0D
                11030 ;
                11040 ;        Entry to keyboard driver
                11050 ;
0923 79         11060 KIBGN    LD       A,C             ;Get the character
0924 F5         11070          PUSH     AF              ;Save flags
0925 CD8900     11080          CALL     @KITSK          ;Hook for KI task
0928 F1         11090          POP      AF
                11100 ;
                11110 ;        Screen print (Control-*) processing
                11120 ;
0929 CDD50A     11130          CALL     TYPAHD          ;Chain downstream
```

Keyboard Driver

```
092C D0         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 $K1     POP     AF              ;Clean the stack
0935 3A6D00     11240         LD      A,(DFLAG$)      ;Check on Graphic bit
0938 07         11250         RLCA
0939 3E3E       11260         LD      A,3EH           ;Init for LD a,'.'
093B 3002       11270         JR      NC,$+4          ;Go if not Graphic
093D 3EFE       11280         LD      A,0FEH          ;Change to CPR n
093F 325C09     11290         LD      ($K4),A         ;Stuff cpr or ld
0942 217400     11300         LD      HL,KFLAG$       ;Reset the BREAK bit
0945 CB86       11310         RES     0,(HL)
0947 E5         11320         PUSH    HL              ;Save on stack
0948 210000     11330         LD      HL,0            ;Init for row,col
094B 0601       11340 $K2     LD      B,1             ;Get a character at the
094D CD990B     11350         CALL    @VDCTL          ;  row-H, col-L
0950 2027       11360         JR      NZ,$K6          ;Go on error
0952 FE20       11370         CP      20H
0954 3002       11380         JR      NC,$+4          ;Convert control codes
0956 C640       11390         ADD     A,40H           ;  to cap A-Z, +
0958 FE80       11400         CP      80H             ;Cvrt anything from X'80'
095A 3802       11410         JR      C,$K5           ;  thru X'FF' to a '.'
095C 3E2E       11420 $K4     LD      A,'.'           ;  unless graphic bit set
095E CD3D06     11430 $K5     CALL    @PRT            ;Print the char & loop
0961 2016       11440         JR      NZ,$K6
0963 2C         11450         INC     L               ;Bump column counter
0964 7D         11460         LD      A,L             ;Check for end-of-line
0965 D650       11470         SUB     80
0967 20E2       11480         JR      NZ,$K2          ;Loop if not EOL
0969 6F         11490         LD      L,A             ;Reset to column 0
096A 2D         11500         DEC     L               ;Adj for CR force
096B E3         11510         EX      (SP),HL         ;Get KFLAG$
096C CB46       11520         BIT     0,(HL)          ;Exit with A=0 on
096E E3         11530         EX      (SP),HL         ;  entrance of BREAK
096F 2008       11540         JR      NZ,$K6
0971 24         11550         INC     H               ;Bump row counter
0972 7C         11560         LD      A,H             ;Test for end of screen
0973 FE18       11570         CP      24
0975 3E0D       11580         LD      A,CR
0977 20E5       11590         JR      NZ,$K5          ;Put the CR & loop
0979 3E0D       11600 $K6     LD      A,CR            ;Close out with CR if
097B CD3D06     11610         CALL    @PRT            ;  BREAK key detected
097E E1         11620         POP     HL              ;Pop the KFLAG
097F CB86       11630         RES     0,(HL)          ;  & reset BREAK bit
0981 1832       11640         JR      NOCHAR
                11650 ;
                11660 ;       Driver to scan the keyboard
                11670 ;
                11680 *MOD
0983 DD21FC08   11690 KISCAN  LD      IX,KIDATA$      ;Point to data area
0987 210009     11700         LD      HL,KIDATA$+KBROW0 ;Load kbd image start
098A 0101F4     11710         LD      BC,KB0          ;Load start of keyboard
098D 1600       11720         LD      D,0             ;Zero the key counter
```

Keyboard Driver

```
098F 0A         11730 $L1     LD      A,(BC)          ;Load 1st char from kbd
0990 5F         11740         LD      E,A
0991 AE         11750         XOR     (HL)            ;XOR with old value
0992 2026       11760         JR      NZ,$L2          ;Go if different
0994 14         11770         INC     D               ;Bump key counter
0995 23         11780         INC     HL              ;Bump image pointer
0996 CB01       11790         RLC     C               ;Go to next row
0998 F28F09     11800         JP      P,$L1           ;Loop until end of rows
099B 0A         11810         LD      A,(BC)          ;Get row 7
099C E678       11820         AND     078H            ;Strip SHIFT, CTL
099E 5F         11830         LD      E,A
099F AE         11840         XOR     (HL)
09A0 2018       11850         JR      NZ,$L2
09A2 DD7E00     11860         LD      A,(IX+0)        ;Key down? It's same as
09A5 B7         11870         OR      A               ; the last if so
09A6 280D       11880         JR      Z,NOCHAR        ;Ret if no key
09A8 3A2C00     11890         LD      A,(TIMER$)      ;Do we repeat the
09AB DD9601     11900         SUB     (IX+1)          ;  same key?
09AE 286E       11910         JR      Z,$L10          ;Go repeat if time up
09B0 DD9602     11920         SUB     (IX+RPTINIT)    ;Beyond 0.75 seconds?
09B3 3869       11930         JR      C,$L10          ;Go if yes
09B5 F601       11940 NOCHAR  OR      1               ;Else don't repeat
09B7 3E00       11950         LD      A,0             ;Show NZ with A=0
09B9 C9         11960         RET
                11970 ;
                11980 ;       Found change in key matrix
                11990 ;
09BA 73         12000 $L2     LD      (HL),E          ;Stuff KB image with new
09BB A3         12010         AND     E               ;  KB row value
09BC CA880A     12020         JP      Z,NOKEY         ;Go if new is none
                12030 ;
                12040 ;       Convert the depressed key
                12050 ;
09BF 5F         12060         LD      E,A             ;Save the active bit
09C0 7A         12070         LD      A,D             ;Calculate 8 * row
09C1 07         12080         RLCA
09C2 07         12090         RLCA
09C3 07         12100         RLCA
09C4 57         12110         LD      D,A             ;Save 8 * row
09C5 0E01       12120         LD      C,1             ;Add 8 * row + column
09C7 79         12130 $L3     LD      A,C
09C8 A3         12140         AND     E               ;Check if bits match
09C9 2019       12150         JR      NZ,$L6          ;Go if match
09CB 14         12160         INC     D               ;  else bump value
09CC CB01       12170         RLC     C               ;Shift compare bit
09CE 18F7       12180         JR      $L3             ;Loop to test next
                12190 ;
                12200 ;       Key pressed was not an alpha
                12210 ;
09D0 D690       12220 $L4     SUB     90H             ;Adjust for non-alpha
09D2 3052       12230         JR      NC,$L9          ;Go if special key
09D4 C640       12240         ADD     A,40H           ;Cvrt to numeric/symbol
09D6 FE3C       12250         CP      3CH             ;Manipulate to get
09D8 3802       12260         JR      C,$L5           ;  proper code
09DA EE10       12270         XOR     10H
09DC CB43       12280 $L5     BIT     0,E             ;Check SHIFT
09DE 2860       12290         JR      Z,$L11          ;Go if unshift
09E0 EE10       12300         XOR     10H             ;  else adjust for SHIFT
09E2 185C       12310         JR      $L11
```

```
              12320 ;
              12330 ;       Found a key - Set up the function codes
              12340 ;
09E4 3A80F4   12350 $L6     LD     A,(SHIFT)        ;P/u the SHIFT key
09E7 5F       12360         LD     E,A              ;Merge RH & LH Shift keys
09E8 E602     12370         AND    2                ;Only merge bit 1
09EA 0F       12380         RRCA                    ;Bit 1 to bit 0
09EB B3       12390         OR     E                ;Merge bits 0 & 1
09EC 5F       12400         LD     E,A              ;Value of (RHorLH) shift
09ED 7A       12410         LD     A,D              ;Load semi-converted
09EE C660     12420         ADD    A,60H            ;If alpha, convert to
09F0 FE80     12430         CP     80H              ;  correct value
09F2 217400   12440         LD     HL,KFLAG$
09F5 30D9     12450         JR     NC,$L4           ;Go if not alpha
              12460 ;
              12470 ;       Alpha <@-Z> - If caps lock or <SHIFT>,
              12480 ;       Convert to caps unless CLEAR
              12490 ;
09F7 CB53     12500         BIT    2,E              ;CTRL key down?
09F9 2018     12510         JR     NZ,CTLA2Z        ;CTRL sets <00-1A>
09FB FE60     12520         CP     60H              ;Invert @ and `
09FD 2004     12530         JR     NZ,$L7
09FF EE20     12540         XOR    20H              ;Invert & bypass test
0A01 180A     12550         JR     $L8              ;  for CAPs lock
0A03 DDCB0A4E 12560 $L7     BIT    1,(IX+KBROW6)    ;If CLEAR, don't test
0A07 2004     12570         JR     NZ,$L8           ;  for CAPs lock
0A09 CB6E     12580         BIT    5,(HL)           ;Caps lock?
0A0B 2031     12590         JR     NZ,TGLCASE
0A0D CB43     12600 $L8     BIT    0,E              ;Shift key down?
0A0F 282F     12610         JR     Z,$L11           ;Bypass if not shifted
0A11 182B     12620         JR     TGLCASE          ;Convert to upper case
0A13 D660     12630 CTLA2Z  SUB    60H              ;Convert CTRL A-Z
0A15 2029     12640         JR     NZ,$L11          ;Go on A-Z
0A17 CB43     12650         BIT    0,E              ;Shifted?
0A19 37       12660         SCF                     ;Set C-flag for CTL-@
0A1A C8       12670         RET    Z                ;  & return if unshifted
0A1B 3E1C     12680         LD     A,1CH            ;  else set EOF error
0A1D C9       12690         RET
0A1E 3A2C00   12700 $L10    LD     A,(TIMER$)       ;Advance time check
0A21 DD8603   12710         ADD    A,(IX+RPTRATE)   ;  by 0.067 seconds
0A24 1872     12720         JR     $L12             ;Go output the key
              12730 ;
              12740 ;       Special keys - rows 6 & 7
              12750 ;
0A26 FE0B     12760 $L9     CP     11               ;Compress F1-F3 keys
0A28 284F     12770         JR     Z,CAPSKEY        ;  while checking for CAP
0A2A 3802     12780         JR     C,$+4            ;  F1-F3 to 8-10
0A2C D604     12790         SUB    4
0A2E 210809   12800         LD     HL,KBTBL         ;Pt to special char table
0A31 07       12810         RLCA                    ;Index into table,
0A32 CB43     12820         BIT    0,E              ;  shifted code is +1
0A34 2801     12830         JR     Z,$+3
0A36 3C       12840         INC    A
0A37 4F       12850         LD     C,A              ;Index the table
0A38 0600     12860         LD     B,0
0A3A 09       12870         ADD    HL,BC
0A3B 7E       12880         LD     A,(HL)           ;Load char from table
0A3C 1802     12890         JR     $L11             ;Bypass restore of char
0A3E EE20     12900 TGLCASE XOR    20H              ;Toggle the case
```

Keyboard Driver

```
0A40 FE80      12910 $L11    CP      80H              ;BREAK key?
0A42 200F      12920         JR      NZ,$L11A         ;Ck on <BREAK> disable
0A44 217C00    12930         LD      HL,SFLAG$        ;Break disabled?
0A47 CB66      12940         BIT     4,(HL)
0A49 2007      12950         JR      NZ,$L11B         ;Don't set bit if disabl
0A4B 217400    12960         LD      HL,KFLAG$
0A4E CBC6      12970         SET     0,(HL)           ;  otherwise set it
0A50 1801      12980         JR      $L11A
0A52 17        12990 $L11B   RLA                      ;Rotate bit-7 out
0A53 DDCB0A4E  13000 $L11A   BIT     1,(IX+KBROW6)    ;CLEAR key pressed?
0A57 280E      13010         JR      Z,NOTALPH        ;Go if not down
0A59 57        13020         LD      D,A              ;Save code
0A5A CBAF      13030         RES     5,A              ;Set to upper-case for
0A5C D641      13040         SUB     'A'              ;  test A-Z
0A5E FE1A      13050         CP      'Z'-'A'+1
0A60 7A        13060         LD      A,D              ;Get back actual char
0A61 3002      13070         JR      NC,$+4           ;Go if not A-Z
0A63 EE20      13080         XOR     20H              ;Shift keyboard case
0A65 F680      13090         OR      80H              ;Set bit 7 for CLEAR key
0A67 CB43      13100 NOTALPH BIT     0,E              ;SHIFT key down?
0A69 2819      13110         JR      Z,FIXCLR         ;Go if not
0A6B FE9F      13120 GOTSHFT CP      9FH              ;Shift-clear?
0A6D 2813      13130         JR      Z,FIXSCL         ;Go if so
0A6F FE20      13140 TSTSPA  CP      20H              ;Shift 0 or shift sp?
0A71 2016      13150         JR      NZ,KEYOK         ;Go if not
0A73 DDCB0846  13160         BIT     0,(IX+KBROW4)    ;Ck zero key
0A77 2810      13170         JR      Z,KEYOK          ;Go if not down
               13180 ;
               13190 ;       Toggle the caps lock bit in the KFLAG$
               13200 ;
0A79 3E20      13210 CAPSKEY LD      A,20H            ;CAPs wasn't 20H
0A7B 217400    13220 CASHK$  LD      HL,KFLAG$        ;Reverse case by
0A7E AE        13230         XOR     (HL)             ;  flipping bit 5
0A7F 77        13240         LD      (HL),A
0A80 1806      13250         JR      NOKEY
0A82 EE80      13260 FIXSCL  XOR     80H              ;Reset bit 7
0A84 FE9F      13270 FIXCLR  CP      9FH              ;Clear key?
0A86 2001      13280         JR      NZ,KEYOK         ;Go if not
0A88 AF        13290 NOKEY   XOR     A
0A89 DD7700    13300 KEYOK   LD      (IX+0),A
0A8C 018401    13310         LD      BC,0184H         ;Delay
0A8F CD8203    13320 TYPHK$  CALL    PAUSE@
0A92 3A2C00    13330         LD      A,(TIMER$)       ;Set initialization
0A95 DD8602    13340 DELAY2  ADD     A,(IX+RPTINIT)   ;  repeat key delay
0A98 DD7701    13350 $L12    LD      (IX+1),A         ;Save new repeat time
0A9B DD7E00    13360         LD      A,(IX+0)         ;Check if any key
0A9E B7        13370         OR      A                ;  code was saved
0A9F CAB509    13380         JP      Z,NOCHAR         ;Ret if none
0AA2 CB53      13390         BIT     2,E              ;Shift key down?
0AA4 37        13400         SCF                      ;Init carry
0AA5 2004      13410         JR      NZ,SPECL         ;Ret if CTRL
0AA7 3F        13420         CCF
0AA8 CB7F      13430 DVREXIT BIT     7,A              ;Z-flag set on non-CLEAR
0AAA C8        13440         RET     Z                ;Go if not CLEAR+key
0AAB F5        13450 SPECL   PUSH    AF               ;Save code
0AAC 211E09    13460 $L13    LD      HL,SPCLTB        ;Special char table
0AAF CBBF      13470         RES     7,A              ;Turn off "CLEAR"
0AB1 015B05    13480         LD      BC,5<8!5BH       ;5 chars, starting char
0AB4 3001      13490         JR      NC,$+3           ;  if not CTRL
```

```
0AB6 05        13500          DEC     B                 ;  else only 4
0AB7 BE        13510 SPCLLP   CP      (HL)              ;Is this it?
0AB8 2812      13520          JR      Z,HIT             ;Go if so
0ABA EE10      13530          XOR     10H               ;Flip shift state
0ABC BE        13540          CP      (HL)              ;Is that it?
0ABD 280B      13550          JR      Z,HITWS           ;Go if so
0ABF EE10      13560          XOR     10H               ;Flip back
0AC1 23        13570          INC     HL                ;Bump specl table ptr
0AC2 0C        13580          INC     C                 ;Bump "convert to" char
0AC3 10F2      13590          DJNZ    SPCLLP            ;Loop through table
0AC5 F1        13600          POP     AF                ;Not found in table
0AC6 380A      13610          JR      C,CKCTL2          ;Ck CTL for C-flag
0AC8 BF        13620 CKCTL1   CP      A                 ;Set Z-flag
0AC9 C9        13630          RET
0ACA CBE9      13640 HITWS    SET     5,C               ;Move to LC set
0ACC F1        13650 HIT      POP     AF                ;Restore orig char
0ACD 79        13660          LD      A,C               ;Load converted one
0ACE 30F8      13670 CKCTL    JR      NC,CKCTL1         ;Go if ctl key not down
0AD0 E61F      13680          AND     1FH               ;Force ctl code
0AD2 BF        13690 CKCTL2   CP      A                 ;Set Z-flag
0AD3 37        13700          SCF                       ;Set C-flag for CTRL
0AD4 C9        13710          RET
               13720 ;
               13730 ;        Check the type ahead buffer for any character
               13740 ;
               13750 *MOD
               13760 TYPAHD
0AD5 CD1708    13770          CALL    ENADIS_DO_RAM     ;Bring up Keyboard ram
0AD8 2180FF    13780          LD      HL,TYPBUF         ;P/u start of type buffer
0ADB 36FF      13790          LD      (HL),0FFH         ;Turn off type ahead
0ADD 381D      13800          JR      C,$M1             ;Go on @GET
0ADF 2842      13810          JR      Z,TYPON           ;No PUT to *KI
0AE1 FE03      13820          CP      3                 ;CTL 3 function?
0AE3 CA650B    13830          JP      Z,CLRTYP          ;Clear buffer if so
0AE6 3C        13840          INC     A
0AE7 2803      13850          JR      Z,CTLFF           ;Go if CTL 255 function
0AE9 AF        13860          XOR     A                 ;Nothing done, No error
0AEA 1837      13870          JR      TYPON
               13880 ;
               13890 ;        Handle CTL-255 - scan keyboard into user rowbuf
               13900 ;
               13910 CTLFF
0AEC 2101F4    13920          LD      HL,KB0            ;Start of keyboard image
0AEF 0608      13930          LD      B,8               ;Do 8 rows
0AF1 7E        13940 $M0      LD      A,(HL)            ;P/u the image
0AF2 FD7700    13950          LD      (IY),A            ;  and xfer to user buffer
0AF5 FD23      13960          INC     IY
0AF7 CB15      13970          RL      L
0AF9 10F6      13980          DJNZ    $M0
0AFB C9        13990          RET
               14000 ;
0AFC E5        14010 $M1      PUSH    HL
0AFD 23        14020          INC     HL                ;Bump to PUT pointer
0AFE 7E        14030          LD      A,(HL)            ;  & pick it up
0AFF 23        14040          INC     HL                ;Bump to GET pointer
0B00 BE        14050          CP      (HL)              ;The same?
0B01 281C      14060          JR      Z,$M4             ;Go if so
0B03 E5        14070          PUSH    HL                ;Save pointer to GETPTR
0B04 5E        14080          LD      E,(HL)            ;P/u offset to buffer
```

Keyboard Driver

```
0B05 23         14090          INC     HL              ;Pt to buffer start
0B06 1600       14100          LD      D,0             ;Add offset to start
0B08 19         14110          ADD     HL,DE           ;  to point to char posn
0B09 46         14120          LD      B,(HL)          ;GET the stored char
0B0A E1         14130          POP     HL              ;Rcvr GETPTR
0B0B 34         14140          INC     (HL)            ;Bump by 1 for char
0B0C 3E50       14150          LD      A,80            ;Check for >80
0B0E BE         14160          CP      (HL)            ;  after INC
0B0F 3002       14170          JR      NC,$M2          ;Go if not at end
0B11 3600       14180          LD      (HL),0          ;Reset to start of buf
0B13 7E         14190 $M2      LD      A,(HL)          ;If we emptied the
0B14 2B         14200          DEC     HL              ;  type-ahead buffer,
0B15 BE         14210          CP      (HL)            ;  update KFLAG$
0B16 CC6A0B     14220          CALL    Z,R7KFLG        ;Reset bit-7 if empty
0B19 E1         14230          POP     HL              ;Pointed to & get switch
0B1A 3600       14240          LD      (HL),0          ;Turn type back on
0B1C 78         14250          LD      A,B             ;Transfer char/flag
0B1D BF         14260          CP      A               ;Set flag "Z"
0B1E C9         14270          RET
                14280 ;
                14290 ;        No character in type ahead buffer - get from kbd
                14300 ;
0B1F CD8309     14310 $M4      CALL    KISCAN          ;Call keyboard driver
0B22 E1         14320          POP     HL              ;Rcvr switch
0B23 3600       14330 TYPON    LD      (HL),0          ;Type ahead back on
0B25 C9         14340          RET
                14350 ;
                14360 ;        Type ahead task 10 - scans keyboard & saves key
                14370 ;
0B26 280B       14380 TYPTSK$ DW       $M5             ;Task entry for processor
0B28 3A6D00     14390 $M5      LD      A,(DFLAG$)      ;If type-ahead suppressed
0B2B E602       14400          AND     2H              ;  then return
0B2D C8         14410          RET     Z
0B2E CD1708     14420          CALL    ENADIS_DO_RAM   ;Bring up the keyboard
0B31 2180FF     14430          LD      HL,TYPBUF       ;P/u type switch
0B34 7E         14440          LD      A,(HL)          ;If previous driver is
0B35 B7         14450          OR      A               ;  currently executing,
0B36 C0         14460          RET     NZ              ;  do not stack more keys
0B37 23         14470          INC     HL              ;Bump to PUTPTR
0B38 E5         14480          PUSH    HL              ;  & save it
0B39 CD8309     14490 KIHOOK   CALL    KISCAN          ;  and scan for a character
0B3C E1         14500          POP     HL
0B3D C0         14510          RET     NZ              ;Ret if no char
0B3E F5         14520          PUSH    AF              ;  else xfer char
0B3F C1         14530          POP     BC              ;  & flag to BC
0B40 FE80       14540          CP      80H             ;Check for <BREAK>
0B42 F5         14550          PUSH    AF
0B43 E5         14560          PUSH    HL
0B44 CC660B     14570          CALL    Z,$M6           ;If so clear type buf
0B47 E1         14580          POP     HL              ;Restore
0B48 F1         14590          POP     AF
0B49 FEC0       14600          CP      0C0H            ;If CLEAR @, reset keybuf
0B4B 2819       14610          JR      Z,$M6
0B4D 5E         14620          LD      E,(HL)          ;P/u PUTPTR & compare
0B4E 7B         14630          LD      A,E             ;GETPTR
0B4F 23         14640          INC     HL
0B50 BE         14650          CP      (HL)
0B51 2821       14660          JR      Z,$M8           ;Jump if key buffer empty
0B53 3A2C00     14670          LD      A,(TIMER$)      ;Check if we expired the
```

Keyboard Driver

```
0B56 DD8603    14680          ADD    A,(IX+RPTRATE)  ;  time interval between
0B59 DDBE01    14690          CP     (IX+1)          ;  repeating keys
0B5C 2012      14700          JR     NZ,$M7          ;Go if time not up
0B5E DD8603    14710          ADD    A,(IX+RPTRATE)  ;Re-adjust time check so
0B61 DD7701    14720          LD     (IX+1),A        ;  we don't repeat in
0B64 C9        14730          RET                    ;  type-ahead task
               14740 ;
               14750 ;        CLEAR @ control key entered, clear the buffer
               14760 ;
0B65 23        14770 CLRTYP   INC    HL              ;Bump to PUT pointer
0B66 AF        14780 $M6      XOR    A
0B67 77        14790          LD     (HL),A          ;1st PUT is loc'n 0
0B68 23        14800          INC    HL              ;Pt to GETPTR
0B69 77        14810          LD     (HL),A          ;1st GET is loc'n 0
0B6A 217400    14820 R7KFLG   LD     HL,KFLAG$       ;Show buffer empty
0B6D CBBE      14830          RES    7,(HL)
0B6F C9        14840          RET
               14850 ;
               14860 ;        Char to stuff - check if buffer will overflow
               14870 ;
0B70 7B        14880 $M7      LD     A,E             ;P/u current PUT pointer
0B71 3C        14890          INC    A               ;If the next loc'n wraps
0B72 BE        14900          CP     (HL)            ;  to the GET loc'n,
0B73 C8        14910          RET    Z               ;  don't permit overrun
0B74 E5        14920 $M8      PUSH   HL              ;Save ptr to GETPTR
0B75 23        14930          INC    HL              ;Pt to start of keybuf
0B76 1600      14940          LD     D,0             ;  & calculate PUT loc'n
0B78 19        14950          ADD    HL,DE
0B79 70        14960          LD     (HL),B          ;Store the char
0B7A 217400    14970          LD     HL,KFLAG$       ;Show type buffer
0B7D CBFE      14980          SET    7,(HL)          ;  is not empty
0B7F E1        14990          POP    HL              ;Rcvr ptr to GETPTR
0B80 2B        15000          DEC    HL              ;Backup to PUTPTR
0B81 34        15010          INC    (HL)            ;Bump past the char
0B82 3E50      15020          LD     A,80            ;Check for >80
0B84 BE        15030          CP     (HL)
0B85 D0        15040          RET    NC              ;Back if not over 80
0B86 72        15050          LD     (HL),D          ;  else reset to 1st
0B87 C9        15060          RET                    ;  position in buf (0)
               15070 ;
               15080 ;        Type ahead buffer area
               15090 ;
FF80           15100 TYPBUF
               15110 ;
               15120 ;        TYPBUF+0 = On/Off Flag
               15130 ;        TYPBUF+1 = Storage pointer
               15140 ;        TYPBUF+2 = Retrieve pointer
               15150 ;        TYPBUF+3 = Start of actual buffer
               15160 ;
0B87           15170 KILAST   EQU    $-1
               01580          ENDIF
               01590          IF     @GERMAN
               01600 FREN     EQU    00
               01610 GERM     EQU    -1
               01630          ENDIF
               01640          IF     @FRENCH
               01650 FREN     EQU    -1
               01660 GERM     EQU    00
               01680          ENDIF
```

Keyboard Driver

```
0B88          01690 *GET    DODVR:3                 ;Video driver
              15180 ;DODVR/ASM - LS-DOS 6.2
0B88          15190         SUBTTL  '<Video Driver>'
```

```
              15210 *MOD
0084          15220 @OPREG                                ;Mem mgt & video control
0088          15230 CRTCADD                               ;CRTC address port
0089          15240 CRTCDAT                               ;CRTC data port
0050          15250 LINESIZ EQU     80
0018          15260 NUMROWS EQU     24
FFB0          15270 NEGLINE EQU     -LINESIZ
0780          15280 CRTSIZE EQU     LINESIZ*NUMROWS
0800          15290 RAMSIZE EQU     2048
F800          15300 CRTBGN$
FF7F          15310 CRTEND  EQU     CRTBGN$+CRTSIZE-1
              15320 ;
              15330 ;               Driver entry point
              15340 ;
0B88 1812     15350 DODVR   JR      DOBGN           ;Branch around linkage
0B8A 000E     15360         DW      DOEND           ;Last memory location used
0B8C 03       15370         DB      3,'$DO'
     24 44 4F
0B90 1002     15380         DW      DODCB$          ;DCB used
0B92 0000     15390         DW      0               ;Reserved
0B94          15400 DODATA$ EQU     $
0000          15410 DO_MASK EQU     $-DODATA$
0007          15420 SCRPROT EQU     7               ;Bits 0-2: scroll protect
0003          15430 TABS    EQU     3               ;Bit 3: 0=tabs, 1=chars
0004          15440 CTL     EQU     4               ;Bit 4, display controls
              15450         IF      @USA
0B94 00       15460         DB      0               ;Tab/Spec, Scroll protect
              15470         ENDIF
              15480         IF      @INTL
              15490         DB      08              ;Space compression off
              15500         ENDIF
0B95 00F8     15510 CURSOR  DW      CRTBGN$
0B97 20       15520 CRSAVE  DB      20H             ;Character under cursor
0B98 5F       15530 CRSCHAR DB      '_'             ;Cursor character
              15540 ;
              15550 ;               Entry from SVC 15, @VDCTL
              15560 ;
0B99 C3420D   15570 @VDCTL  JP      @_VDCTL
              15580 ;
              15590 ;               Continue regular driver functions
              15600 ;
0B9C DD21940B 15610 DOBGN   LD      IX,DODATA$
0BA0 CD1708   15620         CALL    ENADIS_DO_RAM   ;Bring up the video RAM
0BA3 DAB30D   15630         JP      C,$N0           ;Go on 'GET' request
0BA6 CDB30D   15640         CALL    $N0             ;Handle cursor
0BA9 C5       15650         PUSH    BC              ;Need to save C
0BAA 79       15660         LD      A,C             ;Get char to display
0BAB DDCB0066 15670         BIT     CTL,(IX+DO_MASK) ;Display controls set?
0BAF 2009     15680         JR      NZ,$N1A         ;Go if so
0BB1 B7       15690         OR      A               ;Char a 0?
0BB2 CA9F0C   15700         JP      Z,TGGLCTL       ;Switch Bit CTL if so
0BB5 FE20     15710         CP      20H             ;Video control char?
0BB7 DA440C   15720         JP      C,DO_CONTROL    ;Go if so
0BBA FEC0     15730 $N1A    CP      0C0H            ;Tab or special?
0BBC 3806     15740         JR      C,DONORM        ;Go on normal characters
              15750 ;
              15760 ;               Character is => 0C0H
              15770 ;
0BBE DDCB005E 15780         BIT     TABS,(IX+DO_MASK) ;Tabs or spec chars
0BC2 2826     15790         JR      Z,DO_TABS       ;Go if video tabs
```

```
              15800 ;
              15810 ;       Character is not tab expansion - do it
              15820 ;
0BC4 CDB80C   15830 DONORM  CALL    DO_DSPCHAR      ;Display the char
0BC7 DDCB00A6 15840         RES     CTL,(IX+DO_MASK) ;Turn off CTL bit
0BCB C1       15850 DO_RET  POP     BC              ;Get orig char
0BCC F3       15860 DO_RET1 DI                      ;Disable intr
0BCD 3A970B   15870         LD      A,(CRSAVE)      ;If a cursor is on, then
0BD0 B7       15880         OR      A               ;  we need to save the
0BD1 2810     15890         JR      Z,$N1           ;  current char & display
0BD3 1A       15900         LD      A,(DE)          ;  the cursor character
0BD4 32970B   15910         LD      (CRSAVE),A      ;Save current char
0BD7 3A7F00   15920         LD      A,(VFLAG$)      ;Allow tasker to blink
0BDA CBBF     15930         RES     7,A
0BDC 327F00   15940         LD      (VFLAG$),A
0BDF 3A980B   15950         LD      A,(CRSCHAR)     ;P/u cusor character
0BE2 12       15960         LD      (DE),A          ;Put it on the screen
0BE3 ED53950B 15970 $N1     LD      (CURSOR),DE     ;Update cursor position
0BE7 BF       15980         CP      A               ;Clear status
0BE8 79       15990         LD      A,C             ;Restore the char
0BE9 C9       16000         RET
              16010 ;
              16020 ;       Perform a tab expansion {C0H-FFH}
              16030 ;
              16040 DO_TABS
0BEA D6C0     16050         SUB     0C0H            ;Compute spaces
0BEC 28DD     16060         JR      Z,DO_RET        ;Forget it if TAB(0)
0BEE 47       16070         LD      B,A             ;Display requested
0BEF 0E20     16080 $N2     LD      C,' '           ;  number of spaces
0BF1 CDB80C   16090         CALL    DO_DSPCHAR
0BF4 10F9     16100         DJNZ    $N2
0BF6 18D3     16110         JR      DO_RET
              16120 ;
              16130 ;       Routine to move the cursor to begin of line {29}
              16140 ;
              16150 CRSBOL
0BF8 EB       16160         EX      DE,HL           ;Cursor addr to HL
0BF9 CDF40D   16170         CALL    ADDR1           ;Find row,col
0BFC 6F       16180         LD      L,A             ;Set col to start
0BFD C3D00D   16190         JP      ROWCOL_2_ADDR   ;Calc address of BOL
              16200 ;
              16210 ;       Routines to turn on/off the cursor {14/15}
              16220 ;
0C00 1A       16230 CRSON   LD      A,(DE)          ;Get screen character
0C01 32970B   16240 CRSOFF  LD      (CRSAVE),A      ;Save zero or CRT char
0C04 C9       16250         RET
              16260 ;
              16270 ;       Routine moves cursor to start of video page {28}
              16280 ;        set to 80 column, and turns off inverse video
              16290 ;
              16300 CRSHOME
0C05 1100F8   16310         LD      DE,CRTBGN$      ;Home the cursor
0C08 3A7600   16320         LD      A,(MODOUT$)     ;P/u the mask &
0C0B E6FB     16330         AND     0FBH            ;  set to 80 cpl
0C0D CDB20C   16340         CALL    SETMOD
0C10 187A     16350         JR      DO_INVERT_DIS   ;Set to normal video
              16360 ;
              16370 ;       Routine to backspace & erase cursor {08}
              16380 ;
```

Video Driver

```
                16390 BACKSPA
0C12 CD1B0C     16400           CALL    CRSBKSP         ;Backspace the cursor
0C15 C8         16410           RET     Z               ;If not at start,
0C16 0E20       16420           LD      C,' '           ;  put a space at
0C18 C3CA0D     16430           JP      PUT_@           ;  at the new loc'n
                16440 ;
                16450 ;         Routine to backspace the cursor {24}
                16460 ;
                16470 CRSBKSP
0C1B 3A7600     16480           LD      A,(MODOUT$)     ;If double width chars,
0C1E E604       16490           AND     4               ;  need to do twice
0C20 C4230C     16500           CALL    NZ,$+3
0C23 2100F8     16510           LD      HL,CRTBGN$      ;See if at home position
0C26 ED52       16520           SBC     HL,DE           ;  prior to adjusting
0C28 C8         16530           RET     Z
0C29 1B         16540           DEC     DE              ;Decrement the cursor pos
0C2A C9         16550           RET
                16560 ;
                16570 ;         Routine to move the cursor up one line {27}
                16580 ;
                16590 CRSUP
0C2B 21B0FF     16600           LD      HL,NEGLINE      ;Move up one line
0C2E 1803       16610           JR      MOVCRS
                16620 ;
                16630 ;         Routine to move the cursor down one line {26}
                16640 ;
                16650 CRSDOWN
0C30 215000     16660           LD      HL,LINESIZ      ;Add the line length
0C33 19         16670 MOVCRS    ADD     HL,DE           ;  to the current pos
0C34 7C         16680           LD      A,H             ;Make sure we did not
0C35 FEF8       16690           CP      CRTBGN$<-8      ;  go over the top
0C37 D8         16700           RET     C
0C38 EB         16710           EX      DE,HL           ;  & switch back to DE
0C39 1B         16720           DEC     DE              ;Adjust for fall thru
0C3A C3C30C     16730           JP      CRSFRW0
                16740 ;
                16750 ;         Set to 40 cpl mode {23}
                16760 ;
0C3D 3A7600     16770 SET40     LD      A,(MODOUT$)     ;Get image of the port
0C40 F604       16780           OR      04H             ;Merge in 40 cpl bit
0C42 186E       16790           JR      SETMOD
                16800 ;
                16810 ;         Routines to parse control functions
                16820 ;
                16830 DO_CONTROL
0C44 21CB0B     16840           LD      HL,DO_RET       ;Establish RET
0C47 E5         16850           PUSH    HL
0C48 FE08       16860           CP      08H             ;Backspace?
0C4A 28C6       16870           JR      Z,BACKSPA
0C4C FE0A       16880           CP      0AH             ;Line feed?
0C4E 2802       16890           JR      Z,$+4           ;  is same as <ENTER>
0C50 D60D       16900           SUB     0DH             ;Carriage return?
0C52 CA020D     16910           JP      Z,LINFEED
0C55 3D         16920           DEC     A               ;Cursor on?
0C56 28A8       16930           JR      Z,CRSON
0C58 3D         16940           DEC     A               ;Cursor off?
0C59 28A6       16950           JR      Z,CRSOFF
0C5B 3D         16960           DEC     A               ;Reverse video?
0C5C 282B       16970           JR      Z,DO_INVERT_ENA
```

Video Driver

```
0C5E 3D         16980           DEC     A
0C5F 283A       16990           JR      Z,DO_INVERT_OFF
0C61 D604       17000           SUB     4                ;Swap tab/alternate?
0C63 2841       17010           JR      Z,TGGLTAB
0C65 3D         17020           DEC     A                ;Special/alternate?
0C66 2845       17030           JR      Z,TGGLALT
0C68 3D         17040           DEC     A                ;40 cpl?
0C69 28D2       17050           JR      Z,SET40
0C6B 3D         17060           DEC     A                ;Cursor backspace?
0C6C 28AD       17070           JR      Z,CRSBKSP
0C6E 3D         17080           DEC     A                ;Cursor forward?
0C6F 284A       17090           JR      Z,CRSFRWD
0C71 3D         17100           DEC     A                ;Cursor down?
0C72 28BC       17110           JR      Z,CRSDOWN
0C74 3D         17120           DEC     A                ;Cursor up?
0C75 28B4       17130           JR      Z,CRSUP
0C77 3D         17140           DEC     A                ;Cursor home?
0C78 CA050C     17150           JP      Z,CRSHOME
0C7B 3D         17160           DEC     A                ;Cursor BOL?
0C7C CAF80B     17170           JP      Z,CRSBOL
0C7F 3D         17180           DEC     A                ;Clear to EOL?
0C80 CA120D     17190           JP      Z,CLREOL
0C83 3D         17200           DEC     A
0C84 CA1E0D     17210           JP      Z,CLREOF         ;Clear to end-of-frame?
0C87 AF         17220           XOR     A                ;Clear A reg.
0C88 C9         17230           RET
                17240 ;
                17250 ;         Routine to enable inverse video
                17260 ;
                17270 DO_INVERT_ENA
0C89 0608       17280           LD      B,8              ;Set for enable
0C8B 21         17290           DB      21H              ;Ignore next load
                17300 DO_INVERT_DIS
0C8C 0600       17310           LD      B,0
0C8E 2A3508     17320           LD      HL,(OPREG_SV_PTR)       ;Real OPREG$
0C91 7E         17330           LD      A,(HL)           ;P/u OPREG mask
0C92 E6F7       17340           AND     0F7H             ;Strip bit 3
0C94 B0         17350           OR      B                ;Set/reset invideo bit
0C95 77         17360           LD      (HL),A           ;  and restuff
0C96 78         17370           LD      A,B              ;Get mode mask byte
0C97 07         17380           RLCA                     ;Rotate left 4 times to
0C98 07         17390           RLCA                     ;  make an 8 into 80H
0C99 07         17400           RLCA                     ;  for inverse on
0C9A 07         17410           RLCA                     ;Inverse off remains 0
                17420 DO_INVERT_OFF
0C9B 32CB0D     17430           LD      (INVIDEO),A      ;Set the mask byte
0C9E C9         17440           RET
                17450 ;
                17460 ;         Routine to toggle display of controls
                17470 ;
0C9F 21CB0B     17480 TGGLCTL   LD      HL,DO_RET        ;Establish ret addr
0CA2 E5         17490           PUSH    HL
0CA3 3E10       17500           LD      A,10H            ;Toggle bit 4
0CA5 21         17510           DB      21H              ;Ignore next
                17520 ;
                17530 ;         Toggle tabs & alternate character set
                17540 ;
                17550 TGGLTAB
0CA6 3E08       17560           LD      A,8              ;Toggle bit 3
```

Video Driver

```
0CA8 DDAE00     17570          XOR      (IX+DO_MASK)    ;P/u mask value
0CAB 1850       17580          JR       SETMASK
                17590 ;
                17600 ;        Toggle special & alternate character set
                17610 ;
                17620 TGGLALT
0CAD 3A7600     17630          LD       A,(MODOUT$)     ;P/u port mask
0CB0 EE08       17640          XOR      8               ;Flip the bit
0CB2 327600     17650 SETMOD   LD       (MODOUT$),A     ;Resave port mask
0CB5 D3EC       17660                                   ;  and send the byte
0CB7 C9         17670          RET
                17680 ;
                17690 ;        Display character <C> at current cursor position
                17700 ;
                17710 DO_DSPCHAR
0CB8 CDCA0D     17720          CALL     PUT_@           ;Display the char
                17730 ;
                17740 ;        Routine to perform cursor forward {25}
                17750 ;
                17760 CRSFRWD
0CBB 3A7600     17770          LD       A,(MODOUT$)     ;If double width chars,
0CBE E604       17780          AND      4               ;  need to do twice
0CC0 2801       17790          JR       Z,CRSFRW0
0CC2 13         17800          INC      DE              ;Move cursor forward
0CC3 13         17810 CRSFRW0  INC      DE
0CC4 217FFF     17820          LD       HL,CRTEND       ;Off the screen?
0CC7 ED52       17830          SBC      HL,DE
0CC9 D0         17840          RET      NC              ;Back if not
0CCA CD2B0C     17850          CALL     CRSUP           ;Put cursor back on
0CCD D5         17860          PUSH     DE              ;Save cursor position
                17870 DO_SCROLL
0CCE DD7E00     17880          LD       A,(IX+DO_MASK)  ;Get scroll protect
0CD1 E607       17890          AND      SCRPROT
0CD3 2100F8     17900          LD       HL,CRTBGN$      ;Point to CRT start
0CD6 118007     17910          LD       DE,CRTSIZE      ;P/u CRT size
0CD9 C5         17920          PUSH     BC
0CDA 015000     17930          LD       BC,LINESIZ      ;Set line size
0CDD 3C         17940          INC      A               ;Adjust scroll protect
0CDE 09         17950 $N4      ADD      HL,BC           ;Move logical start
0CDF EB         17960          EX       DE,HL           ;  down one line
0CE0 B7         17970          OR       A               ;  and subtract one line
0CE1 ED42       17980          SBC      HL,BC           ;  from the CRTSIZE for
0CE3 EB         17990          EX       DE,HL           ;  each protected line
0CE4 3D         18000          DEC      A               ;Dec scroll protect
0CE5 20F7       18010          JR       NZ,$N4          ;Loop until done
0CE7 D5         18020          PUSH     DE              ;Save the move length
0CE8 E5         18030          PUSH     HL              ;Save the move-from
0CE9 ED42       18040          SBC      HL,BC           ;Move start back one
0CEB EB         18050          EX       DE,HL           ;  line, Source =
0CEC E1         18060          POP      HL              ;  start + one
0CED C1         18070          POP      BC              ;Get back dest locn
0CEE EDB0       18080          LDIR                     ;Scroll unprotected
0CF0 C1         18090          POP      BC              ;Recover line size
0CF1 182C       18100          JR       CLREOF1         ;Clear to EOF from DE
                18110 ;
                18120 ;        Set scroll protect value
                18130 ;                 C = scroll protect <0-7>
                18140 ;                 B = 7
                18150 ;                 SVC = 15, @VDCTL
```

```
                 18160 ;
                 18170 SET_SCROLL
0CF3 79          18180         LD      A,C             ;Get user value
0CF4 E607        18190         AND     7               ;Make modulo 8
0CF6 4F          18200         LD      C,A
0CF7 3A940B      18210         LD      A,(DODATA$)     ;P/u current mask
0CFA E6F8        18220         AND     0F8H            ;Remove current scroll
0CFC B1          18230         OR      C               ;Merge in the new value
0CFD 32940B      18240 SETMASK LD      (DODATA$),A     ;  & reload mask
0D00 AF          18250         XOR     A               ;Z-flag return
0D01 C9          18260         RET
                 18270 ;
                 18280 ;       Routine to move down one line {10/13}
                 18290 ;
0D02 CDF80B      18300 LINFEED CALL    CRSBOL          ;Move to BOL
0D05 D5          18310         PUSH    DE              ;Save cursor position
0D06 CD300C      18320         CALL    CRSDOWN         ;Move down one line
0D09 B7          18330         OR      A               ;Reset the carry flag
0D0A 2180FF      18340         LD      HL,CRTEND+1<    ;  & check if off of
0D0D ED52        18350         SBC     HL,DE           ;  the screen
0D0F 28BD        18360         JR      Z,DO_SCROLL     ;Scroll if so
0D11 E1          18370         POP     HL              ;Discard old position
0D12 D5          18380 CLREOL  PUSH    DE              ;Save new cursor pos
0D13 CDF80B      18390         CALL    CRSBOL          ;Get start of line
0D16 214F00      18400         LD      HL,79           ;Calculate end of line
0D19 19          18410         ADD     HL,DE           ;HL = end of line
0D1A D1          18420         POP     DE              ;DE = current position
0D1B D5          18430         PUSH    DE
0D1C 1804        18440         JR      CLREOF2         ;Clear the line
                 18450 ;
                 18460 ;       Clear to the end of the frame
                 18470 ;
0D1E D5          18480 CLREOF  PUSH    DE              ;Save current cursor pos
0D1F 217FFF      18490 CLREOF1 LD      HL,CRTEND       ;Point to last RAM byte
0D22 3ACB0D      18500 CLREOF2 LD      A,(INVIDEO)     ;P/u normal/reverse
0D25 CBEF        18510         SET     5,A             ;  & make it a space
0D27 12          18520         LD      (DE),A          ;Stuff the "space"
0D28 B7          18530         OR      A               ;Reset carry for subtract
0D29 ED52        18540         SBC     HL,DE           ;Calculate length
0D2B 2809        18550         JR      Z,CLREOF3       ;Back if at end already
0D2D C5          18560         PUSH    BC
0D2E 44          18570         LD      B,H             ;Xfer length to BC
0D2F 4D          18580         LD      C,L
0D30 62          18590         LD      H,D             ;Xfer start to HL
0D31 6B          18600         LD      L,E
0D32 13          18610         INC     DE              ;Bump up by one
0D33 EDB0        18620         LDIR                    ;Propagate the space
0D35 C1          18630         POP     BC
0D36 D1          18640 CLREOF3 POP     DE
0D37 C9          18650         RET
                 18660 ;
                 18670 ;       Routine to stuff the video cursor RAM address
                 18680 ;
0D38 CDD00D      18690 @VDCTL3 CALL    ROWCOL_2_ADDR   ;Calculate video address
0D3B C0          18700         RET     NZ              ;Back on error
0D3C F3          18710         DI                      ;Disable any video tasks
0D3D ED53950B    18720         LD      (CURSOR),DE     ;  until cursor is updated
0D41 C9          18730         RET
                 18740 ;
                 18750 ;       Video control SVC processor
                 18760 ;
                 18770 @_VDCTL
0D42 CD1708      18780         CALL    ENADIS_DO_RAM   ;Bring up the video RAM
                 18790 ;
                 18800 ;       Test if in Task processor
                 18810 ;
0D45 3A7700      18820         LD      A,(NFLAG$)      ;P/u NFLAG$
0D48 CB77        18830         BIT     6,A             ;Test for task process
0D4A 2015        18840         JR      NZ,VDCTL        ;If so skip setup
                 18850 ;
                 18860 ;       HANDLES @VDCTL screen set up for normal use
                 18870 ;
0D4C D5          18880         PUSH    DE
0D4D CDB30D      18890         CALL    $N0             ;Normalize character at cursor
0D50 D1          18900         POP     DE              ;Recover value
0D51 D5          18910         PUSH    DE
0D52 CD610D      18920         CALL    VDCTL           ;Do function request
0D55 F5          18930         PUSH    AF              ;Save the error status
0D56 F3          18940         DI                      ;Stop video tasks tempy
0D57 ED5B950B    18950         LD      DE,(CURSOR)
0D5B CDCC0B      18960         CALL    DO_RET1         ;Normalize screen and cursor
0D5E F1          18970         POP     AF
0D5F D1          18980         POP     DE
0D60 C9          18990         RET
                 19000 ;
0D61 3E09        19010 VDCTL   LD      A,9             ;Check for VIDLINE,
0D63 B8          19020         CP      B               ;  function 9
0D64 2825        19030         JR      Z,VIDLIN
0D66 3E2B        19040         LD      A,43            ;Prepare for user ERROR
0D68 05          19050         DEC     B
0D69 2843        19060         JR      Z,GET_@_ROWCOL  ;<C> from row-H, col-L
0D6B 05          19070         DEC     B
0D6C 2858        19080         JR      Z,PUT_@_ROWCOL  ;<C> to row-H, col-L
0D6E 05          19090         DEC     B
0D6F 28C7        19100         JR      Z,@VDCTL3       ;Set cursor to H,L
0D71 05          19110         DEC     B
0D72 287D        19120         JR      Z,ADDR_2_ROWCOL ;Cursor row,col to H,L
0D74 1100F8      19130         LD      DE,CRTBGN$      ;Init to start of video
0D77 05          19140         DEC     B
0D78 282D        19150         JR      Z,VIDMOV1       ;User RAM to video
0D7A 05          19160         DEC     B
0D7B 2822        19170         JR      Z,VIDMOVE       ;Video RAM to user
0D7D 05          19180         DEC     B
0D7E CAF30C      19190         JP      Z,SET_SCROLL    ;Set scroll protect
0D81 05          19200         DEC     B
0D82 C0          19210         RET     NZ              ;Return if bad request
                 19220 ;
                 19230 ;       Establish cursor character
                 19240 ;
0D83 E5          19250         PUSH    HL
0D84 21980B      19260         LD      HL,CRSCHAR      ;Point to cursor char storage
0D87 7E          19270         LD      A,(HL)          ;P/u current cursor character
0D88 71          19280         LD      (HL),C          ;  & update with new one
0D89 E1          19290         POP     HL
0D8A C9          19300         RET
                 19310 ;
                 19320 ;       VIDLIN routine function - 9 in register B
                 19330 ;
```

Video Driver

```
0D8B 2E00      19340 VIDLIN  LD      L,0             ;Always starts at col 0
0D8D D5        19350         PUSH    DE              ;Save user buffer
0D8E CDD00D    19360         CALL    ROWCOL_2_ADDR   ;Get address to DE
0D91 E1        19370         POP     HL              ;Recover user buffer
0D92 C0        19380         RET     NZ              ;Quit on bad address
0D93 0C        19390         INC     C               ;Check direction
0D94 0D        19400         DEC     C               ;If Z then to screen
0D95 2801      19410         JR      Z,MOVLIN        ;Set to go
0D97 EB        19420         EX      DE,HL           ;Reverse direction
0D98 015000    19430 MOVLIN  LD      BC,LINESIZ      ;Set line size
0D9B EDB0      19440         LDIR                    ;Move it
0D9D AF        19450         XOR     A               ;Z on RET
0D9E C9        19460         RET
               19470 ;
               19480 ;       Routine to move video RAM
               19490 ;
0D9F 7C        19500 VIDMOVE LD      A,H             ;Check on user buffer
0DA0 C608      19510         ADD     A,8             ;  not above X'F800' &
0DA2 FE2C      19520         CP      24H+8           ;  not below X'2400'
0DA4 3847      19530         JR      C,PERR
0DA6 EB        19540         EX      DE,HL           ;Xchng user buffer,screen
0DA7 018007    19550 VIDMOV1 LD      BC,CRTSIZE      ;Set for full screen xfer
0DAA EDB0      19560         LDIR
0DAC BF        19570         CP      A               ;Set Z flag
0DAD C9        19580         RET
               19590 ;
               19600 ;       Routine to get the character at row,col
               19610 ;
               19620 GET_@_ROWCOL
0DAE CDD00D    19630         CALL    ROWCOL_2_ADDR   ;Get Address of req
0DB1 1A        19640         LD      A,(DE)          ;P/u the character
0DB2 C9        19650         RET                     ;Back on error or no error
               19660 ;
               19670 ;       Routine to halt blinking cursor & restore char
               19680 ;
0DB3 E5        19690 $N0     PUSH    HL
0DB4 217F00    19700         LD      HL,VFLAG$
0DB7 CBFE      19710         SET     7,(HL)          ;Disable blinking cursor
0DB9 E1        19720         POP     HL
0DBA ED5B950B  19730         LD      DE,(CURSOR)     ;Get cursor pos in DE
0DBE 3A970B    19740         LD      A,(CRSAVE)      ;P/u saved character
0DC1 B7        19750         OR      A               ;If one is saved, put
               19760                                 ;  it on screen, else
0DC2 2009      19770         JR      NZ,PUTA@DE      ;  ignore it
0DC4 1A        19780         LD      A,(DE)          ;Cursor not ON but get
0DC5 C9        19790         RET                     ;  character anyway
               19800 ;
               19810 ;       Routine to put a character at row,col
               19820 ;
               19830 PUT_@_ROWCOL
0DC6 CDD00D    19840         CALL    ROWCOL_2_ADDR   ;Get address of req
0DC9 C0        19850         RET     NZ              ;Back on error
0DCA 3E00      19860 PUT_@   LD      A,0             ;Merge in reverse video
0DCB           19870 INVIDEO EQU     $-1
0DCC B1        19880         OR      C
0DCD 12        19890 PUTA@DE LD      (DE),A          ;Put the character
0DCE BF        19900         CP      A               ;Set Z-flag for return
0DCF C9        19910         RET
               19920 ;
```

```
                 19930 ;       Routine to calculate cursor position from row,col
                 19940 ;
                 19950 ROWCOL_2_ADDR
0DD0 3E4F        19960         LD      A,79
0DD2 BD          19970         CP      L
0DD3 3818        19980         JR      C,PERR          ;Error if > 79
0DD5 7C          19990         LD      A,H             ;P/u row number
0DD6 FE18        20000         CP      24
0DD8 3013        20010         JR      NC,PERR         ;Error if > 23
0DDA E5          20020         PUSH    HL
0DDB C5          20030         PUSH    BC
0DDC 4D          20040         LD      C,L             ;Save column
0DDD 06F8        20050         LD      B,CRTBGN$<-8    ;Set to start of DO RAM
0DDF 215000      20060         LD      HL,LINESIZ
0DE2 CDC906      20070         CALL    @MUL16          ;Rows * line size
0DE5 65          20080         LD      H,L             ;Shift to HL
0DE6 6F          20090         LD      L,A
0DE7 09          20100         ADD     HL,BC           ;Add in col & RAM start
0DE8 EB          20110         EX      DE,HL           ;Address to DE
0DE9 C1          20120         POP     BC
0DEA E1          20130         POP     HL
0DEB AF          20140         XOR     A               ;Set Z flag
0DEC C9          20150         RET
0DED 3E2B        20160 PERR    LD      A,43            ;SVC parameter error
0DEF B7          20170         OR      A               ;Set NZ condition
0DF0 C9          20180         RET
                 20190 ;
                 20200 ;       Routine to get row,col of video cursor
                 20210 ;
                 20220 ADDR_2_ROWCOL
0DF1 2A950B      20230         LD      HL,(CURSOR)     ;Get addr in HL
0DF4 7C          20240 ADDR1   LD      A,H             ;Make address relative
0DF5 E607        20250         AND     7               ;  to origin 0
0DF7 67          20260         LD      H,A
0DF8 3E50        20270         LD      A,LINESIZ       ;Set divisor
0DFA CDE306      20280         CALL    @DIV16
0DFD 65          20290         LD      H,L             ;Row to register H
0DFE 6F          20300         LD      L,A             ;Column to register L
0DFF AF          20310         XOR     A               ;Set zero return code
0E00 C9          20320         RET
0E00             20330 DOEND   EQU     $-1
0E01             01700 *GET    PRDVR:3                 ;Printer driver & filter
                 20350 ;PRDVR/ASM  -  LS-DOS 6.2
0E01             20360         SUBTTL  '<Printer Driver>'
```

Printer Driver

```
               20380 *MOD
00F8           20390 PRPORT
               20400 ;
               20410 ;        PR driver entry point
               20420 ;        It passes X'00'-X'FF'
               20430 ;        Unless INTL version
               20440 ;
0E01 180A      20450 PRDVR    JR      PRBGN           ;Branch around linkage
0E03 3C0E      20460          DW      PREND           ;Last byte used
0E05 03        20470          DB      3,'$PR'
     24 50 52
0E09 1802      20480          DW      PRDCB$          ;Pointer to its DCB
0E0B 0000      20490          DW      0               ;Reserved
               20500 ;
               20510 ;        Driver code
               20520 ;
0E0D 280A      20530 PRBGN    JR      Z,$02           ;Go if output
0E0F 3804      20540          JR      C,$01           ;Go if input req
               20550 ;
               20560 ;        Character CTL request
               20570 ;
0E11 79        20580          LD      A,C             ;If CTL 0, return
0E12 B7        20590          OR      A               ;  status else
0E13 2821      20600          JR      Z,$04           ;  treat as a Get
               20610 ;
               20620 ;        Character GET request
               20630 ;
0E15 F6FF      20640 $01      OR      0FFH            ;Set nz
0E17 2F        20650          CPL                     ;  & A=0       to show
0E18 C9        20660          RET                     ;  no char available
               20670 ;
               20680 ;        Character PUT request
               20690 ;
0E19 11D007    20700 $02      LD      DE,2000         ;Check status 2000 times
0E1C CD360E    20710 $02A     CALL    $04             ;PR ready?
0E1F 2811      20720          JR      Z,$03           ;Go if so
               20730 ;
               20740 ;        Ten second timout delay loop
               20750 ;
0E21 C5        20760          PUSH    BC              ;Printer was not ready
0E22 015401    20770          LD      BC,340
0E25 CD8203    20780          CALL    PAUSE@          ;Delay a bit
0E28 C1        20790          POP     BC
0E29 1B        20800          DEC     DE              ;Time up?
0E2A 7A        20810          LD      A,D
0E2B B3        20820          OR      E
0E2C 20EE      20830          JR      NZ,$02A         ;Nope, continue check
0E2E 3E08      20840          LD      A,8             ;Device not avail...
0E30 B7        20850          OR      A               ;Set NZ condition
0E31 C9        20860          RET
0E32           20870 $03      EQU     $
               20880 ;
               20890          IF      @INTL
               20900          LD      A,(IFLAG$)
               20910          BIT     6,A             ;Special DMP PR?
               20920          ENDIF
               20930 ;
0E32 79        20940          LD      A,C
               20950 ;
               20960          IF      @INTL
```

Printer Driver

```
               20970        JR      Z,PVAL3
               20980        CP      0C0H           ;Values C0-FF (-20H)
               20990        JR      C,PVAL2        ;Go if less
               21000        SUB     20H            ;Shift to European chars
               21010        JR      PVAL3
               21020 PVAL2  CP      0A0H           ;A0-BF (+40H)
               21030        JR      C,PVAL3        ;Go if less
               21040        ADD     A,40H          ;Shift to graphics
               21050        ENDIF
               21060 ;
0E33 D3F8      21070 PVAL3                         ;Put out char
               21080 ;
               21090        IF      @INTL
               21100        LD      A,C            ;Restore original
               21110        CP      A              ;Set Z
               21120        ENDIF
               21130 ;
0E35 C9        21140        RET
               21150 ;
0E36 DBF8      21160 $04    IN      A,(PRPORT)     ;Scan PR status
0E38 E6F0      21170        AND     0F0H           ;Mask unused positions
0E3A FE30      21180        CP      30H            ;PR ready?
0E3C C9        21190        RET                    ;Return with answer
0E3C           21200 PREND  EQU     $-1
0E3D           01710 *GET   FDCDVR:3               ;Floppy disk driver
               21210 ;FDCDVR/ASM - LS-DOS 6.2
0E3D           21220        SUBTTL  '<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 ;
00E4            21310 WRNMIPORT                         ;NMI mask register
00F0            21320 FDCADR                            ;FDC command
00F0            21330 FDCSTAT                           ;FDC status
00F1            21340 TRKREG                            ;FDC track register
00F2            21350 SECREG                            ;FDC sector register
00F3            21360 DATREG                            ;FDC data register
00F4            21370 DSELCT                            ;Drive select port
                21380 ;
                21390 ;
                21400 ;       Disk Driver Entry Point
                21410 ;
0E3D 184F       21420 FDCDVR  JR      FDCBGN            ;Branch to entry code
0E3F F30F       21430         DW      FDCEND            ;Last byte used
0E41 03         21440         DB      3,'$FD'           ;Module name
     24 46 44
                21450 ;
                21460 ;       Automatic density recognition and retry density switch
                21470 ;
                21480 SWDEN
0E45 3E03       21490         LD      A,3               ;Check counter for 2
0E47 B8         21500         CP      B                 ;  tries left after this one
0E48 285E       21510         JR      Z,RESTOR          ;If so try a RESTORE
                21520 ;
0E4A FD7E03     21530         LD      A,(IY+3)          ;Flip the density bit,
0E4D EE40       21540         XOR     40H               ;  Bit 6, (IY+3)
0E4F FD7703     21550         LD      (IY+3),A
0E52 010924     21560         LD      BC,2409H          ;Set alloc to SDEN
0E55 CB77       21570         BIT     6,A               ;Test SDEN/DDEN
0E57 2803       21580         JR      Z,SDEN            ;Do SDEN if it was DDEN
0E59 011145     21590         LD      BC,4511H          ;  else set alloc to DDEN
0E5C FD7107     21600 SDEN    LD      (IY+7),C
0E5F FD7008     21610         LD      (IY+8),B
0E62 C9         21620         RET
                21630 ;
                21640 ;       Verify routine
                21650 ;
0E63 21E80F     21660 VERFIN  LD      HL,BUCKET         ;Set byte bucket
0E66 3E2D       21670         LD      A,2DH             ;Set for DEC L,...
0E68 1E         21680         DB      1EH               ;Ignore next with LD E,n
                21690 ;
                21700 ;       Read routine
                21710 ;
0E69 AF         21720 RDIN    XOR     A                 ;Set for NOP
0E6A 327D0E     21730         LD      (CKVER),A
0E6D CD2B0F     21740         CALL    RWINIT            ;Initialize
0E70 1E16       21750         LD      E,16H             ;Status mask
0E72 DBF0       21760 RDIN1   IN      A,(FDCSTAT)       ;Get status
0E74 A3         21770         AND     E                 ;Loop until DRQ
0E75 28FB       21780         JR      Z,RDIN1           ;  or error
0E77 EDA2       21790         INI                       ;Grab byte
0E79 F3         21800         DI
0E7A 7A         21810         LD      A,D               ;Get drive sel + WSGEN
0E7B D3F4       21820 RDIN2   OUT     (DSELCT),A        ;Initiate wait state
```

Floppy Disk Driver

```
0E7D 00         21830 CKVER   NOP                    ;DEC L: if verify
0E7E EDA2       21840         INI                    ;Xfer byte
0E80 20F9       21850         JR      NZ,RDIN2       ;Loop then TSTBSY
                21860 ;
                21870 ;       Reselect drive while controller is busy
                21880 ;
0E82 DBF0       21890 TSTBSY  IN      A,(FDCSTAT)    ;Ck FDC status
0E84 CB47       21900         BIT     0,A            ;Busy?
0E86 C8         21910         RET     Z              ;RET if not
0E87 3A1B00     21920         LD      A,(PDRV$)      ;P/u drive
0E8A D3F4       21930         OUT     (DSELCT),A     ;  & reselect
0E8C 18F4       21940         JR      TSTBSY         ;Loop until idle
                21950 ;
                21960 ;       Driver start
                21970 ;
0E8E 78         21980 FDCBGN  LD      A,B            ;P/u primitive request
0E8F A7         21990         AND     A              ;NOP?
0E90 C8         22000         RET     Z              ;Quit if so
0E91 FE07       22010         CP      7
0E93 28ED       22020         JR      Z,TSTBSY       ;Jump on TSTBSY request
0E95 D2440F     22030         JP      NC,IORQST      ;Jump on I/O request
0E98 FE06       22040         CP      6
0E9A 284C       22050         JR      Z,SEEKTRK      ;Jump on track seek
0E9C 3D         22060         DEC     A
0E9D 2811       22070         JR      Z,SELECT       ;Jump on drive select
0E9F FD3405     22080         INC     (IY+5)         ;Bump current cylinder
0EA2 FE04       22090         CP      4
0EA4 0658       22100         LD      B,58H          ;FDC step-in command
0EA6 2872       22110         JR      Z,STEPIN
0EA8 FD360500   22120 RESTOR  LD      (IY+5),0       ;Set track to 0
0EAC 0608       22130         LD      B,8            ;Restore drive
0EAE 186A       22140         JR      STEPIN
                22150 ;
0EB0 CD820E     22160 SELECT  CALL    TSTBSY         ;Check drive status
0EB3 07         22170         RLCA
0EB4 F5         22180         PUSH    AF             ;Save NOT READY flag
0EB5 C5         22190         PUSH    BC
0EB6 FD7E03     22200         LD      A,(IY+3)       ;P/u SDEN/DDEN
0EB9 17         22210         RLA                    ;Bit 6=>7, bit 4=>4
0EBA CB2F       22220         SRA     A
0EBC E690       22230         AND     90H            ;Keep only DDEN & side 1
0EBE 4F         22240         LD      C,A            ;Save the bits
0EBF CB7F       22250         BIT     7,A            ;Check if SDEN or DDEN
0EC1 2808       22260         JR      Z,NOPCMP       ;No precomp if SDEN
0EC3 FD7E09     22270         LD      A,(IY+9)       ;Set precomp on all
0EC6 BA         22280         CP      D              ;  tracks above DIR
0EC7 3002       22290         JR      NC,NOPCMP      ;Go if no precomp needed
0EC9 CBE9       22300         SET     5,C            ;Request precomp
0ECB FD7E04     22310 NOPCMP  LD      A,(IY+4)       ;Get drive sel code
0ECE E60F       22320         AND     0FH            ;Keep only sel bits
0ED0 B1         22330         OR      C              ;Merge in bits 4,5,7
0ED1 C1         22340         POP     BC
0ED2 D3F4       22350         OUT     (DSELCT),A     ;Select drive
0ED4 321B00     22360         LD      (PDRV$),A      ;Store port byte
0ED7 F1         22370         POP     AF             ;Retrieve Not Ready bit
0ED8 D0         22380         RET     NC             ;Ret if was ready
0ED9 FDCB0356   22390         BIT     2,(IY+3)       ;Check DELAY=0.5 or 1.0
0EDD CCE00E     22400         CALL    Z,FDCDLY       ;Double delay if 1.0
0EE0 C5         22410 FDCDLY  PUSH    BC             ;Delay routine
```

Floppy Disk Driver

```
0EE1 067F     22420          LD     B,7FH
0EE3 CD8203   22430          CALL   PAUSE@
0EE6 C1       22440          POP    BC
0EE7 C9       22450          RET
              22460 ;
              22470 ;        Routine to seek a track
              22480 ;
0EE8 CD820E   22490 SEEKTRK  CALL   TSTBSY              ;Wait till not busy
0EEB FD7E05   22500          LD     A,(IY+5)            ;P/u current cylinder
0EEE D3F1     22510          OUT    (TRKREG),A          ;  & set FDC to current
0EF0 FD7E07   22520          LD     A,(IY+7)            ;P/u alloc data
0EF3 E61F     22530          AND    1FH                 ;Get highest # sector
0EF5 93       22540          SUB    E                   ;Form req sector minus
0EF6 2F       22550          CPL                        ;  max, setting CY flag if
0EF7 FDCB03A6 22560          RES    4,(IY+3)            ;  init side select to 0
0EFB 300B     22570          JR     NC,SETSECT          ;Go if sector on side 0
0EFD FDCB046E 22580          BIT    5,(IY+4)            ;If not 2-sided media,
0F01 2806     22590          JR     Z,FRCSID0           ;  don't set side 1
0F03 FDCB03E6 22600          SET    4,(IY+3)            ;Set side 1
0F07 1E       22610          DB     1EH                 ;Ignore next with LD E,n
0F08 7B       22620 SETSECT  LD     A,E                 ;Restore unaltered sec #
0F09 D3F2     22630 FRCSID0  OUT    (SECREG),A          ;Set sector
0F0B 7A       22640          LD     A,D
0F0C D3F3     22650          OUT    (DATREG),A          ;Set desired track
0F0E FDBE05   22660          CP     (IY+5)              ;If at desired track,
0F11 0618     22670          LD     B,18H               ;  use seek, else use
0F13 2805     22680          JR     Z,STEPIN            ;  seek w/verify
0F15 FD7205   22690          LD     (IY+5),D            ;Update current cylinder
0F18 061C     22700          LD     B,1CH               ;Seek w/verify ocmmand
0F1A CDB00E   22710 STEPIN   CALL   SELECT              ;Select drive
0F1D FD7E03   22720          LD     A,(IY+3)
0F20 E603     22730          AND    3                   ;Strip all but step rate
0F22 B0       22740          OR     B
0F23 D3F0     22750 PASSCMD  OUT    (FDCADR),A          ;Give FDC its command
0F25 0612     22760          LD     B,12H
0F27 10FE     22770          DJNZ   $                   ;Wait
0F29 AF       22780          XOR    A
0F2A C9       22790 FDCRET   RET
              22800 ;
              22810 ;        Read and write init routines
              22820 ;
0F2B 7A       22830 RWINIT   LD     A,D                 ;Restuff trk reg
0F2C D3F1     22840          OUT    (TRKREG),A
0F2E 3A1B00   22850          LD     A,(PDRV$)           ;Get select code
0F31 F640     22860          OR     40H                 ;Set WSGEN bit
0F33 57       22870          LD     D,A                 ;Save code in D
0F34 E610     22880          AND    10H                 ;Get side sel bit
0F36 0F       22890          RRCA                       ;  to bit 3
0F37 CB49     22900          BIT    1,C                 ;Check if doing side cmp
0F39 2001     22910          JR     NZ,GETCMD           ;Go if so
0F3B AF       22920          XOR    A
0F3C B1       22930 GETCMD   OR     C
0F3D 0EF3     22940          LD     C,DATREG            ;Get port into C
0F3F CD0E00   22950          CALL   FDDINT$             ;Interrupts on or off?
0F42 18DF     22960          JR     PASSCMD             ;Pass command to ctrlr
              22970 ;
              22980 ;        I/O request handler
              22990 ;
0F44 CB50     23000 IORQST   BIT    2,B                 ;Write command?
```

Floppy Disk Driver

```
0F46 ED4B7A00 23010         LD      BC,(RFLAG$-1)    ;P/u retry count
0F4A 0E82     23020         LD      C,82H            ;FDC cmd=readsec
0F4C 2010     23030         JR      NZ,WRCMD         ;Go if write command
0F4E FE0A     23040         CP      10               ;Verify sector?
0F50 2806     23050         JR      Z,VERFY
0F52 CD790F   23060         CALL    GRABNDO          ;Grab next code & insert
0F55 01       23070         DB      1                ;Error code start
0F56 690E     23080         DW      RDIN             ;Read entry point
0F58 CD790F   23090 VERFY   CALL    GRABNDO          ;Stuff I/O direction
0F5B 01       23100         DB      1                ;Error code start
0F5C 630E     23110         DW      VERFIN           ;Verify entry point
0F5E FDCB037E 23120 WRCMD   BIT     7,(IY+3)         ;Software WP?
0F62 2803     23130         JR      Z,WRCMD1         ;Bypass if not
0F64 3E0F     23140         LD      A,15             ;Else set WP error
0F66 C9       23150         RET
0F67 0EA2     23160 WRCMD1  LD      C,0A2H           ;Write sector FDC command
0F69 FE0E     23170         CP      14               ;Directory sector?
0F6B 3806     23180         JR      C,DOWRIT
0F6D 0EA3     23190         LD      C,0A3H           ;Change DAM if directory
0F6F 2802     23200         JR      Z,DOWRIT
0F71 0EF0     23210         LD      C,0F0H           ;  else write track
0F73 CD790F   23220 DOWRIT  CALL    GRABNDO          ;Switch code
0F76 09       23230         DB      9                ;Error code start
0F77 BC0F     23240         DW      WROUT            ;Write entry point
              23250 ;
              23260 ;       Routine stuffs error start byte & I/O vector
              23270 ;
0F79 E3       23280 GRABNDO EX      (SP),HL          ;Save HL & get ret addr
0F7A 7E       23290         LD      A,(HL)           ;P/u & stuff error code
0F7B 23       23300         INC     HL               ;  start byte
0F7C 32B50F   23310         LD      (ERRSTRT+1),A
0F7F 7E       23320         LD      A,(HL)           ;Set up data transfer
0F80 23       23330         INC     HL               ;  direction vector
0F81 66       23340         LD      H,(HL)
0F82 6F       23350         LD      L,A
0F83 22930F   23360         LD      (CALLIO),HL      ;Stuff CALL vector
0F86 E1       23370         POP     HL               ;Restore buffer addr
              23380 ;
              23390 ;       Main I/O handler routine
              23400 ;
0F87 C5       23410 RETRY   PUSH    BC               ;Save retry & FDC command
0F88 D5       23420         PUSH    DE               ;Save track/sector
0F89 E5       23430         PUSH    HL               ;Save buffer
0F8A CB61     23440         BIT     4,C              ;Test for track command
0F8C CCE80E   23450         CALL    Z,SEEKTRK        ;Seek if not track write
0F8F CD820E   23460         CALL    TSTBSY           ;Wait till not busy
0F92 CD0000   23470         CALL    0                ;Call I/O routine
0F93          23480 CALLIO  EQU     $-2              ;Data xfer direction
0F95 00       23490 DISKEI  NOP                      ;Will be changed to a EI after
              23500                                  ;  BOOT has read in SYS0
0F96 DBF0     23510         IN      A,(FDCSTAT)      ;Get status
0F98 E67C     23520         AND     7CH              ;Strip all but 2-6
0F9A E1       23530         POP     HL
0F9B D1       23540         POP     DE               ;Rcvr track & sector
0F9C C1       23550         POP     BC               ;Rcvr retry count & cmd
0F9D C8       23560         RET     Z                ;Ret if no error
0F9E CB57     23570         BIT     2,A              ;Lost data?
0FA0 20E5     23580         JR      NZ,RETRY         ;Don't count this retry
0FA2 F5       23590         PUSH    AF
```

Floppy Disk Driver

```
0FA3 E618       23600          AND     18H             ;Record not found or CRC
0FA5 280B       23610          JR      Z,DISKDUN       ;No retries if otherwise
0FA7 CB67       23620          BIT     4,A             ;Record not found?
0FA9 C5         23630          PUSH    BC              ;If so, switch
0FAA C4450E     23640          CALL    NZ,SWDEN        ;  density or restore
0FAD C1         23650          POP     BC
0FAE F1         23660          POP     AF
0FAF 10D6       23670          DJNZ    RETRY           ;Count down retry
0FB1 06         23680          DB      6               ;Ignore next with "LD B,nn"
0FB2 F1         23690 DISKDUN  POP     AF              ;Adjust ret code
0FB3 47         23700          LD      B,A
0FB4 3E00       23710 ERRSTRT  LD      A,0             ;Start with R=1, W=9
0FB6 CB08       23720 ERRTRAN  RRC     B
0FB8 D8         23730          RET     C
0FB9 3C         23740          INC     A
0FBA 18FA       23750          JR      ERRTRAN
                23760 ;
                23770 ;        Write routine
                23780 ;
0FBC CD2B0F     23790 WROUT    CALL    RWINIT          ;Set up initialization
0FBF 1E76       23800          LD      E,76H           ;Status mask
0FC1 DBF0       23810 WR01     IN      A,(FDCSTAT)     ;P/u status
0FC3 A3         23820          AND     E               ;Fall out on DRQ or error
0FC4 28FB       23830          JR      Z,WR01          ;  else loop
0FC6 EDA3       23840          OUTI                    ;Xfer byte to FDC
0FC8 F3         23850          DI                      ;Now kill the interrupts
0FC9 DBF0       23860          IN      A,(FDCSTAT)     ;Check for errors
0FCB 1F         23870          RRA                     ;Did BUSY drop?
0FCC D0         23880          RET     NC              ;Quit now if so
0FCD 3EC0       23890          LD      A,0C0H          ;Enable INTRQ and timeout
0FCF D3E4       23900          OUT     (WRNMIPORT),A
0FD1 0650       23910          LD      B,50H           ;Time delay for WRSEC
0FD3 10FE       23920          DJNZ    $
0FD5 46         23930          LD      B,(HL)          ;Get next byte early
0FD6 23         23940          INC     HL
0FD7 7A         23950 WR03     LD      A,D             ;Enable wait states
0FD8 D3F4       23960          OUT     (DSELCT),A
0FDA DBF0       23970          IN      A,(FDCSTAT)     ;Check if timed out
0FDC A3         23980          AND     E               ;Loop back if it timed
0FDD 28F8       23990          JR      Z,WR03          ;  out (must be WRTRK)
0FDF ED41       24000          OUT     (C),B           ;Pass 2nd byte
0FE1 7A         24010          LD      A,D             ;Get sel code + WSGEN bit
0FE2 D3F4       24020 WR02     OUT     (DSELCT),A      ;Pass until FDC times out
0FE4 EDA3       24030          OUTI                    ;  & generates NMI
0FE6 18FA       24040          JR      WR02
                24050          IFEQ    $&0FFH,0FFH
                24060          ERR     'Warning... BUCKET position error
                24070          ENDIF
0FE8 53         24080 BUCKET   DB      'S'
                24090 ;
0FE9 AF         24100 @RSTNMI  XOR     A               ;NMI vectors here
0FEA D3E4       24110          OUT     (WRNMIPORT),A   ;Disable INTRQ & timeout
0FEC 016400     24120          LD      BC,100          ;Need to wait a moment
0FEF CD8203     24130          CALL    PAUSE@          ;Call pause
0FF2 E1         24140          POP     HL              ;Discard return
0FF3 C9         24150          RET
0FF3            24160 FDCEND   EQU     $-1
0FF4            01720 DVREND$  EQU     $               ;Start of low I/O area, to 12FFH
                01730          IFGT    $,1200H+START$
```

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 16390 DODVR    16870
$MAIN     BAR$               0201 00640 CLOCKS   10310
BOOT4     BOOT               4300 01620
BOOT4     BOOTBUF            43F6 00100 BOOT4    02120
BOOT4     BOOTST$            439D 02470 BOOT4    02480
IODVR     BRKTEST            055E 03680 IODVR    03720
FDCDVR    BUCKET             0FE8 24080 FDCDVR   21660
BOOT4     BUFFER             1200 00090 BOOT4    02010 02080 02110 02180 02250
$MAIN     BUR$               0200 00630 CLOCKS   10190
FDCDVR    CALLIO             0F93 23480 FDCDVR   23360
KIDVR     CAPSKEY            0A79 13210 KIDVR    12770
KIDVR     CASHK$             0A7B 13220
$MAIN     CFLAG$             006C 00470 IODVR    04650
KIDVR     CKCTL              0ACE 13670
KIDVR     CKCTL1             0AC8 13620 KIDVR    13670
KIDVR     CKCTL2             0AD2 13690 KIDVR    13610
FDCDVR    CKVER              0E7D 21830 FDCDVR   21730
CLOCKS    CLOCK              0787 07930 CLOCKS   07510
DODVR     CLREOF             0D1E 18480 DODVR    17210
DODVR     CLREOF1            0D1F 18490 DODVR    18100
DODVR     CLREOF2            0D22 18500 DODVR    18440
DODVR     CLREOF3            0D36 18640 DODVR    18550
DODVR     CLREOL             0D12 18380 DODVR    17190
IODVR     CLRFRM             001F 02990 IODVR    03520
KIDVR     CLRTYP             0B65 14770 KIDVR    13830
BOOT4    +CORE$              0300 01600 BOOT4    02920
KIDVR     CR                 000D 10680 IODVR    03380
                                        KIDVR    10930 11020 11580 11600
DODVR     CRSAVE             0B97 15520 CLOCKS   07140 07350
                                        DODVR    15870 15910 16240 19740
DODVR     CRSBKSP            0C1B 16470 DODVR    16400 17070
DODVR     CRSBOL             0BF8 16150 DODVR    17170 18300 18390
DODVR     CRSCHAR            0B98 15530 CLOCKS   07370
                                        DODVR    15950 19260
DODVR     CRSDOWN            0C30 16650 DODVR    17110 18320
DODVR     CRSFRW0            0CC3 17810 DODVR    16730 17790
DODVR     CRSFRWD            0CBB 17760 DODVR    17090
DODVR     CRSHOME            0C05 16300 DODVR    17150
DODVR     CRSOFF             0C01 16240 DODVR    16950
DODVR     CRSON              0C00 16230 DODVR    16930
DODVR     CRSUP              0C2B 16590 DODVR    17130 17850
DODVR     CRTBGN$            F800 15300 BOOT4    01690 01700 02390
                                        CLOCKS   07950 08300
                                        DODVR    15310 15510 16310 16510 16690 17900
                                        DODVR    19130 20050
DODVR     CRTCADD            0088 15230
DODVR     CRTCDAT            0089 15240
BOOT4     CRTCTAB            02FD 01550 BOOT4    01320
DODVR     CRTEND             FF7F 15310 DODVR    17820 18340 18490
DODVR     CRTSIZE            0780 15280 BOOT4    01710
                                        DODVR    15310 17910 19550
DODVR     CTL                0004 15440 DODVR    15670 15840
KIDVR     CTLA2Z             0A13 12630 KIDVR    12510
KIDVR     CTLFF              0AEC 13910 KIDVR    13850
DODVR     CURSOR             0B95 15510 CLOCKS   07340
                                        DODVR    15970 18720 18950 19730 20230
$MAIN     DATE$              0033 00440 CLOCKS   07670 08170
FDCDVR    DATREG             00F3 21360 FDCDVR   22650 22940
$MAIN     DAYTBL$            04C7 01470
```

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

$MAIN     DCBKL$          0031 00850
$MAIN     DCT$            0470 01150 BOOT4    00160 00780 01970 02030
KIDVR     DELAY2          0A95 13340
$MAIN     DFLAG$          006D 00480 KIDVR    11240 14390
BOOT4     DIRTRK          4302 01640 BOOT4    01960
FDCDVR    DISKDUN         0FB2 23690 FDCDVR   23610
FDCDVR    DISKEI          0F95 23490 BOOT4    00380
BOOT4     DISKERR         43E0 02880 BOOT4    02350
CLOCKS    DIS_DO_RAM      0846 09450 CLOCKS   09290
DODVR     DOBGN           0B9C 15610 DODVR    15350
DODVR     DODATA$         0B94 15400 DODVR    15410 15610 18210 18240
$MAIN     DODCB$          0210 00710 $MAIN    00810
                                     IODVR    03290 05330
                                     DODVR    15380
DODVR     DODVR           0B88 15350 $MAIN    00720
DODVR     DOEND           0E00 20330 DODVR    15360
DODVR     DONORM          0BC4 15830 DODVR    15740
CLOCKS    DOOPREG         0858 09570 CLOCKS   09410
FDCDVR    DOWRIT          0F73 23220 FDCDVR   23180 23200
DODVR     DO_CONTROL      0C44 16830 DODVR    15720
DODVR     DO_DSPCHAR      0CB8 17710 DODVR    15830 16090
DODVR     DO_INVERT_DIS   0C8C 17300 DODVR    16350
DODVR     DO_INVERT_ENA   0C89 17270 DODVR    16970
DODVR     DO_INVERT_OFF   0C9B 17420 DODVR    16990
DODVR     DO_MASK         0000 15410 DODVR    15670 15780 15840 17570 17880
DODVR     DO_RET          0BCB 15850 DODVR    16060 16110 16840 17480
DODVR     DO_RET1         0BCC 15860 DODVR    18960
DODVR     DO_SCROLL       0CCE 17870 DODVR    18360
DODVR     DO_TABS         0BEA 16040 DODVR    15790
FDCDVR    DSELCT          00F4 21370 FDCDVR   21820 21930 22350 23960 24020
$MAIN     DSKTYP$         04C0 01410
IODVR     DSPBYT          054D 03530 IODVR    03500 04910
BOOT4     DSPLY           021B 00080
$MAIN     DTPMT$          04C2 01430
$MAIN     DVREND$         0FF4 01720 $MAIN    00670
KIDVR     DVREXIT         0AA8 13430
$MAIN     DVRHI$          0206 00670
CLOCKS    ENADIS_DO_RAM   0817 09130 CLOCKS   07290 07940 08290 08660
                                     KIDVR    13770 14420
                                     DODVR    15620 18780
BOOT4     ERRLEN          000A 02900 BOOT4    02380
FDCDVR    ERRSTRT         0FB4 23710 FDCDVR   23310
FDCDVR    ERRTRAN         0FB6 23720 FDCDVR   23750
FDCDVR    FDCADR          00F0 21320 FDCDVR   22750
FDCDVR    FDCBGN          0E8E 21980 FDCDVR   21420
FDCDVR    FDCDLY          0EE0 22410 FDCDVR   22400
FDCDVR    FDCDVR          0E3D 21420 $MAIN    01160 01180 01210 01240
FDCDVR    FDCEND          0FF3 24160 FDCDVR   21430
BOOT4     FDCMD           43D9 02840 BOOT4    02520 02640
FDCDVR    FDCRET          0F2A 22790 $MAIN    01270 01300 01330 01360
FDCDVR    FDCSTAT         00F0 21330 FDCDVR   21760 21890 23510 23810 23860 23970
$MAIN     FDDINT$         000E 00390 BOOT4    00220
                                     FDCDVR   22950
KIDVR     FIXCLR          0A84 13270 KIDVR    13110
KIDVR     FIXSCL          0A82 13260 KIDVR    13130
$MAIN     FLGTAB$         006A 00460 $MAIN    00470 00480 00490 00500 00510 00520
                                     $MAIN    00530 00540 00550 00560 00570
                                     BOOT4    00200
FDCDVR    FRCSID0         0F09 22630 FDCDVR   22590
```

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

BOOT4    GETADR         0288 00610 BOOT4    00440 00520
FDCDVR   GETCMD         0F3C 22930 FDCDVR   22910
BOOT4    GETEXT         02B1 00950 BOOT4    00330 00880
DODVR    GET_@_ROWCOL   0DAE 19620 DODVR    19060
BOOT4    GOTERR         4385 02350
KIDVR    GOTSHFT        0A6B 13120
FDCDVR   GRABNDO        0F79 23280 FDCDVR   23060 23090 23220
BOOT4    HALTS          4394 02410 BOOT4    02410
CLOCKS   HERTZ$         0750 07470
MULDIV   HEXDEC1        06FA 06860 MULDIV   06880
MULDIV   HEXDEC2        0700 06910 MULDIV   07010
$MAIN    HIGH$          040E 01050
KIDVR    HIT            0ACC 13650 KIDVR    13520
KIDVR    HITWS          0ACA 13640 KIDVR    13550
CLOCKS   HLSAV          086B 09710 CLOCKS   09150 09470
IODVR    HOME           001C 02980 IODVR    03490
CLOCKS   HXD1           07CB 08440 CLOCKS   08420
$MAIN    IFLAG$         0072 00490
$MAIN    INBUF$         0420 01100
BOOT4    INITCRTC       02DD 01300 BOOT4    01920
$MAIN    INTVC$         003E 00450
DODVR    INVIDEO        0DCB 19870 DODVR    17430 18500
IODVR    IOBGN          0648 05360 IODVR    05190 05300
FDCDVR   IORQST         0F44 23000 FDCDVR   22030
$MAIN    JCLCB$         0203 00660 IODVR    05250
$MAIN    JLDCB$         0230 00830 $MAIN    00850
                                   IODVR    03070 03160
KIDVR    KB0            F401 10690 KIDVR    11710 13920
CLOCKS   KB1            F401 08570 CLOCKS   08790
KIDVR    KB6            F440 10700
CLOCKS   KB7            F440 08650 CLOCKS   08950
KIDVR    KBROW0         0004 10840 KIDVR    11700
KIDVR    KBROW4         0008 10860 KIDVR    13160
KIDVR    KBROW6         000A 10880 KIDVR    12560 13000
KIDVR    KBTBL          0908 10930 KIDVR    12800
CLOCKS   KCK1           07F5 08920 CLOCKS   08780
CLOCKS   KCK1A          07FC 08950 CLOCKS   08860 08880
CLOCKS   KCK1B          0805 08990 CLOCKS   08940 08970
CLOCKS   KCK2           0815 09070 CLOCKS   09010 09020 09050
CLOCKS   KCK@           07D6 08660
BOOT4    KEYIN          0040 00060
KIDVR    KEYOK          0A89 13300 KIDVR    13150 13170 13280
$MAIN    KFLAG$         0074 00500 IODVR    03620
                                   CLOCKS   08670
                                   KIDVR    11300 12440 12960 13220 14820 14970
KIDVR    KIBGN          0923 11060 KIDVR    10730
KIDVR    KIDATA$        08FC 10780 KIDVR    10800 10820 10840 10860 10880 11690
                                   KIDVR    11700
$MAIN    KIDCB$         0208 00680 $MAIN    00780
                                   IODVR    03730 05270
                                   KIDVR    10760
KIDVR    KIDVR          08F0 10730 $MAIN    00690
KIDVR    KIHOOK         0B39 14490
KIDVR    KILAST         0B87 15170 KIDVR    10740
KIDVR    KISCAN         0983 11690 KIDVR    14310 14490
$MAIN    LBANK$         0202 00650 IODVR    05430
                                   CLOCKS   10020 10480 10520
BOOT4    LBOOT          0238 00160 BOOT4    02200
KIDVR    LF             000A 10670 KIDVR    10950
```

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

DODVR   LINESIZ        0050 15250 DODVR    15270 15280 16660 17930 19430 20060
                                  DODVR    20270
DODVR   LINFEED        0D02 18300 DODVR    16910
BOOT4   LOAD           0268 00410 BOOT4    00360 00490 00570
BOOT4   LOAD1          0271 00450 BOOT4    00480
BOOT4   LOAD2          027A 00510 BOOT4    00430
BOOT4   LOAD3          0281 00550 BOOT4    00560
IODVR   LOGBUF         051D 03200 IODVR    03120
$MAIN   MAXDAY$        0401 01030 CLOCKS   07680
BOOT4   MLTCA          02D3 01200 BOOT4    01240
BOOT4   MLTCA1         02D9 01240 BOOT4    01220
$MAIN   MODOUT$        0076 00510 DODVR    16320 16480 16770 17630 17650 17770
$MAIN   MONTBL$        04DC 01480
DODVR   MOVCRS         0C33 16670 DODVR    16610
DODVR   MOVLIN         0D98 19430 DODVR    19410
BOOT4   MULTCA         02CE 01160 BOOT4    01040 01090
DODVR   NEGLINE        FFB0 15270 DODVR    16600
$MAIN   NFLAG$         0077 00520 DODVR    18820
BOOT4   NMIRET         43CD 02760 BOOT4    01740
BOOT4   NMIVECT        0066 00070 BOOT4    01750 01770
IODVR   NOBRK          0575 03790 IODVR    03640
KIDVR   NOCHAR         09B5 11940 KIDVR    11640 11880 13380
KIDVR   NOKEY          0A88 13290 KIDVR    12020 13250
FDCDVR  NOPCMP         0ECB 22310 FDCDVR   22260 22290
CLOCKS  NOSOLID        072F 07290 CLOCKS   07270
BOOT4   NOSYS          43EA 02890 BOOT4    02370 02900
KIDVR   NOTALPH        0A67 13100 KIDVR    13010
BOOT4   NOTDBL         0255 00270 BOOT4    00250
BOOT4   NOTSYS         4389 02370 BOOT4    02100
DODVR   NUMROWS        0018 15260 DODVR    15280
$MAIN   OPREG$         0078 00530 BOOT4    01680
                                  CLOCKS   09350 09580 10430 10460
CLOCKS  OPREG_SV_AREA  086E 09740 CLOCKS   08930 09320
CLOCKS  OPREG_SV_PTR   0835 09330 CLOCKS   08920 09510 09570
                                  DODVR    17320
$MAIN   PAKNAM$        0410 01060
FDCDVR  PASSCMD        0F23 22750 FDCDVR   22960
$MAIN   PAUSE@         0382 00970 IODVR    03700
                                  KIDVR    13320
                                  PRDVR    20780
                                  FDCDVR   22430 24130
CLOCKS  PCSAVE$        07AF 08210 CLOCKS   08270
$MAIN   PDRV$          001B 00400 FDCDVR   21920 22360 22850
DODVR   PERR           0DED 20160 CLOCKS   09900 10010 10270
                                  DODVR    19530 19980 20010
CLOCKS  PERRX          088F 10010 CLOCKS   10330
PRDVR   PRBGN          0E0D 20530 PRDVR    20450
$MAIN   PRDCB$         0218 00740 IODVR    03240 05310
                                  PRDVR    20480
PRDVR   PRDVR          0E01 20450 $MAIN    00750
PRDVR   PREND          0E3C 21200 PRDVR    20460
PRDVR   PRPORT         00F8 20390 PRDVR    21070 21160
DODVR   PUTA@DE        0DCD 19890 DODVR    19770
DODVR   PUT_@          0DCA 19860 DODVR    16430 17720
DODVR   PUT_@_ROWCOL   0DC6 19830 DODVR    19080
PRDVR   PVAL3          0E33 21070
KIDVR   R7KFLG         0B6A 14820 KIDVR    14220
DODVR   RAMSIZE        0800 15290
BOOT4   RDB1           02AA 00870 BOOT4    00840
```

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

BOOT4    RDB2           02AE 00890 BOOT4    00750
BOOT4    RDBOOT         432D 01860 BOOT4    01910
BOOT4    RDBYTE         0297 00730 BOOT4    00410 00450 00530 00550 00610 00630
                                   BOOT4    00660
FDCDVR   RDIN           0E69 21720 FDCDVR   23080
FDCDVR   RDIN1          0E72 21760 FDCDVR   21780
FDCDVR   RDIN2          0E7B 21820 FDCDVR   21850
BOOT4    RDS1           4379 02270 BOOT4    02340
BOOT4    RDSECT         4374 02250 BOOT4    02000 02070
BOOT4    RDSEQ          4377 02260 BOOT4    01860
BOOT4    READ           4396 02430 BOOT4    02290
BOOT4    READLP1        43BD 02670 BOOT4    02690
BOOT4    READLP2        43C5 02720 BOOT4    02740
FDCDVR   RESTOR         0EA8 22120 FDCDVR   21510
FDCDVR   RETRY          0F87 23410 FDCDVR   23580 23670
$MAIN    RFLAG$         007B 00540 FDCDVR   23010
DODVR    ROWCOL_2_ADDR  0DD0 19950 DODVR    16190 18690 19360 19630 19840
KIDVR    RPTINIT        0002 10800 KIDVR    11920 13340
KIDVR    RPTRATE        0003 10820 KIDVR    12710 14680 14710
IODVR    RSTBNK         0679 05700 IODVR    05580
$MAIN    RSTOR$         04C4 01450
FDCDVR   RWINIT         0F2B 22830 FDCDVR   21740 23790
$MAIN    S1DCB$         0238 00840
DODVR    SCRPROT        0007 15420 DODVR    17890
FDCDVR   SDEN           0E5C 21600 FDCDVR   21580
FDCDVR   SECREG         00F2 21350 FDCDVR   22630
BOOT4    SECTRK         02A5 00830 BOOT4    00270
BOOT4    SEEK1          43A3 02530 BOOT4    02550
FDCDVR   SEEKTRK        0EE8 22490 FDCDVR   22050 23450
FDCDVR   SELECT         0EB0 22160 FDCDVR   22070 22710
DODVR    SET40          0C3D 16770 DODVR    17050
DODVR    SETMASK        0CFD 18240 DODVR    17580
DODVR    SETMOD         0CB2 17650 DODVR    16340 16790
FDCDVR   SETSECT        0F08 22620 FDCDVR   22570
DODVR    SET_SCROLL     0CF3 18170 DODVR    19190
$MAIN    SFLAG$         007C 00550 IODVR    04010
                                   CLOCKS   09030
                                   KIDVR    12930
KIDVR    SHIFT          F480 10710 CLOCKS   08680
                                   KIDVR    12350
CLOCKS   SHIFT          F480 08550
$MAIN    SIDCB$         0220 00770
$MAIN    SODCB$         0228 00800
KIDVR    SPCLLP         0AB7 13510 KIDVR    13590
KIDVR    SPCLTB         091E 11020 KIDVR    13460
KIDVR    SPECL          0AAB 13450 KIDVR    13410
CLOCKS   SPSAV          0863 09650 CLOCKS   09260
$MAIN    STACK$         0380 00960 $MAIN    00970
                                   CLOCKS   09270
$MAIN    START$         0000 00350 $MAIN    00590
                                   BOOT4    01830
                                   $MAIN    01730 01760
FDCDVR   STEPIN         0F1A 22710 FDCDVR   22110 22140 22680
FDCDVR   SWDEN          0E45 21480 FDCDVR   23640
DODVR    TABS           0003 15430 DODVR    15780
DODVR    TGGLALT        0CAD 17620 DODVR    17030
DODVR    TGGLCTL        0C9F 17480 DODVR    15700
DODVR    TGGLTAB        0CA6 17550 DODVR    17010
KIDVR    TGLCASE        0A3E 12900 KIDVR    12590 12620
```

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

$MAIN    TIME$          002D 00430 CLOCKS   07540 07960
CLOCKS   TIME1          0792 07980 CLOCKS   08190
CLOCKS   TIME2          0794 07990 CLOCKS   08130
CLOCKS   TIME3          0797 08010 CLOCKS   08030
$MAIN    TIMER$         002C 00420 $MAIN    00430
                                   KIDVR    11890 12700 13330 14670
CLOCKS   TIMER1         0761 07560 CLOCKS   07630
CLOCKS   TIMETBL        070F 07120 CLOCKS   07390 07550
$MAIN    TIMSL$         002B 00410
CLOCKS   TIMTSK$        0713 07130
$MAIN    TMPMT$         04C3 01440
CLOCKS   TRACE_INT      07B1 08250
FDCDVR   TRKREG         00F1 21340 FDCDVR   22510 22840
FDCDVR   TSTBSY         0E82 21890 FDCDVR   21940 22020 22160 22490 23460
KIDVR    TSTSPA         0A6F 13140
KIDVR    TYPAHD         0AD5 13760 KIDVR    11130
KIDVR    TYPBUF         FF80 15100 KIDVR    13780 14430
KIDVR    TYPHK$         0A8F 13320
KIDVR    TYPON          0B23 14330 KIDVR    13810 13870
KIDVR    TYPTSK$        0B26 14380
DODVR    VDCTL          0D61 19010 DODVR    18840 18920
FDCDVR   VERFIN         0E63 21660 FDCDVR   23110
FDCDVR   VERFY          0F58 23090 FDCDVR   23050
$MAIN    VFLAG$         007F 00560 CLOCKS   07160
                                   DODVR    15920 15940 19700
DODVR    VIDLIN         0D8B 19340 DODVR    19030
DODVR    VIDMOV1        0DA7 19550 DODVR    19150
DODVR    VIDMOVE        0D9F 19500 DODVR    19170
FDCDVR   WRCMD          0F5E 23120 FDCDVR   23030
FDCDVR   WRCMD1         0F67 23160 FDCDVR   23130
FDCDVR   WRNMIPORT      00E4 21310 FDCDVR   23900 24110
FDCDVR   WR01           0FC1 23810 FDCDVR   23830
FDCDVR   WR02           0FE2 24020 FDCDVR   24040
FDCDVR   WR03           0FD7 23950 FDCDVR   23990
FDCDVR   WROUT          0FBC 23790 FDCDVR   23240
$MAIN    ZERO$          0401 01020

00340 Symbols declared  -  00541 References
```

NOTES:

NOTES:

SYSRES - Resident portion of the DOS

This source code assembles the load module file SYS0/SYS, commonly referred to as Sysres. It contains the BDOS and other routines. It is origined at 1300H, 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         01930 IORETZ  XOR     A               ;Set Z flag--no error
136B C9         01940         RET
                01950 ;
                01960 ;       WRCHR needs the next sector - if UPDATE, ck EOF
                01970 ;
136C DD7E01     01980 WRCH2   LD      A,(IX+1)        ;Ck if UPD bit set
136F E607       01990         AND     7               ;Mask for prot level
1371 FE04       02000         CP      4               ;Check for UPD
1373 2004       02010         JR      NZ,NSEC1        ;Bypass EOF ck on > UPD
1375 CD9215     02020 NXTSECT CALL    CKEOF1          ;Ck for end of file
1378 C0         02030         RET     NZ              ;Can't extend in update mode
1379 DD7E01     02040 NSEC1   LD      A,(IX+1)        ;Read access?
137C E607       02050         AND     7
137E FE06       02060         CP      6
1380 3044       02070         JR      NC,RWRIT3       ;"Illegal access..." if not
1382 CDCB15     02080 NSEC2   CALL    IOREC           ;Calc cylinder/sector
1385 C0         02090         RET     NZ
1386 DDCB01AE   02100         RES     5,(IX+1)        ;Show buffer current
138A DD6E03     02110         LD      L,(IX+3)        ;P/u buffer address
138D DD6604     02120         LD      H,(IX+4)
1390 CDF419     02130         CALL    @RDSEC          ;Read the sector
1393 2803       02140         JR      Z,BUMPNRN       ;Go if no error
1395 FE06       02150         CP      6               ;Test for prot sector
1397 C0         02160         RET     NZ              ;Quit if error not 6
1398 DD340A     02170 BUMPNRN INC     (IX+10)         ;Inc the NRN ptr LSB
139B 2003       02180         JR      NZ,ZEROA@
139D DD340B     02190         INC     (IX+11)         ;  and MSB if necessary
13A0 AF         02200 ZEROA@  XOR     A
13A1 C9         02210         RET
                02220 ;
                02230 ;       Repositioning needs to write out the buffer
                02240 ;
13A2 DD7E01     02250 RWRIT@  LD      A,(IX+1)
13A5 E690       02260         AND     90H             ;Test for non-sector i/o and
13A7 FE90       02270         CP      90H             ;  buffer contents changed
13A9 2805       02280         JR      Z,RWRIT1        ;Go if conditions true
13AB 18F3       02290         JR      ZEROA@          ;  else no need to write
13AD CD6815     02300 @RWRIT  CALL    CKOPEN@         ;Ck file open, save regs
13B0 CD0C14     02310 RWRIT1  CALL    GETNRN          ;P/u NRN
13B3 7C         02320         LD      A,H             ;Ignore if rewound
13B4 B5         02330         OR      L
13B5 C8         02340         RET     Z
13B6 2B         02350         DEC     HL              ;Dec & reset NRN
13B7 DD750A     02360         LD      (IX+10),L
13BA DD740B     02370         LD      (IX+11),H
                02380 ;
                02390 ;       Check access protection level
                02400 ;
13BD DD7E01     02410 RWRIT2  LD      A,(IX+1)        ;Get prot
13C0 E607       02420         AND     7
13C2 FE05       02430         CP      5               ;Update access or better?
13C4 3804       02440         JR      C,RWRIT4
13C6 3E25       02450 RWRIT3  LD      A,25H           ;Illegal access error code
13C8 B7         02460         OR      A               ;Return NZ
13C9 C9         02470         RET
                02480 ;
13CA E604       02490 RWRIT4  AND     4               ;If UPDATE access, then
13CC 2805       02500         JR      Z,RWRIT5        ;  can't extend if at EOF
13CE CD9215     02510         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 DDCB01F6 03110          SET     6,(IX+1)        ;Upd eof only if NRN>EOF
143B DDCB017E 03120          BIT     7,(IX+1)        ;Jump if sector i/o only
143F 281D     03130          JR      Z,POSN1
1441 60       03140          LD      H,B             ;Record ptr to HL
1442 69       03150          LD      L,C
1443 DDB609   03160          OR      (IX+9)          ;P/u LRL
1446 2816     03170          JR      Z,POSN1         ;Skip nxt if LRL=256
1448 CDC906   03180          CALL    @MUL16          ;Calc sector & offset
144B 44       03190          LD      B,H             ;Physical sector =>BC
144C 4D       03200          LD      C,L
144D DD7705   03210          LD      (IX+5),A        ;Set byte ptr
1450 DDCB016E 03220          BIT     5,(IX+1)        ;Jump if buffer does not
1454 200B     03230          JR      NZ,POSN2        ;  contain current sector
1456 CD0C14   03240          CALL    GETNRN          ;P/u the NRN
1459 37       03250          SCF
145A ED42     03260          SBC     HL,BC
145C 2812     03270          JR      Z,$CKEOF        ;Pass on to CKEOF
145E DD7705   03280 POSN1    LD      (IX+5),A        ;Offset in buffer
1461 C5       03290 POSN2    PUSH    BC
1462 CDA213   03300 POSN2A   CALL    RWRIT@          ;Write current if needed
1465 C1       03310          POP     BC              ;  before moving
1466 C0       03320          RET     NZ              ;Back on write error
1467 DD710A   03330          LD      (IX+10),C       ;NRN
146A DD700B   03340          LD      (IX+11),B
146D CD2A13   03350          CALL    SET5            ;Show bfr does not
1470 C39215   03360 $CKEOF   JP      CKEOF1          ;  contain current sector
              03370 ;
              03380 ;        Entry to force a physical read
              03390 ;
1473 CD6815   03400 @RREAD   CALL    CKOPEN@
1476 0E01     03410          LD      C,1             ;Cause ADJUST to bump
              03420 ;                                ;  NRN when called
1478 CD0C14   03430 BKSP1    CALL    GETNRN          ;Get current record #
147B 7C       03440          LD      A,H             ;If file is rewound,
147C B5       03450          OR      L               ;  then ignore the req
147D 2815     03460          JR      Z,BKSP0         ;  & force OFFSET = 0
147F 2B       03470          DEC     HL              ;Back up by 1
1480 CDBA15   03480          CALL    ADJ2            ;RET if sector I/O only,
              03490                                  ;  else bump fwd if RREAD
              03500                                  ;  then back up if bit 5=0
1483 E5       03510          PUSH    HL              ;Will be popped into BC
1484 18DC     03520          JR      POSN2A          ;Finish the job
              03530 ;
              03540 ;        Entry to backspace one logical record
              03550 ;
1486 CD6815   03560 @BKSP    CALL    CKOPEN@
1489 4F       03570          LD      C,A             ;Keep ADJUST from bumping
148A DD4609   03580          LD      B,(IX+9)        ;P/u LRL
148D B0       03590          OR      B               ;Is it a 0
148E 28E8     03600          JR      Z,BKSP1         ;Go if so
1490 DD7E05   03610          LD      A,(IX+5)        ;P/u next byte pointer
1493 90       03620          SUB     B               ;Sub one record length
1494 DD7705   03630 BKSP0    LD      (IX+5),A
1497 38DF     03640          JR      C,BKSP1         ;Go if crossed sec bdry
1499 AF       03650          XOR     A               ;  else all done
149A C9       03660          RET
              03670 ;
              03680 ;        Entry to Rewind to beginning
              03690 ;
```

File positioning subroutines

```
149B CD6815     03700 @REW     CALL     CKOPEN@
149E 47         03710          LD       B,A              ;Zero NRN
149F 4F         03720          LD       C,A
14A0 18BC       03730          JR       POSN1            ;Will also zero offset
                03740 ;
                03750 ;        Entry to Position to end-of-file
                03760 ;
14A2 CD6815     03770 @PEOF    CALL     CKOPEN@
14A5 DD4E0C     03780          LD       C,(IX+12)        ;ERN to BC
14A8 DD460D     03790          LD       B,(IX+13)
14AB DDB608     03800          OR       (IX+8)           ;P/u EOF byte
14AE 28AE       03810          JR       Z,POSN1          ;Go if full sector
14B0 0B         03820          DEC      BC               ;Point to last rec
14B1 18AB       03830          JR       POSN1            ;Use POSN to get end
                03840 ;
                03850 ;        Entry to Locate current record number
                03860 ;
14B3 CD6815     03870 @LOC     CALL     CKOPEN@
14B6 CD0C14     03880          CALL     GETNRN           ;P/u NRN
14B9 CDB715     03890          CALL     ADJUST           ;Get offset and adj NRN
14BC DD5E09     03900 LOC1     LD       E,(IX+9)         ;P/u LRL
14BF 7B         03910          LD       A,E              ;Test LRL for zero
14C0 B7         03920          OR       A                ;If zero, then give NRN
14C1 2816       03930          JR       Z,LOC3           ;LRL=0, NRN is correct
14C3 0C         03940          INC      C                ;If offset is zero,
14C4 0D         03950          DEC      C                ;  then it's at 256,
14C5 2801       03960          JR       Z,LOC2           ;  and we don't dec NRN
14C7 2B         03970          DEC      HL
                03980 ;
                03990 ;        Divide the three byte pointer (HLC) by the LRL
                04000 ;
14C8 CDE306     04010 LOC2     CALL     @DIV16           ;Divide (NRN-1)/LRL
14CB 45         04020          LD       B,L              ;Save high order result
14CC 54         04030          LD       D,H              ;Save possible overflow
14CD 67         04040          LD       H,A              ;Prepare 2nd dividend
14CE 69         04050          LD       L,C              ;P/u low order dividend
14CF 7B         04060          LD       A,E              ;P/u LRL divisor again
14D0 CDE306     04070          CALL     @DIV16
14D3 60         04080          LD       H,B              ;Xfer high order result
14D4 B7         04090          OR       A                ;If remainder, we have a
14D5 2801       04100          JR       Z,$+3            ;  partial record to round
14D7 23         04110          INC      HL               ;  up to next record #
14D8 7A         04120          LD       A,D              ;Xfer possible overflow
14D9 C1         04130 LOC3     POP      BC               ;Pop RESTREG return adr
14DA E3         04140          EX       (SP),HL          ;Exchange value with BC
14DB C5         04150          PUSH     BC               ;Restore RESTREG
                04160 ;
                04170          IF       @MOD4
14DC            04180 ORARET@  EQU      $
                04190          ENDIF
14DC B7         04200          OR       A
14DD C9         04210          RET
                04220 ;
                04230 ;        Entry to Locate the end-of-file record
                04240 ;
14DE CD6815     04250 @LOF     CALL     CKOPEN@
14E1 DD6E0C     04260          LD       L,(IX+12)        ;P/u ERN
14E4 DD660D     04270          LD       H,(IX+13)
14E7 DD4E08     04280          LD       C,(IX+8)         ;EOF byte
```

File positioning subroutines

```
14EA 18D0     04290          JR      LOC1            ;Handle all LRLs
              04300 ;
              04310 ;        Entry to Write an end-of-file mark
              04320 ;
14EC CD6815   04330 @WEOF    CALL    CKOPEN@
14EF CDA213   04340          CALL    RWRIT@          ;Write buffer if needed
14F2 DD4607   04350 WEOF1    LD      B,(IX+7)        ;P/u DEC of FPDE
14F5 DD4E06   04360          LD      C,(IX+6)        ;P/u drive #
14F8 CDBB18   04370          CALL    @DIRRD          ;Read file's dir record
14FB C0       04380          RET     NZ              ;Back if read error
14FC 2C       04390          INC     L               ;Pt to ERN offset
14FD 2C       04400          INC     L
14FE 2C       04410          INC     L
14FF DD7E08   04420          LD      A,(IX+8)        ;P/u EOF offset
1502 77       04430          LD      (HL),A          ;Put in direc
1503 111100   04440          LD      DE,17           ;Pt to EOF in dir
1506 19       04450          ADD     HL,DE
1507 DD7E0C   04460          LD      A,(IX+12)       ;P/u lo EOF
150A 77       04470          LD      (HL),A          ;Put EOF in direc
150B 23       04480          INC     HL
150C DD7E0D   04490          LD      A,(IX+13)       ;P/u hi EOF
150F 77       04500          LD      (HL),A
1510 C30318   04510          JP      @DIRWR          ;Write direc and return
              04520 ;
              04530 ;        Entry to Read a record
              04540 ;
1513 CD6815   04550 @READ    CALL    CKOPEN@
1516 E5       04560          PUSH    HL
1517 CDA213   04570          CALL    RWRIT@          ;Write buffer if needed
151A E1       04580          POP     HL
151B C0       04590          RET     NZ              ;Back on write error
151C DD4609   04600          LD      B,(IX+9)        ;P/u LRL
151F 78       04610          LD      A,B             ;If LRL=256, just
1520 B7       04620          OR      A
1521 CA7513   04630          JP      Z,NXTSECT       ;  get the next sector
1524 E5       04640 RDREC    PUSH    HL              ;Save buffer posn
1525 C5       04650          PUSH    BC              ;Save LRL
1526 CD1213   04660          CALL    RDCHAR          ;Read next byte
1529 C1       04670          POP     BC
152A E1       04680          POP     HL
152B C0       04690          RET     NZ              ;Back on read error
152C 77       04700          LD      (HL),A          ;Put char into buffer
152D 23       04710          INC     HL              ;Bump buffer ptr
152E 10F4     04720          DJNZ    RDREC           ;Loop for entire record
1530 C9       04730          RET
              04740 ;
              04750 ;        Entry to Write a record
              04760 ;
1531 CD6815   04770 @WRITE   CALL    CKOPEN@
1534 32EA13   04780 WRIT1    LD      (VEROP+1),A     ;Turn on/off verify
1537 DD4609   04790          LD      B,(IX+9)        ;P/u LRL
153A 78       04800          LD      A,B             ;Bypass if LRL=256
153B B7       04810          OR      A
153C CABD13   04820          JP      Z,RWRIT2
153F E5       04830          PUSH    HL              ;Save some FCB values
1540 DD6605   04840          LD      H,(IX+5)        ;P/u buffer offset loc
1543 DD6E08   04850          LD      L,(IX+8)        ;P/U EOF offset byte
1546 E3       04860          EX      (SP),HL         ;Put values on stack
              04870                                  ;  and recover HL
```

File positioning subroutines

```
1547 7E         04880 WRREC   LD      A,(HL)          ;Pass the logical record
1548 23         04890         INC     HL              ;  to the writing routine
1549 E5         04900         PUSH    HL              ;  byte by byte
154A C5         04910         PUSH    BC
154B CD2F13     04920         CALL    WRCHAR
154E C1         04930         POP     BC
154F E1         04940         POP     HL
1550 2005       04950         JR      NZ,WRERROR      ;Exit and fix FCB
1552 10F3       04960         DJNZ    WRREC           ;Loop for entire record
1554 E3         04970         EX      (SP),HL         ;Remove stored FCB info
1555 E1         04980         POP     HL              ;Recover HL
1556 C9         04990         RET
1557 E3         05000 WRERROR EX      (SP),HL         ;Get FCB Values
1558 DD7405     05010         LD      (IX+5),H        ;  and put them back
155B DD7508     05020         LD      (IX+8),L
155E E1         05030         POP     HL              ;Restore HL
155F C9         05040         RET                     ;Go back with error
                05050 ;
                05060 ;       Entry to Verify after write of a record
                05070 ;
1560 CD6815     05080 @VER    CALL    CKOPEN@
1563 3C         05090         INC     A               ;Set verify byte
1564 18CE       05100         JR      WRIT1
1566 37         05110 LNKFCB@ SCF                     ;Init to force file open
1567 D2         05120         DB      0D2H            ;  test by JP NC,aaaa
1568 1A         05130 CKOPEN@ LD      A,(DE)          ;Ignore if from LNKFCB
1569 07         05140         RLCA                    ;Test hi bit of FCB
156A E3         05150         EX      (SP),HL
156B 222600     05160         LD      (JRET$),HL      ;Save ret
156E ED532400   05170         LD      (JDCB$),DE      ;Save DCB
1572 E3         05180         EX      (SP),HL
1573 300F       05190         JR      NC,NOTOPEN      ;Go if not an open FCB
1575 F1         05200         POP     AF              ;Get return
1576 D5         05210         PUSH    DE              ;Dcb addr to IX
1577 DDE3       05220         EX      (SP),IX
1579 E5         05230         PUSH    HL              ;Save regs
157A D5         05240         PUSH    DE
157B C5         05250         PUSH    BC
157C E5         05260         PUSH    HL              ;Estab ret
157D 218915     05270         LD      HL,RESTREG      ;  to restore registers
1580 E3         05280         EX      (SP),HL
1581 F5         05290         PUSH    AF              ;Put back ret
1582 AF         05300         XOR     A
1583 C9         05310         RET                     ;Go back
                05320 ;
1584 F1         05330 NOTOPEN POP     AF
1585 3E26       05340         LD      A,26H           ;File not open
1587 B7         05350         OR      A
1588 C9         05360         RET
                05370 ;
1589 C1         05380 RESTREG POP     BC              ;Pop back registers save
158A D1         05390         POP     DE              ;  in CKOPEN@
158B E1         05400         POP     HL
158C DDE1       05410         POP     IX
158E C9         05420         RET
                05430 ;
                05440 ;       Entry to Check if at end-of-file
                05450 ;
158F CD6815     05460 @CKEOF  CALL    CKOPEN@
```

File positioning subroutines

```
1592 CD0C14    05470 CKEOF1  CALL    GETNRN          ;P/U NRN into HL
1595 E5        05480         PUSH    HL              ;Save un-adjusted NRN
1596 CDB715    05490         CALL    ADJUST          ;Adjust for special cases
1599 7C        05500         LD      A,H             ;Compare hi byte
159A DDBE0D    05510         CP      (IX+13)
159D 200E      05520         JR      NZ,CKEOF2       ;Go if not equal
159F 7D        05530         LD      A,L             ;Compare lo byte
15A0 DDBE0C    05540         CP      (IX+12)
15A3 2008      05550         JR      NZ,CKEOF2       ;Go if not equal
15A5 0D        05560         DEC     C               ;Adjust for 00=256
15A6 DD7E08    05570         LD      A,(IX+8)        ;Compare offset byte
15A9 3D        05580         DEC     A
15AA 91        05590         SUB     C
15AB 3F        05600         CCF
15AC 03        05610         INC     BC              ;Restore old C value
15AD E1        05620 CKEOF2  POP     HL              ;Restore unadjusted NRN
15AE 3E1D      05630         LD      A,1DH           ;Rec # out of range code
15B0 2002      05640         JR      NZ,CKEOF3       ;Go if not at EOF
15B2 3D        05650         DEC     A               ;X'1C'=EOF encountered
15B3 C9        05660         RET                     ;Return with NZ flag
15B4 D0        05670 CKEOF3  RET     NC              ;Return with error
15B5 AF        05680         XOR     A               ;No error
15B6 C9        05690         RET
               05700 ;
               05710 ;       File positioning adjustment routines
               05720 ;
15B7           05730 ADJUST  EQU     $               ;Entry from @CKEOF & @LOC
15B7 DD4E05    05740         LD      C,(IX+5)        ;Pick up offset
15BA           05750 ADJ2    EQU     $               ;Entry from @BKSP/@RREAD
15BA DDCB017E  05760         BIT     7,(IX+1)        ;Sector I/O only?
15BE C8        05770         RET     Z               ;No adjustment if so
15BF 79        05780         LD      A,C             ;Offset =0? (or "RREAD?")
15C0 B7        05790         OR      A
15C1 2801      05800         JR      Z,$+3           ;Go if zero
15C3 23        05810         INC     HL              ;Adjust
15C4 DDCB016E  05820         BIT     5,(IX+1)        ;Check magic bit
15C8 C0        05830         RET     NZ              ;Go if set
15C9 2B        05840         DEC     HL              ;Adjust
15CA C9        05850         RET
               05860 ;
               05870 ;       Calculate the cylinder/sector of needed record
               05880 ;
15CB CD0C14    05890 IOREC   CALL    GETNRN          ;P/u record number
15CE CD261A    05900         CALL    @DCTBYT-5       ;Get # of sectors/gran
15D1 E61F      05910         AND     1FH
15D3 3C        05920         INC     A
15D4 CDE306    05930         CALL    @DIV16          ;By # of sectors/gran
15D7 326016    05940         LD      (CALS5+1),A     ;Sv rmndr (sector offset)
15DA DDE5      05950         PUSH    IX              ;Xfer fcb to HL
15DC E3        05960         EX      (SP),HL
15DD 010E00    05970         LD      BC,14           ;Pt to 1st extent info
15E0 09        05980         ADD     HL,BC
15E1 C1        05990         POP     BC              ;Pop gran ptr HL into BC
15E2 3E05      06000         LD      A,5             ;Init to ck 4 extents
15E4 110000    06010         LD      DE,0            ;  & extended FXDE ptr
15E7 F5        06020 GREC1   PUSH    AF
15E8 7E        06030         LD      A,(HL)          ;P/u starting cyl byte
15E9 23        06040         INC     HL              ;  & bypass if FF
15EA 3C        06050         INC     A
```

File positioning subroutines

```
15EB 280B      06060         JR      Z,GREC2
15ED E5        06070         PUSH    HL             ;Xfer the # of grans up
15EE 62        06080         LD      H,D            ;  to but not including
15EF 6B        06090         LD      L,E            ;  this extent into HL
15F0 AF        06100         XOR     A              ;Sub gran pointer from
15F1 ED42      06110         SBC     HL,BC          ;  cumulative figure & go
15F3 380E      06120         JR      C,GREC3        ;  if not in previous ext
15F5 E1        06130         POP     HL
15F6 2829      06140         JR      Z,CALCSEC
15F8 23        06150 GREC2   INC     HL
15F9 F1        06160         POP     AF
15FA 3D        06170         DEC     A
15FB 2819      06180         JR      Z,GREC4        ;Jump when all quads c'kd
15FD 5E        06190         LD      E,(HL)         ;P/u cumulative # grans
15FE 23        06200         INC     HL             ;  up to but not
15FF 56        06210         LD      D,(HL)         ;  including this extent
1600 23        06220         INC     HL
1601 18E4      06230         JR      GREC1
1603 24        06240 GREC3   INC     H              ;Within 256 grans?
1604 7D        06250         LD      A,L            ;Xfer lo-order difference
1605 E1        06260         POP     HL             ;Rcvr # of contig grans
               06270 ;                              ;  in this extent
1606 20F0      06280         JR      NZ,GREC2       ;Go if not within 256
1608 D5        06290         PUSH    DE             ;Save cumulative count
1609 5F        06300         LD      E,A            ;Xfer gran dif (neg)
160A 7E        06310         LD      A,(HL)         ;P/u # of grans
160B E61F      06320         AND     1FH            ;  in this extent
160D 83        06330         ADD     A,E            ;Add to negative diff
160E 7B        06340         LD      A,E            ;Put neg diff into A
160F D1        06350         POP     DE
1610 30E6      06360         JR      NC,GREC2       ;Go if not in this extent
1612 ED44      06370         NEG                    ;Is in this extent, make
1614 180B      06380         JR      CALCSEC        ;  diff positive & use it
               06390 ;
               06400 ;       All current quads checked - Need directory info
               06410 ;
               06420 GREC4
1616 CD6416    06430         CALL    ALLOC          ;Get # of grans
1619 C0        06440         RET     NZ             ;  into the extent
161A 325016    06450         LD      (CALS4+1),A    ;  or error RET
161D 302A      06460         JR      NC,CALS3       ;Jp if record in 1st ext
161F 181F      06470         JR      CALS1          ;  else jp if in another
               06480 ;
               06490 ;       Calc sector in gran
               06500 ;
1621 325016    06510 CALCSEC LD      (CALS4+1),A    ;Stuff # grans into
1624 46        06520         LD      B,(HL)         ;  this extent
1625 2B        06530         DEC     HL             ;P/u # contig grans &
1626 4E        06540         LD      C,(HL)         ;  rel start & start cyl
1627 23        06550         INC     HL
1628 F1        06560         POP     AF             ;Rcvr # of quad
1629 2F        06570         CPL
162A C604      06580         ADD     A,4
162C 3019      06590         JR      NC,CALS2       ;Jump if 1st ext or quad
162E 3C        06600         INC     A              ;If not 1st, set up to move
162F 07        06610         RLCA                   ;  matching quad to the
1630 07        06620         RLCA                   ;  first position by
1631 C5        06630         PUSH    BC             ;  shuffling the others up
1632 D5        06640         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

```
1680 BB        07240          CP      E                ;  needed gran alloc
1681 2006      07250          JR      NZ,ALL2          ;  appear in this extent?
1683 7E        07260          LD      A,(HL)           ;Now see if needed gran is
1684 AA        07270          XOR     D                ;  in this extent field
1685 E6E0      07280          AND     0E0H             ;  by checking its starting gran
1687 2819      07290          JR      Z,ALL4
1689 05        07300 ALL2     DEC     B                ;Dec the count down loop
168A 2805      07310          JR      Z,ALL3           ;Done if no match
168C 23        07320          INC     HL               ;Go to next extent
168D 23        07330          INC     HL               ;  info in FCB
168E 23        07340          INC     HL
168F 18ED      07350          JR      ALL1
1691 D5        07360 ALL3     PUSH    DE               ;Save needed extent info
1692 EB        07370          EX      DE,HL            ;Set up to shuffle extent
1693 21FCFF    07380          LD      HL,-4            ;  info
1696 19        07390          ADD     HL,DE
1697 010C00    07400          LD      BC,12
169A EDB8      07410          LDDR
169C EB        07420          EX      DE,HL
169D C1        07430          POP     BC
169E AF        07440          XOR     A                ;Set Z, no error
169F 37        07450          SCF                      ;Set CF, extent not found
16A0 1803      07460          JR      ALL5
16A2 72        07470 ALL4     LD      (HL),D
16A3 EB        07480          EX      DE,HL
16A4 AF        07490          XOR     A                ;Set Z no error
16A5 D1        07500 ALL5     POP     DE
16A6 3E00      07510 ALL6     LD      A,0              ;# of grans into this ext
16A8 C9        07520          RET                      ;Where desired gran is
               07530 ;
               07540 ;        Extent is unused - need to allocate more space
               07550 ;
16A9 CDF216    07560 CG06     CALL    CG07             ;Try to allocate more
16AC C1        07570          POP     BC               ;Get back desired gran
16AD C0        07580          RET     NZ               ;Return on error
               07590                                   ;Look for gran again
               07600 ;
               07610 ;        Find extent containing desired granule
               07620 ;
16AE C5        07630 CYL_GRN  PUSH    BC               ;Save desired gran #
16AF 110000    07640          LD      DE,0             ;Init gran counter
16B2 DD4607    07650          LD      B,(IX+7)         ;P/u DEC of file
16B5 78        07660 CG01     LD      A,B
16B6 32AC17    07670          LD      (STUFDEC+1),A    ;Stuff
16B9 DD4E06    07680          LD      C,(IX+6)         ;P/u drive for file
16BC CDBB18    07690          CALL    @DIRRD           ;Read its directory
16BF 011600    07700          LD      BC,22            ;Point to 1st extent
16C2 09        07710          ADD     HL,BC            ;  of its directory
16C3 EB        07720          EX      DE,HL            ;Gran count to HL
16C4 C1        07730          POP     BC               ;Restore desired gran
16C5 C0        07740          RET     NZ               ;Return on read error
16C6 1A        07750 CG02     LD      A,(DE)           ;Is this extent
16C7 FEFE      07760          CP      0FEH             ;  allocated?
16C9 301F      07770          JR      NC,CG05          ;Jump if it is not
16CB 13        07780          INC     DE               ;Point to allocation
16CC 1A        07790          LD      A,(DE)           ;P/u relative gran & #
16CD E5        07800          PUSH    HL               ;  of contiguous grans
16CE E61F      07810          AND     1FH              ;Keep contiguous grans
16D0 3C        07820          INC     A                ;  & bump for 0 offset
```

File positioning subroutines

```
16D1 85         07830          ADD     A,L             ;Add to count in HL
16D2 6F         07840          LD      L,A
16D3 3001       07850          JR      NC,CG03
16D5 24         07860          INC     H               ;Bump hi order
16D6 E5         07870 CG03     PUSH    HL              ;Save gran count to
16D7 2B         07880          DEC     HL              ;  end of extent
16D8 AF         07890          XOR     A               ;Test if EOF is in this
16D9 ED42       07900          SBC     HL,BC           ;  allocation
16DB E1         07910          POP     HL
16DC 3004       07920          JR      NC,CG04         ;EOF not > this alloc
16DE 13         07930          INC     DE              ;Get rid of old
16DF F1         07940          POP     AF              ;  current quantity
16E0 18E4       07950          JR      CG02            ;Check next extent
                07960 ;
                07970 ;        The EOF is within this allocation. Recover
                07980 ;        the allocation data and exit
                07990 ;
16E2 E1         08000 CG04     POP     HL              ;P/u gran count to extent
16E3 EB         08010          EX      DE,HL           ;Gran count to DE
16E4 7E         08020          LD      A,(HL)          ;P/u granule data
16E5 2B         08030          DEC     HL
16E6 6E         08040          LD      L,(HL)          ;P/u starting cylinder
16E7 67         08050          LD      H,A
16E8 AF         08060          XOR     A
16E9 C9         08070          RET
                08080 ;
                08090 ;        This extent is 1) unused, or 2) FXDE pointer
                08100 ;        and the needed gran has not been found yet
                08110 ;
16EA C5         08120 CG05     PUSH    BC              ;Gran count to DE &
16EB EB         08130          EX      DE,HL           ;DIR ptr to HL
16EC 20BB       08140          JR      NZ,CG06         ;Jump if unused
16EE 23         08150          INC     HL              ;Point to DEC of FXDE
16EF 46         08160          LD      B,(HL)          ;P/u the DEC
16F0 18C3       08170          JR      CG01            ;  & loop
                08180 ;
                08190 ;        See if the drive has enough free space left
                08200 ;
16F2 C5         08210 CG07     PUSH    BC              ;Save needed gran
16F3 DD4E06     08220          LD      C,(IX+6)        ;P/u file's drive
16F6 CD7418     08230          CALL    @GATRD          ;Get GAT
16F9 C1         08240          POP     BC              ;Rcvr needed gran
16FA C0         08250          RET     NZ              ;Return if GAT error
16FB E5         08260          PUSH    HL
16FC 60         08270          LD      H,B             ;Xfer the requested
16FD 69         08280          LD      L,C             ;  gran to HL &
16FE AF         08290          XOR     A               ;  subtract current gran
16FF ED52       08300          SBC     HL,DE           ;Count to calculate how
1701 44         08310          LD      B,H             ;  many excess grans
1702 4D         08320          LD      C,L             ;  are needed
1703 03         08330          INC     BC
1704 D1         08340          POP     DE              ;Rcvr dir byte ptr
1705 13         08350          INC     DE              ;Pt to next DIR byte
1706 2623       08360          LD      H,DIRBUF$<-8    ;Start looking at TRK #1
1708 3A6A00     08370          LD      A,(AFLAG$)      ;P/u Search start CYL
170B 6F         08380          LD      L,A             ;  and put it in L
170C C5         08390          PUSH    BC              ;Save excess grans needed
170D 7B         08400          LD      A,E             ;Is this extent the 1st?
170E E61E       08410          AND     1EH             ;Jump if so, else we can
```

File positioning subroutines

```
1710 FE16      08420        CP      16H             ;  use it for allocation
1712 2842      08430        JR      Z,CG14
1714 1D        08440        DEC     E               ;Backup to previous
1715 1D        08450        DEC     E               ;  extent
1716 1A        08460 CG12   LD      A,(DE)          ;P/u # of contig grans to
1717 E61F      08470        AND     1FH             ;  see if the last gran
1719 3C        08480        INC     A               ;  used can be extended
171A 4F        08490        LD      C,A             ;Is current # the max
171B FE20      08500        CP      20H             ;  an extent can hold?
171D 2820      08510        JR      Z,CG13          ;Jump if a full extent
171F 1A        08520        LD      A,(DE)          ;  (32 grans max) - else
1720 E6E0      08530        AND     0E0H            ;  p/u the relative
1722 07        08540        RLCA                    ;  granule offset
1723 07        08550        RLCA
1724 07        08560        RLCA
1725 81        08570        ADD     A,C             ;Add the # of contiguous
1726 D5        08580        PUSH    DE              ;  granules
1727 CD1919    08590        CALL    RELCYL          ;Calc relative cyl needed
172A 47        08600        LD      B,A             ;Save offset
172B 4B        08610        LD      C,E
172C D1        08620        POP     DE
172D 1B        08630        DEC     DE              ;Backup to starting cyl
172E 1A        08640        LD      A,(DE)
172F 13        08650        INC     DE              ;  & repoint to alloc byte
1730 80        08660        ADD     A,B             ;Add cyls used to
1731 6F        08670        LD      L,A             ;  starting cyl
1732 2623      08680        LD      H,DIRBUF$<-8    ;Is it less than max?
1734 FECB      08690        CP      0CBH
1736 3007      08700        JR      NC,CG13         ;Jump if too big
1738 79        08710        LD      A,C
1739 46        08720        LD      B,(HL)          ;P/u the cyl's GAT
173A CD5B18    08730        CALL    TSTBIT          ;Test if gran is free
173D 284B      08740        JR      Z,CG21          ;Bypass if free gran
               08750 ;
               08760 ;      The next gran cannot be used - get another extent
               08770 ;
173F 1C        08780 CG13   INC     E               ;Else point to next
1740 1C        08790        INC     E               ;  extent field
1741 7B        08800        LD      A,E
1742 E61E      08810        AND     1EH             ;Jump if not on the FXDE
1744 FE1E      08820        CP      1EH             ;  field, else we have to
1746 200E      08830        JR      NZ,CG14         ;  obtain an FXDE record
               08840 ;
               08850 ;      Last extent used up, get new dir rec for FXDE
               08860 ;
1748 CDA417    08870        CALL    CG23            ;Write curent GAT & HIT
174B C1        08880        POP     BC
174C C0        08890        RET     NZ              ;Ret if GAT/HIT error
174D C5        08900        PUSH    BC
174E CDAF17    08910        CALL    NEWHIT          ;Get new HIT for FXDE
1751 C1        08920        POP     BC
1752 C0        08930        RET     NZ              ;Loop to process
1753 C3AE16    08940        JP      CYL_GRN         ;  new extent
               08950 ;
               08960 ;      Extent is vacant - use it & get new allocation
               08970 ;
1756 CDFE18    08980 CG14   CALL    MAXCYL          ;Get highest # cyl
1759 326017    08990        LD      (CG17+1),A      ;Stuff highest cyl
175C 0602      09000        LD      B,2
```

File positioning subroutines

```
175E 7D         09010 CG16    LD      A,L             ;Test last cyl used
175F FE00       09020 CG17    CP      0               ;P/u max cyl
1761 3007       09030         JR      NC,CG18
1763 7E         09040         LD      A,(HL)          ;P/u a GAT byte
1764 3C         09050         INC     A
1765 2010       09060         JR      NZ,CG19         ;Go if space in this cyl
1767 2C         09070         INC     L               ;  else bump to next one
1768 18F4       09080         JR      CG16            ;  & loop
176A 2E00       09090 CG18    LD      L,0             ;Now start from begin
176C 10F0       09100         DJNZ    CG16            ;  of disk & recheck
176E C1         09110         POP     BC
176F CDA417     09120         CALL    CG23            ;Write out GAT & HIT
1772 C0         09130         RET     NZ
1773 3E1B       09140         LD      A,1BH           ;"disk space full"
1775 B7         09150         OR      A
1776 C9         09160         RET
                09170 ;
                09180 ;       Found available space in cylinder
                09190 ;
1777 3EFF       09200 CG19    LD      A,0FFH          ;Set DIR extent to FF
1779 12         09210         LD      (DE),A
177A 0E00       09220         LD      C,0
177C 46         09230         LD      B,(HL)          ;P/u current GAT alloc
177D 79         09240 CG20    LD      A,C
177E CD5B18     09250         CALL    TSTBIT          ;Find a free gran
1781 2807       09260         JR      Z,CG21          ;  & jump when found
1783 1A         09270         LD      A,(DE)          ;  else advance starting
1784 C620       09280         ADD     A,20H           ;  rel gran value
1786 12         09290         LD      (DE),A
1787 0C         09300         INC     C               ;Bump pointer to test
1788 18F3       09310         JR      CG20            ;  next gran
                09320 ;
                09330 ;       Next gran in line is free - allocate it
                09340 ;
178A 79         09350 CG21    LD      A,C
178B CD6818     09360         CALL    SETBIT          ;Show it allocated
178E B6         09370         OR      (HL)
178F 77         09380         LD      (HL),A
1790 1D         09390         DEC     E               ;Backup to starting cyl
1791 1A         09400         LD      A,(DE)          ;Bump by one to see if
1792 3C         09410         INC     A               ;  this alloc is the 1st
1793 2002       09420         JR      NZ,CG22         ;  one for the extent &
1795 7D         09430         LD      A,L             ;  we have to set the
                09440                                 ;  starting cylinder
1796 12         09450         LD      (DE),A          ;Stuff starting cyl
1797 1C         09460 CG22    INC     E
1798 1A         09470         LD      A,(DE)          ;Add 1 to # of contiguous
1799 3C         09480         INC     A               ;  granules
179A 12         09490         LD      (DE),A
179B C1         09500         POP     BC              ;Decrement needed gran
179C 0B         09510         DEC     BC              ;  count since we just
179D C5         09520         PUSH    BC              ;  allocated one
179E 78         09530         LD      A,B             ;Loop if we need more
179F B1         09540         OR      C               ;  space allocated
17A0 C21617     09550         JP      NZ,CG12
17A3 C1         09560         POP     BC
17A4 DD4E06     09570 CG23    LD      C,(IX+6)        ;Else p/u the drive #
17A7 CD7518     09580         CALL    @GATWR          ;  & write out the GAT
17AA C0         09590         RET     NZ
```

File positioning subroutines

```
17AB 0600       09600 STUFDEC LD      B,0             ;P/u DEC of FPDE
17AD 1854       09610         JR      @DIRWR
                09620 ;
                09630 ;       Get new HIT for FXDE
                09640 ;
17AF DD4E06     09650 NEWHIT  LD      C,(IX+6)        ;P/u drive #
17B2 CD9718     09660         CALL    @HITRD          ;Read the HIT
17B5 C0         09670         RET     NZ
17B6 DD7E07     09680         LD      A,(IX+7)        ;P/u FPDE DEC so 1st ck
17B9 E61F       09690         AND     1FH             ;  will be for next
17BB CD1F18     09700         CALL    NHIT4           ;  in line
17BE 3E1E       09710         LD      A,1EH           ;Init "full directory...
17C0 C0         09720         RET     NZ              ;Ret if no space
17C1 45         09730         LD      B,L             ;Set DEC for
17C2 7D         09740         LD      A,L             ;  directory read
17C3 320218     09750         LD      (NHIT3+1),A     ;Stuff new DEC from HIT
17C6 54         09760         LD      D,H
17C7 DD5E07     09770         LD      E,(IX+7)        ;P/u current DEC
17CA 1A         09780         LD      A,(DE)          ;Copy filespec hash code
17CB 77         09790         LD      (HL),A          ;  to new DEC
17CC CD9818     09800         CALL    @HITWR
17CF CCBB18     09810         CALL    Z,@DIRRD
17D2 C0         09820         RET     NZ
17D3 3690       09830         LD      (HL),90H        ;Show dir rec in use as
17D5 2C         09840         INC     L               ;  FXDE record
17D6 C5         09850         PUSH    BC              ;P/u DEC of FPDE &
17D7 3AAC17     09860         LD      A,(STUFDEC+1)   ;  stuff it into FXDE's
17DA 77         09870         LD      (HL),A          ;  DIR+1 to link back
17DB 2C         09880         INC     L
17DC 0614       09890         LD      B,20            ;Zero out 20 bytes
17DE 3600       09900 NHIT1   LD      (HL),0          ;  in the FXDE
17E0 2C         09910         INC     L
17E1 10FB       09920         DJNZ    NHIT1
17E3 E5         09930         PUSH    HL              ;Save ptr to 1st extent
17E4 060A       09940         LD      B,10            ;Init to X'FF' 10 bytes
17E6 36FF       09950 NHIT2   LD      (HL),0FFH       ;  or 5 extents
17E8 2C         09960         INC     L
17E9 10FB       09970         DJNZ    NHIT2
17EB D1         09980         POP     DE              ;Rcvr ptr to 1st extent
17EC 13         09990         INC     DE              ;Pt to allocation byte
17ED C1         10000         POP     BC
17EE CD0318     10010         CALL    @DIRWR          ;Write FXDE back to disk
17F1 C0         10020         RET     NZ              ;Return if error
17F2 3AAC17     10030         LD      A,(STUFDEC+1)   ;  else p/u DEC of FPDE
17F5 47         10040         LD      B,A
17F6 CDBB18     10050         CALL    @DIRRD          ;Read its directory
17F9 C0         10060         RET     NZ              ;  & return if error
17FA 7D         10070         LD      A,L
17FB C61E       10080         ADD     A,1EH           ;Point to FXDE posn
17FD 6F         10090         LD      L,A             ;  in FPDE
17FE 36FE       10100         LD      (HL),0FEH       ;Show link to FXDE
1800 2C         10110         INC     L
1801 3600       10120 NHIT3   LD      (HL),0          ;Show what's the FXDE DEC
                10130                                 ;  & write the DIR back
                10140 ;
                10150 ;       Routine to write a directory sector
                10160 ;       B => DEC of FPDE, C => logical drive number
                10170 ;       HL <= points to directory record in SBUFF$
                10180 ;
```

File positioning subroutines

```
1803 CD0718    10190 @DIRWR   CALL    DIRWR             ;Permit two attempts
1806 C8        10200          RET     Z
1807 D5        10210 DIRWR    PUSH    DE                ;Save the reg
1808 CDCA18    10220          CALL    CALCDIR           ;Calc dir cyl
180B 2E00      10230          LD      L,0               ;Set buffer to start
180D CDEC19    10240          CALL    @WRSSC            ;Write the sector
1810 CCDC19    10250          CALL    Z,@VRSEC          ;Verify on no error
1813 D606      10260          SUB     6
1815 D1        10270          POP     DE
1816 C8        10280          RET     Z                 ;Back on system sector
1817 FE09      10290          CP      0FH-6             ;WP error?
1819 3E12      10300          LD      A,18              ;Set dir write error
181B C0        10310          RET     NZ                ;  if not WP
181C D603      10320          SUB     3
181E C9        10330          RET
               10340 ;
               10350 ;        Find a spare HIT entry
               10360 ;
181F F5        10370 NHIT4    PUSH    AF
1820 3E07      10380          LD      A,7               ;Get highest # sector
1822 CD2B1A    10390          CALL    @DCTBYT           ;  on a cylinder
1825 D5        10400          PUSH    DE                ;  into register E
1826 57        10410          LD      D,A
1827 E61F      10420          AND     1FH
1829 5F        10430          LD      E,A
182A 1C        10440          INC     E                 ;& get number of heads
182B AA        10450          XOR     D                 ;  into register A
182C 07        10460          RLCA
182D 07        10470          RLCA
182E 07        10480          RLCA
182F 3C        10490          INC     A
1830 CD0A19    10500          CALL    @MUL8             ;To calc sectors/cylinder
1833 CD3B19    10510          CALL    CKDBLBIT          ;Double if necessary
1836 D1        10520          POP     DE                ;Total sectors per cyl
1837 D602      10530          SUB     2                 ;Reduce for GAT & HIT
1839 324918    10540          LD      (NHIT7+1),A       ;# of directory sectors
183C F1        10550          POP     AF                ;Get DEC init entry
183D 6F        10560          LD      L,A
183E CD4518    10570          CALL    NHIT6             ;Ck if HIT slot is spare
1841 C8        10580          RET     Z                 ;Return if it is spare
1842 2E3F      10590          LD      L,3FH
1844 2C        10600 NHIT5    INC     L
1845 7D        10610 NHIT6    LD      A,L
1846 E61F      10620          AND     1FH
1848 FE00      10630 NHIT7    CP      0                 ;Does value exceed
184A 300D      10640          JR      NC,NHIT9          ;  sectors/cylinder?
184C 7E        10650          LD      A,(HL)
184D B7        10660          OR      A
184E C8        10670          RET     Z
184F 7D        10680 NHIT8    LD      A,L
1850 C620      10690          ADD     A,20H
1852 6F        10700          LD      L,A
1853 30F0      10710          JR      NC,NHIT6
1855 FE1F      10720          CP      1FH               ;Else go to next sector
1857 20EB      10730          JR      NZ,NHIT5          ;  column
1859 B7        10740 NHIT9    OR      A
185A C9        10750          RET
               10760 ;
               10770 ;        Test if gran is free in GAT
```

File positioning subroutines

```
                10780 ;
185B E607       10790 TSTBIT  AND     7               ;Get 0 to 7
185D 07         10800         RLCA                    ;Shift to match BIT n,
185E 07         10810         RLCA                    ;  opcode
185F 07         10820         RLCA
1860 F640       10830         OR      40H
1862 326618     10840         LD      (TBIT1+1),A     ;Modify BIT instruction
1865 CB40       10850 TBIT1   BIT     0,B
1867 C9         10860         RET
                10870 ;
                10880 ;       Set gran to allocated in GAT
                10890 ;
1868 07         10900 SETBIT  RLCA                    ;Shift to create opcode
1869 07         10910         RLCA                    ;  to match current bit
186A 07         10920         RLCA
186B F6C7       10930         OR      0C7H
186D 327218     10940         LD      (SBIT1+1),A     ;Create SET n, opcode
1870 AF         10950         XOR     A
1871 CBC7       10960 SBIT1   SET     0,A
1873 C9         10970         RET
                10980 ;
                10990 ;       Routine reads/writes the Granule Allocation Table
                11000 ;
1874 F6         11010 @GATRD  DB      0F6H            ;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 0
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      0F6H            ;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 FE06     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
18C0 2E00     11520         LD      L,0             ;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 E6E0     11660         AND     0E0H            ;  from the DEC
18D0 6F       11670         LD      L,A
18D1 261D     11680         LD      H,SBUFF$<-8     ;Point to buffer start
18D3 A8       11690         XOR     B               ;Calculate directory
18D4 C602     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 110100   11790         LD      DE,1            ;Pt to tk 0, sec 1
18E0 CDF419   11800         CALL    @RDSEC          ;Read to find dir cyl
18E3 D1       11810         POP     DE
18E4 C0       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 2609     11870         LD      H,9             ;Update memory table
18EB CD341A   11880         CALL    DCTFLD@
18EE 6F       11890         LD      L,A
18EF 72       11900         LD      (HL),D
18F0 E1       11910         POP     HL
18F1 CDF419   11920 READIR  CALL    @RDSEC          ;Retry dir read
18F4 D606     11930         SUB     6               ;Test protected
18F6 C9       11940         RET
              11950 ;
```

```
18F7 3E09     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 3E06     12010 MAXCYL  LD      A,6
1900 C5       12020         PUSH    BC
1901 DD4E06   12030         LD      C,(IX+6)
1904 CD2B1A   12040         CALL    @DCTBYT         ;Get highest # cyl
1907 3C       12050         INC     A               ;Adjust for zero offset
1908 C1       12060         POP     BC
1909 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 0608     12140         LD      B,8
190F 87       12150 MEA1    ADD     A,A
1910 CB23     12160         SLA     E
1912 3001     12170         JR      NC,MEA2
1914 82       12180         ADD     A,D
1915 10F8     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 07       12280         RLCA
191F 07       12290         RLCA
1920 07       12300         RLCA
1921 E607     12310         AND     7
1923 3C       12320         INC     A               ;Adj for 0 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 0608     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 3802     12440         JR      C,DEA2
1932 91       12450         SUB     C
1933 1C       12460         INC     E
1934 10F6     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
```

```
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           03600 *GET    LOADER:3
               12650 ;LOADER/ASM - LS-DOS 6.2
1948           12660 CORE$   DEFL    $
0100           12670         ORG     SVCTAB$
               12680 ;
               12690 ;       Supervisor Call table - Page 5
               12700 ;
0100 F21B      12710         DW      @IPL,@KEY,@DSP,@GET           ;0-3
     2806 4206 3806
0108 4506      12720         DW      @PUT,@CTL,@PRT,@WHERE         ;4-7
     2306 3D06 7919
0110 3506      12730         DW      @KBD,@KEYIN,@DSPLY,@LOGER     ;8-11
     8505 2D05 0305
0118 0005      12740         DW      @LOGOT,@MSG,@PRINT,@VDCTL     ;12-15
     3005 2805 990B
0120 8203      12750         DW      @PAUSE,@PARAM,@DATE,@TIME     ;16-19
     8719 A807 8D07
0128 8906      12760         DW      @CHNIO,@ABORT,@EXIT,SVCERR    ;20-23
     081B 0B1B F41A
0130 7E19      12770         DW      @CMNDI,@CMNDR,@ERROR,@DEBUG   ;24-27
     7B19 0F1B A019
0138 F51C      12780         DW      @CKTSK,@ADTSK,@RMTSK,@RPTSK   ;28-31
     DA1C D71C EB1C
0140 D01C      12790         DW      @KLTSK,@CKDRV,@DODIR,@RAMDIR  ;32-35
     9319 AF19 AC19
0148 F41A      12800         DW      SVCERR,SVCERR,SVCERR,SVCERR   ;36-39
     F41A F41A F41A
0150 B519      12810         DW      @DCSTAT,@SLCT,@DCINIT,@DCRES  ;40-43
     BC19 C019 C419
0158 C819      12820         DW      @RSTOR,@STEPI,@SEEK,@RSLCT    ;44-47
     CC19 D019 D419
0160 D819      12830         DW      @RDHDR,@RDSEC,@VRSEC,@RDTRK   ;48-51
     F419 DC19 E019
0168 E419      12840         DW      @HDFMT,@WRSEC,@WRSSC,@WRTRK   ;52-55
     E819 EC19 F019
0170 9619      12850         DW      @RENAME,@REMOVE,@INIT,@OPEN   ;56-59
     A619 8D19 8A19
0178 9919      12860         DW      @CLOSE,@BKSP,@CKEOF,@LOC      ;60-63
     8614 8F15 B314
0180 DE14      12870         DW      @LOF,@PEOF,@POSN,@READ        ;64-67
     A214 3414 1315
0188 9B14      12880         DW      @REW,@RREAD,@RWRIT,@SEEKSC    ;68-71
     7314 AD13 2114
0190 3014      12890         DW      @SKIP,@VER,@WEOF,@WRITE       ;72-75
     6015 EC14 3115
0198 381B      12900         DW      @LOAD,@RUN,@FSPEC,@FEXT       ;76-79
     1D1B 8119 8419
01A0 9C19      12910         DW      @FNAME,@GTDCT,@GTDCB,@GTMOD   ;80-83
     1E1A 9019 B219
01A8 F41A      12920         DW      SVCERR,@RDSSC,@GATRD,@DIRRD   ;84-87
     D818 7418 BB18
01B0 0318      12930         DW      @DIRWR,@GATWR,@MUL8,@MUL16    ;88-91
     7518 0A19 C906
01B8 F41A      12940         DW      SVCERR,@DIV8,@DIV16,SVCERR    ;92-95
     2719 E306 F41A
01C0 E103      12950         DW      @DECHEX,@HEXDEC,@HEX8,@HEX16  ;96-99
     F606 C207 BD07
01C8 4819      12960         DW      @HIGH$,@FLAGS,@BANK,@BREAK    ;100-103
     6A19 7708 6F19
01D0 9203      12970         DW      @SOUND,@CLS,@CKBRKC,SVCERR    ;104-107
```

System Loader and associated routines

```
     4505 5305 F41A
01D8 F41A      12980           DW      SVCERR,SVCERR,SVCERR,SVCERR     ;108-111
     F41A F41A F41A
01E0 F41A      12990           DW      SVCERR,SVCERR,SVCERR,SVCERR     ;112-115
     F41A F41A F41A
01E8 F41A      13000           DW      SVCERR,SVCERR,SVCERR,SVCERR     ;116-119
     F41A F41A F41A
01F0 F41A      13010           DW      SVCERR,SVCERR,SVCERR,SVCERR     ;120-123
     F41A F41A F41A
01F8 F41A      13020           DW      SVCERR,SVCERR,SVCERR,SVCERR     ;124-127
     F41A F41A F41A
1948           13030           ORG     CORE$
               13040 ;
               13050 ;         Routine to set or retrieve HIGH$/LOW$
               13060 ;
1948 7C        13070 @HIGH$    LD      A,H             ;Test if put or get
1949 B5        13080           OR      L
194A 2812      13090           JR      Z,GETHILO       ;Go if get
194C 3A6C00    13100           LD      A,(CFLAG$)      ;Is HIGH$ changeable?
194F 0F        13110           RRCA
1950 3E2B      13120           LD      A,43            ;Init SVC parm error
1952 D8        13130           RET     C               ;Back with NZ
1953 04        13140           INC     B               ;Test for HIGH$/LOW$
1954 05        13150           DEC     B
1955 200E      13160           JR      NZ,PUTLO        ;Go if LOW$
1957 220E04    13170           LD      (HIGH$),HL      ;Set new HIGH$
195A 2A0E04    13180 GETHI     LD      HL,(HIGH$)      ;P/u the value &
195D C9        13190           RET                     ;  ret with Z-flag
195E 04        13200 GETHILO   INC     B               ;Test for HIGH$/LOW$
195F 05        13210           DEC     B
1960 28F8      13220           JR      Z,GETHI
1962 2A1E00    13230           LD      HL,(LOW$)       ;P/u LOW$
1965 221E00    13240 PUTLO     LD      (LOW$),HL       ;Get LOW$
1968 AF        13250           XOR     A               ;Set Z-flag
1969 C9        13260           RET
               13270 ;
196A FD216A00  13280 @FLAGS    LD      IY,FLGTAB$
196E C9        13290           RET
               13300 ;
196F E5        13310 @BREAK    PUSH    HL              ;Save user vector
1970 2A881C    13320           LD      HL,(BRKVEC$)    ;P/u current vector
1973 E3        13330           EX      (SP),HL         ;Save current & get user
1974 22881C    13340           LD      (BRKVEC$),HL    ;Stuff new vector
1977 E1        13350           POP     HL              ;Recover old vector
1978 C9        13360           RET
               13370 ;
1979 E1        13380 @WHERE    POP     HL
197A E9        13390           JP      (HL)
               13400 ;
               13410 ;         Code for these SVCs is in system overlays
               13420 ;
197B 3EA3      13430 @CMNDR    LD      A,0A3H          ;Interpret command & RET
197D EF        13440           RST     40
197E 3EB3      13450 @CMNDI    LD      A,0B3H          ;Interpret a command
1980 EF        13460           RST     40
1981 3EC3      13470 @FSPEC    LD      A,0C3H          ;Parse a filespec
1983 EF        13480           RST     40
1984 3ED3      13490 @FEXT     LD      A,0D3H          ;Optional default EXT
1986 EF        13500           RST     40
```

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 180E      14100          JR      IOFUNC
19E8 3E0D      14110 @WRSEC   LD      A,13             ;Write standard sector
19EA 180A      14120          JR      IOFUNC
19EC 3E0E      14130 @WRSSC   LD      A,14             ;Write a system sector
19EE 1806      14140          JR      IOFUNC
19F0 3E0F      14150 @WRTRK   LD      A,15             ;Write a track
19F2 1802      14160          JR      IOFUNC
19F4 3E09      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 0
               14230 ;
19F8 C5        14240          PUSH    BC
19F9 AF        14250          XOR     A
19FA 47        14260          LD      B,A              ;Set bank function 0,
19FB 4F        14270          LD      C,A              ;  bank number 0
19FC CD7708    14280          CALL    @BANK            ;Bring up bank
19FF F1        14290          POP     AF               ;Perform EX (SP),BC
1A00 C5        14300          PUSH    BC
1A01 F5        14310          PUSH    AF
1A02 C1        14320          POP     BC
               14330 ;
               14340 ;        Continue disk I/O setup
               14350 ;
1A03 79        14360          LD      A,C              ;Xfer the drive code
1A04 322300    14370          LD      (LDRV$),A
1A07 FDE5      14380          PUSH    IY
1A09 CD1E1A    14390          CALL    @GTDCT           ;Get DCT address in IY
1A0C 3E20      14400          LD      A,20H            ;Set illegal drive #
1A0E B7        14410          OR      A                ;  if drive disabled
1A0F 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,102            ;Set for @BANK
1A18 EF        14500          RST     40               ;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 DD4E06    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 0
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$<-8       ;Point H to DCT hi-order
1A42 C9        15030         RET
               15040 ;
               15050 ;       Process supervisory calls <0-127>
               15060 ;
1A43 FE1A      15070 SVCUSER CP      26              ;Check for @ERROR
1A45 2808      15080         JR      Z,ERRSVC        ;Skip next if so
1A47 320D00    15090         LD      (LSVC$),A       ;Store SVC request
1A4A E3        15100         EX      (SP),HL         ;P/u RET address
1A4B 220B00    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 &
1A60 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),0C9H
1A6A F1       15350          POP     AF              ;Restore AF, HL
1A6B E1       15360          POP     HL
1A6C CD7F1A   15370 HKRES$   CALL    CKMOD@          ;Get overlay if needed
1A6F 3E00     15380          LD      A,0             ;P/u new overlay #
1A70          15390 OVRLYOLD         EQU     $-1
1A71 326900   15400          LD      (OVRLY$),A      ;  & update current
1A74 CD0000   15410 TRANSFR  CALL    0               ;Traadr of SYSx
1A77 F5       15420          PUSH    AF
1A78 3E00     15430          LD      A,0             ;Set to C9 if EXEC only
1A79          15440 SET@EXEC         EQU     $-1
1A7A 329F19   15450          LD      (@DBGHK),A
1A7D F1       15460          POP     AF
1A7E C9       15470          RET
              15480 ;
              15490 ;        DOS command overlay request
              15500 ;
1A7F E5       15510 CKMOD@   PUSH    HL
1A80 67       15520          LD      H,A             ;Save command value
1A81 78       15530          LD      A,B
1A82 32D21A   15540          LD      (EXOVR2+1),A    ;Set overlay #
1A85 7C       15550          LD      A,H
1A86 F601     15560          OR      1               ;Set for SYS6 & SYS7
1A88 FE89     15570          CP      89H             ;Is it either?
1A8A 7C       15580          LD      A,H             ;Get back the correct #
1A8B 2813     15590          JR      Z,EXOVR         ;Sys6/7 req? Use ISAM!
1A8D FE8A     15600          CP      8AH             ;Sys8 also ISAM
1A8F 280F     15610          JR      Z,EXOVR
1A91 3A6900   15620          LD      A,(OVRLY$)      ;P/u current overlay
1A94 AC       15630          XOR     H               ;Ck if it's the one
1A95 E60F     15640          AND     0FH             ;  we need to execute
1A97 7C       15650          LD      A,H
1A98 32701A   15660          LD      (OVRLYOLD),A    ;Update current tempy
1A9B 21001E   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 ;
1AA0 D5       15720 EXOVR    PUSH    DE
1AA1 C5       15730          PUSH    BC
1AA2 E60F     15740          AND     0FH             ;Get right nybble
1AA4 CB5F     15750          BIT     3,A             ;Check for SYS0-7
1AA6 2802     15760          JR      Z,EXOVR1        ;  w/o changing carry
1AA8 C618     15770          ADD     A,18H           ;Adjust for sys8-15
1AAA 329300   15780 EXOVR1   LD      (SFCB$+7),A
1AAD 47       15790          LD      B,A             ;Set DEC for directory
1AAE 3E20     15800          LD      A,20H           ;Set bit 5 of FCB+1
1AB0 328D00   15810          LD      (SFCB$+1),A
1AB3 ED62     15820          SBC     HL,HL           ;Carry is clear here
1AB5 229600   15830          LD      (SFCB$+10),HL   ;Zero NRN
1AB8 4C       15840          LD      C,H             ;Init for drive 0
1AB9 CDBB18   15850          CALL    @DIRRD          ;Read dir entry
1ABC 201A     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 3E07       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 119A00     15950           LD      DE,SFCB$+14     ;Extent field in FCB
1ACE CDE11A     15960           CALL    PAT1            ;Stuff 1st two extents
1AD1 0600       15970 EXOVR2    LD      B,0             ;P/u ISAM # or zero
1AD3 1E8C       15980           LD      E,SFCB$&0FFH
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)
1AF0 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 00         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$
1B02 CD0005     16340           CALL    @LOGOT          ;Log the error & ABORT
1B05 318003     16350           LD      SP,STACK$       ;reset stack
1B08 21FFFF     16360 @ABORT    LD      HL,-1
1B0B 3E93       16370 @EXIT     LD      A,93H           ;Exit to DOS
1B0D EF         16380           RST     40
                16390 ;
1B0E E1         16400 POPERR    POP     HL              ;Pop extended error
1B0F 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 20
1B19 78        16460 ERRNUM  DM      'xxH',CR
     78 48 0D
               16470 ;
               16480 ;       Routine to RUN a program
               16490 ;
1B1D E5        16500 @RUN    PUSH    HL              ;Save register pair
1B1E 217C00    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 20E5      16600         JR      NZ,POPERR
               16610 ;
               16620 ;       Place the INBUF$ pointer in register pair BC
               16630 ;
1B29 012004    16640         LD      BC,INBUF$       ;Reflect buffer pointer
               16650 ;
               16660 ;       Get TRAADR then test if we need to go to DEBUG
               16670 ;
1B2C 3A7C00    16680         LD      A,(SFLAG$)
1B2F CB4F      16690         BIT     1,A             ;Go to the program if
1B31 C0        16700         RET     NZ              ;  its EXEC only access
1B32 CB7F      16710         BIT     7,A             ;  else test if DEBUG
1B34 C23000    16720         JP      NZ,@RST30       ;  is on & go to it
1B37 C9        16730         RET                     ;  else go to program
               16740 ;
               16750 ;       This routine LOADs a Load Module Format file
               16760 ;
1B38 0600      16770 @LOAD   LD      B,0             ;LRL=256
1B3A 217C00    16780         LD      HL,SFLAG$
1B3D CBC6      16790         SET     0,(HL)          ;Don't set "file open"
1B3F 21001D    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 2600      16870         LD      H,0
1B4E F6C0      16880         OR      0C0H            ;Set RETurn & abbrev
1B50 FED8      16890         CP      0D8H            ;Change "file not in dir"
1B52 C0        16900         RET     NZ              ;  to "program not found"
1B53 C607      16910         ADD     A,7
1B55 C9        16920         RET
               16930 ;
               16940 ;       System command file loader
               16950 ;
1B56 78        16960 LOADER  LD      A,B             ;Set overlay # (0 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
1B60 CD6F1B    17010         CALL    LDR01           ;Do the load
1B63 DDE1      17020         POP     IX              ;Recover IX
```

System Loader and associated routines

```
1B65 C9         17030          RET
                17040 ;
                17050 ;        Routine to ignore the LMF record
                17060 ;
1B66 CDD61B     17070 LDR05    CALL    LDR15             ;Get length of "comment"
1B69 47         17080          LD      B,A
1B6A CDD61B     17090 LDR06    CALL    LDR15             ;Read & ignore that many
1B6D 10FB       17100          DJNZ    LDR06             ;  bytes then fall thru
                17110 ;
                17120 ;        Routine to parse LMF record types
                17130 ;
1B6F CDD61B     17140 LDR01    CALL    LDR15             ;Get record type
1B72 FE01       17150 LDR02    CP      1                 ;Start of block?
1B74 281F       17160          JR      Z,LDR08
1B76 FE02       17170          CP      2                 ;Start of TRAADR?
1B78 2814       17180 LDR03    JR      Z,LDR07
1B7A FE04       17190          CP      4                 ;End of LIB member?
1B7C 282A       17200          JR      Z,LDR12
1B7E FE08       17210          CP      8                 ;Begin ISAM table entry?
1B80 2828       17220          JR      Z,LDR13
1B82 FE0A       17230          CP      10                ;End of ISAM map?
1B84 2804       17240          JR      Z,LDR04
1B86 FE20       17250          CP      20H               ;Ignore all other control
1B88 38DC       17260          JR      C,LDR05
1B8A 3E22       17270 LDR04    LD      A,22H             ;Load file format err
1B8C B7         17280          OR      A
1B8D C9         17290          RET
                17300 ;
                17310 ;        Grab transfer address
                17320 ;
1B8E CDD61B     17330 LDR07    CALL    LDR15             ;Bypass 2nd X'02'
1B91 CDE81B     17340          CALL    GETADR            ;P/u transfer address
1B94 C9         17350          RET                       ;Ret Z or NZ
                17360 ;
                17370 ;        Grab load block
                17380 ;
1B95 CDD61B     17390 LDR08    CALL    LDR15             ;P/u block len
1B98 47         17400          LD      B,A
1B99 CDE81B     17410          CALL    GETADR            ;P/u load address
1B9C C0         17420          RET     NZ
1B9D 05         17430          DEC     B                 ;Adj length for adr
1B9E 05         17440          DEC     B
1B9F CDD61B     17450 LDR09    CALL    LDR15             ;P/u block byte
1BA2 77         17460          LD      (HL),A
1BA3 23         17470          INC     HL
1BA4 10F9       17480          DJNZ    LDR09             ;Loop until block end
1BA6 18C7       17490          JR      LDR01
                17500 ;
1BA8 E1         17510 LDR12    POP     HL
1BA9 C9         17520          RET
                17530 ;
                17540 ;        Routine to check ISAM table match
                17550 ;
1BAA CDD61B     17560 LDR13    CALL    LDR15             ;Get record length
1BAD 47         17570          LD      B,A
1BAE CDD61B     17580          CALL    LDR15             ;Get ISAM number
1BB1 05         17590          DEC     B                 ;  & decrement counter
1BB2 FE00       17600 LDR14    CP      0                 ;Either ISAM# or 0
1BB4 20B4       17610          JR      NZ,LDR06          ;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 2027     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 2015     17770           JR      NZ,LODERR
1BD1 CDDB1B   17780           CALL    LDR17            ;Read the sector
1BD4 189C     17790           JR      LDR02            ;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 2802     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
1BF0 BF       18040           CP      A
1BF1 C9       18050           RET
              18060 ;
              18070 ;         BOOT code brings back the ROM
              18080 ;
4300          18090 MOD3BUF
1BF2 21FB03   18100 @IPL      LD      HL,BOOTCOD       ;Code to toggle in ROM
1BF5 110043   18110           LD      DE,MOD3BUF       ;Buffer used by ROM
1BF8 D5       18120           PUSH    DE               ;This is return address
1BF9 010500   18130           LD      BC,BOOTLEN
1BFC EDB0     18140           LDIR                     ;Transfer boot code and
1BFE C9       18150           RET                      ;  jump to it
              18160 ;
              18170 ;         End of loader module
              18180 ;
1BFF          03610           SUBTTL  '<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
```

```
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 2600       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      $-1             ;Current task vector
                20020 ;
1CEB 210000     20030 @RPTSK   LD      HL,0            ;P/u last task done
1CEE 5E         20040          LD      E,(HL)          ;P/u task vector addr
1CEF 23         20050          INC     HL
1CF0 56         20060          LD      D,(HL)
1CF1 EB         20070          EX      DE,HL
1CF2 D1         20080          POP     DE              ;Pop ret addr
1CF3 18EE       20090          JR      CHGTASK
                20100 ;
                20110 ;        Routine to check if task slot active
                20120 ;
1CF5 07         20130 @CKTSK   RLCA                    ;Task number * 2
1CF6 C64F       20140          ADD     A,TCB$&0FFH+1   ;Index into task table
1CF8 6F         20150          LD      L,A
1CF9 2600       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
                03640          IFGT    $,1D00H+START$
                03650          ERR     'SYSRES memory overflow
                03660          ENDIF
1CFF            03670 CORE$    DEFL    $
1CFF 00         03680          DC      1D00H-CORE$,0
1CFF            03690          ORG     CORE$
1D00            03700          ORG     1D00H+START$
1D00            03710 SBUFF$   EQU     $
0100            03720          DS      256             ;Page disk I/O buffer
2300            03730 DIRBUF$  EQU     MAXCOR$-256     ;Another file buffer
                03740 ;
                03750 ;        get the system initialization module
                03760 ;
1E00            03770 OVERLAY  EQU     $
1E00            03780          SUBTTL  '<System initialization routines>'
```

```
1E00           03800 *GET    SYSINIT4:3
               20200 ;SYSINIT4/ASM - LS-DOS 6.2
               20210 ;
               20220 ;       This is the initialization part of SYSRES
               20230 ;
00F1           20240 TRKREG                          ;FDC track register
F401           20250 KB1                             ;Keyboard row 1
F460           20260 KB67                            ;Keyboard rows 6&7
F440           20270 KB7                             ;Keyboard row 7
001D           20280 BOL                             ;Beginning of line
               20290 ;
1E00           20300         ORG     1E00H+START$
               20310 ;
1E00 F3        20320         DI
1E01 21E90F    20330         LD      HL,@RSTNMI      ;Reset NMI vector to
1E04 226700    20340         LD      (@NMI+1),HL     ;  SYSRES's needs
1E07 211004    20350         LD      HL,PAKNAM$      ;Pt to pack name
1E0A 11BEF8    20360         LD      DE,2*80+CRTBGN$+30
1E0D 010800    20370         LD      BC,8
1E10 EDB0      20380         LDIR                    ;move pack name to crt
1E12 0E08      20390         LD      C,8             ;B contains 0 already
1E14 13        20400         INC     DE              ;Leave 2 spaces
1E15 13        20410         INC     DE
1E16 EDB0      20420         LDIR                    ;Move pack date to crt
               20430 ;
               20440 ;       Initialization routines
               20450 ;
1E18 AF        20460         XOR     A               ;Clear out stack area
1E19 218103    20470         LD      HL,STACK$+1     ;Stack start +1
1E1C 2D        20480 CLRLOOP DEC     L               ;Move down a byte
1E1D 77        20490         LD      (HL),A          ;Now loop and fill
1E1E 20FC      20500         JR      NZ,CLRLOOP      ;  and fill with 0's
               20510 ;
1E20 ED56      20520         IM      1
1E22 318003    20530         LD      SP,STACK$       ;Set the stack area
1E25 AF        20540         XOR     A
1E26 320202    20550         LD      (LBANK$),A      ;Set logical bank #
1E29 D3E4      20560                                 ;Disable INTRQ & DRQ
               20570 ;
1E2B 213802    20580         LD      HL,S1DCB$
1E2E 77        20590 ZERDCB  LD      (HL),A          ;Zero spare dcb area
1E2F 2C        20600         INC     L
1E30 20FC      20610         JR      NZ,ZERDCB
               20620 ;
1E32 3A7600    20630         LD      A,(MODOUT$)     ;Set hi-speed
1E35 D3EC      20640                                 ;  and external bus
1E37 3A8000    20650         LD      A,(WRINT$)
1E3A D3E0      20660                                 ;Enable RTC interrupts
1E3C 3A7800    20670         LD      A,(OPREG$)      ;Set memory configuration
1E3F 47        20680         LD      B,A
1E40 3EA7      20690         LD      A,0A7H          ;Value for AUX/RAM
1E42 0E84      20700         LD      C,@OPREG        ;Set the memory mgt port
1E44 ED41      20710                                 ;Bring up reg RAM
1E46 21FFFF    20720         LD      HL,-1           ;Ck for extended RAM
1E49 220E04    20730         LD      (HIGH$),HL
1E4C 221C00    20740         LD      (PHIGH$),HL
               20750 ;       Check the BANKS
1E4F 56        20760         LD      D,(HL)          ;Save what's in RAM
1E50 3655      20770         LD      (HL),55H        ;Stuff in reg RAM
1E52 ED79      20780                                 ;Switch in alt RAM
```

System initialization routines

```
1E54 5E         20790         LD      E,(HL)          ;Save the byte there
1E55 77         20800         LD      (HL),A          ;Stuff alt RAM
1E56 ED41       20810                                 ;Switch to reg RAM
1E58 BE         20820         CP      (HL)            ;See what's there now
1E59 72         20830         LD      (HL),D          ;Restore original value
1E5A ED79       20840                                 ;Back to alt RAM
1E5C 73         20850         LD      (HL),E          ;Restore original byte
1E5D ED41       20860                                 ;Back to reg RAM
1E5F 3EFE       20870         LD      A,0FEH          ;Init BAR$ for bank-0
1E61 2802       20880         JR      Z,$+4           ;Bypass if only 64K
1E63 3EF8       20890         LD      A,0F8H          ;Init BAR$ for bank 0-2
1E65 320102     20900         LD      (BAR$),A        ;Load Bank Avail RAM
1E68 320002     20910         LD      (BUR$),A        ;Load Bank Used RAM
1E6B 3A6F00     20920         LD      A,(FEMSK$)      ;P/u port FE mask
1E6E D3FE       20930                                 ;  & set it
1E70 00         20940         DC      3,0             ;Space for a JUMP
     00 00
                20950 ;
                20960 ;       Update DCT$ info for SYSTEM drive
                20970 ;
1E73 3A9D43     20980         LD      A,(BOOTST$)     ;P/u Boot Step rate
1E76 E603       20990         AND     3               ;Strip all but it
1E78 47         21000         LD      B,A             ;Save tempy
1E79 217304     21010         LD      HL,DCT$+3       ;Pt to DCT step
1E7C 7E         21020         LD      A,(HL)          ;P/u DCT Step
1E7D E6FC       21030         AND     0FCH            ;Strip step rate
1E7F B0         21040         OR      B               ;Merge in Boot step
1E80 77         21050         LD      (HL),A          ;Update DCT
1E81 DBF1       21060         IN      A,(TRKREG)      ;Update DCT with current
1E83 327504     21070         LD      (DCT$+5),A      ;  track posn of head
                21080 ;
1E86 110802     21090         LD      DE,KIDCB$       ;Flush type,init ptrs.
1E89 3E03       21100         LD      A,3
1E8B CD2306     21110         CALL    @CTL
1E8E FB         21120         EI                      ;Interrupts on
                21130 ;
                21140 ;       P/u CONFIG status & set ZERO byte
                21150 ;
1E8F 210104     21160         LD      HL,ZERO$
1E92 7E         21170         LD      A,(HL)          ;set to NOP if SYSGEN'd
1E93 3600       21180         LD      (HL),0          ;Make always zero byte
1E95 F5         21190         PUSH    AF              ;save SYSGEN flag
                21200 ;
                21210 ;       Check if date prompt is to be suppressed
                21220 ;
1E96 3AC204     21230         LD      A,(DTPMT$)      ;No prompt for date?
1E99 B7         21240         OR      A
                21250 ;
                21260 ;       Check on currency of date
                21270 ;
1E9A 213300     21280         LD      HL,DATE$        ;Point to Year
1E9D 4E         21290         LD      C,(HL)          ;  & save in reg C
1E9E 3600       21300         LD      (HL),0          ;  while resetting to zero
1EA0 23         21310         INC     HL              ;Bump to day
1EA1 46         21320         LD      B,(HL)          ;  & save in reg B
1EA2 3600       21330         LD      (HL),0          ;  while resetting to zero
1EA4 23         21340         INC     HL              ;Bump to Month
1EA5 7E         21350         LD      A,(HL)          ;  & save in Reg A
1EA6 3600       21360         LD      (HL),0          ;  while resetting to zero
```

```
1EA8 C2991F    21370          JP      NZ,TIMIN        ;Ck time if DATE=OFF
1EAB 2EFF      21380          LD      L,CFGFCB$+31&0FFH       ;Reset pointer
               21390 ;
               21400          IF      @INTL
               21410          LD      (HL),B          ;Stuff day
               21420          DEC     HL
               21430          LD      (HL),A          ;Stuff month
               21440          ELSE
1EAD 77        21450          LD      (HL),A          ;Stuff month
1EAE 2B        21460          DEC     HL
1EAF 70        21470          LD      (HL),B          ;Stuff day
               21480          ENDIF
               21490 ;
1EB0 2B        21500          DEC     HL
1EB1 71        21510          LD      (HL),C          ;Stuff Year
1EB2 EB        21520          EX      DE,HL           ;  & point DE to CFGFCB$+29
1EB3 3D        21530          DEC     A               ;Check for month range <1-12>
1EB4 FE0C      21540          CP      12              ;OK if 0-11 now
1EB6 380E      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 013008    21590          LD      BC,8<+8!'0'     ;Set buf len & char
1EC1 CDAE20    21600          CALL    GETPARM         ;Get response
1EC4 30F2      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 D650      21640          SUB     80              ;Reduce for range test
1ECA FE08      21650          CP      8
1ECC 30EA      21660          JR      NC,DATIN
1ECE E603      21670          AND     3
1ED0 3E1C      21680          LD      A,28            ;Init February
1ED2 2006      21690          JR      NZ,NOTLEAP
1ED4 213700    21700          LD      HL,DATE$+3+1    ;Set leap flag
1ED7 CBFE      21710          SET     7,(HL)
1ED9 3C        21720          INC     A               ;Feb to 29 days
1EDA 210304    21730 NOTLEAP  LD      HL,MAXDAY$+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
1EE0 1A        21820          LD      A,(DE)
1EE1 47        21830          LD      B,A             ;Save month in reg B
1EE2 3D        21840          DEC     A               ;Range check
1EE3 FE0C      21850          CP      12
1EE5 30D1      21860          JR      NC,DATIN        ;Go if 0 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 (0->FF)
1EED BE       21990         CP      (HL)
1EEE 30C8     22000         JR      NC,DATIN        ;Go if too large (or 0)
              22010 ;
              22020 ;       Range checks OK - move into DATE$
              22030 ;
1EF0 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 210304   22170         LD      HL,MAXDAY$+2    ;Init and adjust Feb
1F00 361C     22180         LD      (HL),28         ;  as required
1F02 2001     22190         JR      NZ,$+3
1F04 34       22200         INC     (HL)            ;Bump to 29
1F05 3A3500   22210         LD      A,(DATE$+2)     ;P/u month & xfer to B
1F08 47       22220         LD      B,A
1F09 3A3400   22230         LD      A,(DATE$+1)     ;P/u day of month
              22240 ;
              22250 ;       Compute day of year and day of week
              22260 ;
1F0C 6F       22270         LD      L,A             ;Start off with days
1F0D 2600     22280         LD      H,0             ;  in this month
1F0F 110104   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 10F7     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 0F       22500         RRCA
1F2C 0F       22510         RRCA
1F2D E603     22520         AND     3
1F2F 83       22530         ADD     A,E
1F30 5F       22540         LD      E,A             ;And add it in
```

```
1F31 1600     22550          LD      D,0               ;Add into HL
1F33 19       22560          ADD     HL,DE
1F34 23       22570          INC     HL                ;To start in right place
1F35 0E07     22580          LD      C,7               ;Now divide by 7 (B=0)
1F37 ED42     22590 DIV7     SBC     HL,BC             ;Subtract weeks (7-days)
1F39 30FC     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 07       22640          RLCA                      ;Shift to bits 1-3
1F40 4F       22650          LD      C,A               ;Save tempy
1F41 213700   22660          LD      HL,DATE$+3+1
1F44 7E       22670          LD      A,(HL)            ;Pack into field
1F45 E6F1     22680          AND     0F1H
1F47 B1       22690          OR      C
1F48 77       22700          LD      (HL),A
1F49 C5       22710          PUSH    BC
1F4A 211B15   22720          LD      HL,21<8!27        ;Set video row,col
1F4D 0603     22730          LD      B,3               ;Set function code 3
1F4F CD990B   22740          CALL    @VDCTL            ;  to position cursor
1F52 C1       22750          POP     BC
1F53 21C704   22760          LD      HL,DAYTBL$
1F56 CDFF20   22770          CALL    SPACE4            ;Write out the DAY
1F59 3E2C     22780          LD      A,','
1F5B CD4206   22790          CALL    @DSP
1F5E 3E20     22800          LD      A,' '
1F60 CD4206   22810          CALL    @DSP
1F63 3A3500   22820          LD      A,(DATE$+2)       ;P/u month number
1F66 47       22830          LD      B,A
1F67 2EDC     22840          LD      L,MONTBL$&0FFH    ;Reset HL for month table
1F69 CD0721   22850          CALL    DSPMDY            ;Write out the month name
1F6C 3E20     22860          LD      A,' '
1F6E CD4206   22870          CALL    @DSP
1F71 3A3400   22880          LD      A,(DATE$+1)       ;P/u day
1F74 05       22890          DEC     B                 ;From 0 to X'FF'
1F75 04       22900 DIV10    INC     B                 ;Divide by 10
1F76 D60A     22910          SUB     10                ;  with quotient in B
1F78 30FB     22920          JR      NC,DIV10
1F7A F5       22930          PUSH    AF                ;Save remainder (-10)
1F7B 78       22940          LD      A,B               ;P/u quotient
1F7C C630     22950          ADD     A,'0'             ;Change to ASCII
1F7E FE30     22960          CP      '0'               ;Zero?
1F80 C44206   22970          CALL    NZ,@DSP           ;Display if not
1F83 F1       22980          POP     AF                ;Get back remainder
1F84 C63A     22990          ADD     A,3AH             ;Change to ASCII
1F86 CD4206   23000          CALL    @DSP
1F89 211721   23010          LD      HL,PARTYR         ;Part of year
1F8C CD2D05   23020          CALL    @DSPLY
1F8F 3A3300   23030          LD      A,(DATE$)         ;Form last year digit
1F92 E607     23040          AND     7
1F94 C630     23050          ADD     A,'0'
1F96 CD4206   23060          CALL    @DSP              ;  and display it
              23070 ;
              23080 ;        Prompt for time
              23090 ;
1F99 3AC304   23100 TIMIN    LD      A,(TMPMT$)        ;Time to be prompted
1F9C B7       23110          OR      A
1F9D 2028     23120          JR      NZ,SELDCT         ;Skip if not
1F9F 211B16   23130 TIMIN0   LD      HL,22<8!27
```

System initialization routines

```
1FA2 113021     23140          LD      DE,TIMEPR         ;Set prompt message
1FA5 013008     23150          LD      BC,8<+8!'0'       ;Set len & separ char
1FA8 CDAE20     23160          CALL    GETPARM
1FAB 30F2       23170          JR      NC,TIMIN0         ;Loop on format error
1FAD 21FF00     23180          LD      HL,CFGFCB$+31
1FB0 3E17       23190          LD      A,23
1FB2 BE         23200          CP      (HL)              ;Test hour range
1FB3 38EA       23210          JR      C,TIMIN0
1FB5 2B         23220          DEC     HL
1FB6 3E3B       23230          LD      A,59
1FB8 BE         23240          CP      (HL)              ;Test minute range
1FB9 38E4       23250          JR      C,TIMIN0
1FBB 2B         23260          DEC     HL
1FBC BE         23270          CP      (HL)              ;Test the second range
1FBD 38E0       23280          JR      C,TIMIN0
1FBF 112D00     23290          LD      DE,TIME$          ;Move the time value
1FC2 010300     23300          LD      BC,3              ;  into the TIME$ field
1FC5 EDB0       23310          LDIR
                23320 ;
                23330 ;        Check on any AUTO command
                23340 ;
1FC7 212004     23350 SELDCT   LD      HL,INBUF$
1FCA 7E         23360          LD      A,(HL)            ;Pt to 1st byte of AUTO
1FCB FE2A       23370          CP      '*'               ;BREAK disable?
1FCD 200F       23380          JR      NZ,CKDCR
1FCF 23         23390          INC     HL
1FD0 3EE6       23400          LD      A,0E6H            ;Set BREAK bit in flag by
1FD2 325A20     23410          LD      (STUB1+1),A       ;  changing RES 4,(SFLAG$)
                23420                                    ;  to SET 4,(SFLAG$)
1FD5 181A       23430          JR      AUTO?
1FD7 CD1708     23440 GETKB17  CALL    ENADIS_DO_RAM
1FDA 3A41F4     23450          LD      A,(KB1!KB7)       ;scan row 1 & 7
1FDD C9         23460          RET
1FDE CDD71F     23470 CKDCR    CALL    GETKB17           ;Strobe keyboard
1FE1 CB67       23480          BIT     4,A               ;Is 'D' depressed?
1FE3 E5         23490          PUSH    HL                ;Save auto command pt
1FE4 21081B     23500          LD      HL,@ABORT         ;P/u abort address
1FE7 E3         23510          EX      (SP),HL           ;Swap them around
1FE8 C2A019     23520          JP      NZ,@DEBUG         ;DEBUG on <D>
1FEB D1         23530          POP     DE                ;Stack integrity
1FEC 2F         23540          CPL
1FED E601       23550          AND     1                 ;No AUTO if <ENTER>
1FEF 2803       23560          JR      Z,NOAUT1
1FF1 7E         23570 AUTO?    LD      A,(HL)            ;Any AUTO command?
1FF2 FE0D       23580          CP      CR                ;None if equal
1FF4 D1         23590 NOAUT1   POP     DE                ;Get back SYSGEN flag
1FF5 7A         23600          LD      A,D               ;  & move into reg A
1FF6 110B1B     23610          LD      DE,@EXIT          ;Where to go after boot
1FF9 010000     23620          LD      BC,0              ;Init BC(HL)=0 for @EXIT
1FFC 280F       23630          JR      Z,NOAUT           ;Go if no AUTO
1FFE E5         23640          PUSH    HL                ;Save buffer pointer
1FFF 218420     23650          LD      HL,CURSET         ;Point to cusor setting
2002 34         23660          INC     (HL)              ;Bump it down a line
2003 E1         23670          POP     HL                ;Recover INBUF$ pointer
2004 117E19     23680          LD      DE,@CMNDI         ;Lo order of @CMNDI
2007 D5         23690          PUSH    DE                ;Put on stack for RET
2008 44         23700          LD      B,H               ;Put INBUF$ pointer on
2009 4D         23710          LD      C,L               ;  stack for @CMNDI
200A 112D05     23720          LD      DE,@DSPLY         ;But do this first
```

```
200D D5        23730 NOAUT   PUSH    DE              ;Put on stack for RET
200E C5        23740         PUSH    BC              ;Either INBUF$ or 0
200F 215620    23750         LD      HL,STUB
2012 115043    23760         LD      DE,MOD3BUF+80   ;Must move out of way
2015 015800    23770         LD      BC,STUBLEN      ;  amount to move
2018 D5        23780         PUSH    DE              ;Add ret vector to stack
2019 EDB0      23790         LDIR                    ;Move stub up
201B CD4C20    23800         CALL    GETKB67
201E 117004    23810         LD      DE,DCT$         ;Set up to move DCT's
2021 210043    23820         LD      HL,MOD3BUF      ;  from configed area
2024 015000    23830         LD      BC,80           ;Count for DCTs (8*10)
2027 D9        23840         EXX                     ;Keep in alternate set
2028 E682      23850         AND     82H             ;Load config if zero
202A C0        23860         RET     NZ              ;No config > Go back
202B 210015    23870         LD      HL,21<8         ;Set to line 21
202E 0603      23880         LD      B,3             ;Position cursor
2030 CD990B    23890         CALL    @VDCTL
2033 213F20    23900         LD      HL,CONFIG$      ;Show sysgen message
2036 CD2D05    23910         CALL    @DSPLY
2039 11E000    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 **',03       ; Config DSP
     2A 20 53  59 53 47 45 4E
     20 2A 2A  03
               23960 ;
204C 2160F4    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 217C00    24060 STUB    LD      HL,SFLAG$
2059 CBA6      24070 STUB1   RES     4,(HL)          ;Test or SET Break bit
               24080                                 ;  without changing Z/NZ
205B 200C      24090         JR      NZ,NOTSG        ;Go if no SYSGEN found
205D 217600    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 EDB0      24140         LDIR                    ;Move 'em
2066 CD8600    24150         CALL    @ICNFG          ;Init config
               24160 NOTSG
2069 0E07      24170         LD      C,7
               24180 SETCYL0
206B CD1E1A    24190         CALL    @GTDCT
206E FDCB035E  24200         BIT     3,(IY+3)        ;If hard drive, don't stuff FF
2072 200B      24210         JR      NZ,NOFF         ;  & don't restore
2074 FD3605FF  24220         LD      (IY+5),0FFH     ;Set in case no restore
2078 3AC404    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 0D        24260 NOFF    DEC     C
2080 20E9      24270         JR      NZ,SETCYL0
2082 210015    24280         LD      HL,21<8         ;Set cursor
2084           24290 CURSET  EQU     $-1
```

System initialization routines

```
2085 0603     24300          LD      B,3
2087 CD990B   24310          CALL    @VDCTL
              24320 ;
              24330 ;        Detect Model 4 or 4P and adjust TFLAG$
              24340 ;        Look at 'MODEL' at 4018H. If so MOD-4P (5)
              24350 ;
              24360 ;
208A 114D4F   24370          LD      DE,'OM'
208D 2A1840   24380                                  ;P/u 4P rom leftover
2090 ED52     24390          SBC     HL,DE           ;Check if it's 'MO'
2092 3E04     24400          LD      A,4             ;Init for MOD 4 REG.
2094 2002     24410          JR      NZ,MOD4REG
2096 3E05     24420          LD      A,5             ;Change to MOD 4P
2098 327D00   24430 MOD4REG  LD      (TFLAG$),A
              24440 ;
209B 213800   24450          LD      HL,@RST38
209E 36C3     24460          LD      (HL),0C3H       ;Activate task processor
20A0 E1       24470          POP     HL              ;Pop INBUF$
20A1 C9       24480          RET                     ;To @CMD or @DSPLY,@CMNDI
20A2 00       24490          DC      12,0            ;Space for more code
     00 00 00 00 00 00 00 00
     00 00 00
20AE          24500 STUBEND  EQU     $
0058          24510 STUBLEN  EQU     STUBEND-STUB
              24520 ;
              24530 ;        Date & Time prompting
              24540 ;
20AE C5       24550 GETPARM  PUSH    BC              ;Save separator char
20AF D5       24560          PUSH    DE              ;Save message pointer
20B0 0603     24570          LD      B,3
20B2 CD990B   24580          CALL    @VDCTL          ;Position the cursor
20B5 E1       24590          POP     HL              ;Recover message pointer
20B6 CD2D05   24600          CALL    @DSPLY          ;  & display the message
20B9 21001E   24610          LD      HL,OVERLAY      ;Buffer for reply
20BC C1       24620          POP     BC
20BD C5       24630          PUSH    BC
20BE CD8505   24640          CALL    @KEYIN          ;Get reply & wait a bit
20C1 AF       24650          XOR     A               ;  disable test
20C2 B0       24660          OR      B
20C3 C1       24670          POP     BC              ;  of key prior to AUTO
20C4 C8       24680          RET     Z               ;Ret with NC if no entry
20C5 C5       24690          PUSH    BC
20C6 0640     24700          LD      B,40H
20C8 CD8203   24710          CALL    @PAUSE          ;  to let finger off
20CB C1       24720          POP     BC
              24730 ;
              24740 ;        Routine to parse DATE entry
              24750 ;
20CC 11FF00   24760 PARSDAT  LD      DE,CFGFCB$+31   ;Point to buf end
20CF 0603     24770          LD      B,3             ;Process 3 fields
20D1 D5       24780 PRSD1    PUSH    DE              ;Save pointer
              24790 ;
              24800 ;        Routine to parse a digit pair
              24810 ;
20D2 CDF820   24820          CALL    PRSD3           ;Get a digit
20D5 300F     24830          JR      NC,PRSD2        ;Jump if bad digit
20D7 5F       24840          LD      E,A             ;Multiply by ten
20D8 07       24850          RLCA
20D9 07       24860          RLCA
```

System initialization routines

```
20DA 83         24870           ADD     A,E
20DB 07         24880           RLCA
20DC 5F         24890           LD      E,A
20DD CDF820     24900           CALL    PRSD3           ;Get another digit
20E0 3004       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 D0         24970           RET     NC              ;Ret if bad digit pair
20E8 12         24980           LD      (DE),A          ;Else stuff the value
20E9 05         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 3006       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 D630       25120           SUB     30H
20FC FE0A       25130 PRSD4     CP      10
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 CD2D05     25200           CALL    @DSPLY
2106 E1         25210           POP     HL
2107 05         25220 DSPMDY    DEC     B               ;Point to Bth entry
2108 7D         25230           LD      A,L             ;  in table
2109 80         25240           ADD     A,B
210A 80         25250           ADD     A,B
210B 80         25260           ADD     A,B
210C 6F         25270           LD      L,A
210D 0603       25280           LD      B,3             ;Print 3 characters
210F 7E         25290 DSPM1     LD      A,(HL)
2110 23         25300           INC     HL
2111 CD4206     25310           CALL    @DSP
2114 10F9       25320           DJNZ    DSPM1
2116 C9         25330           RET
2117 2C         25340 PARTYR    DB      ', 198',30,3
     20 31 39   38 1E 03
                25350 ;
                25360           IF      @INTL
                25370 DATEPR    DB      30,'Date DD/MM/YY ? ',3
                25380           ELSE
211E 1E         25390 DATEPR    DB      30,'Date MM/DD/YY ? ',3
     44 61 74   65 20 4D 4D 2F
     44 44 2F   59 59 20 3F 20
     03
                25400           ENDIF
                25410 ;
```

System initialization routines

```
2130 1E       25420 TIMEPR  DB      30,'Time HH:MM:SS ? ',3
     54 69 6D 65 20 48 48 3A
     4D 4D 3A 53 53 20 3F 20
     03
2142 20       25430 SPACE4$ DB      '   ',03,03         ;3 or 4 space string
     20 20 03 03
2147 00       25440         DC      32,00               ;Space for message, or??
     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
2167          03810         SUBTTL  '<Misc. lowcore routines>'
```

Misc. lowcore routines

```
2167            03830 *GET    SOUND:3
                25450 ;SOUND/ASM - LS-DOS 6.2
                25460 ;
                25470 ;       Contains IPL, PAUSE, SOUND, and DECHEX routines
                25480 ;       Will be loaded into lowcore area along with SYSRES
                25490 ;
                25500 *MOD
0090            25510 SNDPORT
0380            25520         ORG     STACK$
0380 0000       25530         DW      00              ;Stack gaurd
                25540 ;
                25550 ;       Pause routine
                25560 ;
0382 C5         25570 @PAUSE  PUSH    BC              ;Save the count
                25580 ;       SRL     B               ;Adjust for WAIT STATES
                25590 ;       RR      C
0383 3A7C00     25600         LD      A,(SFLAG$)      ;If system (FAST)
0386 CB5F       25610         BIT     3,A             ;  then double it
0388 C48C03     25620         CALL    NZ,CDLOOP       ;Call if fast
038B C1         25630         POP     BC              ;Restore the count
038C 0B         25640 CDLOOP  DEC     BC              ;Count down routine
038D 78         25650         LD      A,B
038E B1         25660         OR      C
038F 20FB       25670         JR      NZ,CDLOOP
0391 C9         25680         RET
                25690 ;
                25700 ;       @SOUND SVC-104 - Operates sound generator
                25710 ;       B => sound function
                25720 ;       Bits 0-2 <0-7> = note # (0 highest)
                25730 ;       Bits 3-7 <0-31> = relative sound duration
                25740 ;       All regs except A left unchanged
                25750 ;       Z-flag set on exit
                25760 ;       Note that interrupts disabled during duration
                25770 ;
0392 C5         25780 @SOUND  PUSH    BC              ;Save registers
0393 E5         25790         PUSH    HL
0394 78         25800         LD      A,B             ;P/u sound data
0395 E607       25810         AND     7               ;  & strip off duration
0397 07         25820         RLCA                    ;Adj for 2-byte fields
0398 21D103     25830         LD      HL,SNDTAB
039B 4F         25840         LD      C,A
039C 78         25850         LD      A,B             ;Pick up duration data
039D 0600       25860         LD      B,0             ;Index into tone table
039F 09         25870         ADD     HL,BC           ;  to get note-on/off
03A0 4E         25880         LD      C,(HL)          ;P/u note-on/off data
03A1 23         25890         INC     HL
03A2 6E         25900         LD      L,(HL)          ;P/u note duration
03A3 0F         25910         RRCA                    ;Rotate sound duration
03A4 0F         25920         RRCA                    ;  into bits 0-4
03A5 0F         25930         RRCA
03A6 E61F       25940         AND     1FH             ;Strip off sound #
03A8 3C         25950         INC     A               ;Adjust for offset
03A9 67         25960         LD      H,A             ;Set sound counter
03AA 3A7C00     25970         LD      A,(SFLAG$)      ;If fast, double values
03AD E608       25980         AND     8H
03AF 2806       25990         JR      Z,$A1
03B1 CB24       26000         SLA     H
03B3 CB25       26010         SLA     L
03B5 CB21       26020         SLA     C
03B7 F3         26030 $A1     DI                      ;Can't interrupt timing
```

```
03B8 E5         26040 $A2     PUSH    HL              ;Save note duration
03B9 41         26050 $A3     LD      B,C             ;Play tone
03BA 3E01       26060         LD      A,1             ;Hold output high
03BC D390       26070         OUT     (SNDPORT),A     ;  for count of (B)
03BE 10FE       26080         DJNZ    $
03C0 41         26090         LD      B,C             ;Hold output low for
03C1 3C         26100         INC     A               ;  for count of (B)
03C2 D390       26110         OUT     (SNDPORT),A
03C4 10FE       26120         DJNZ    $
03C6 2D         26130         DEC     L               ;Dec the duration
03C7 20F0       26140         JR      NZ,$A3
03C9 E1         26150         POP     HL              ;Get sound/note durations
03CA 25         26160         DEC     H               ;Count down the sound
03CB 20EB       26170         JR      NZ,$A2          ;  duration counter
03CD FB         26180         EI                      ;Restore interrupts
03CE E1         26190         POP     HL
03CF C1         26200         POP     BC
03D0 C9         26210         RET
                26220 ;
                26230 ;       Note table
                26240 ;
00B4            26250 SNDOFF  EQU     180             ;Sound duration offset
001C            26260 TONER   EQU     28
03D1 50         26270 SNDTAB  DB      108-TONER       ;Note 0 (highest)
03D2 4C         26280         DB      0-SNDOFF
03D3 56         26290         DB      114-TONER
03D4 48         26300         DB      252-SNDOFF
03D5 5C         26310         DB      120-TONER
03D6 44         26320         DB      248-SNDOFF
03D7 62         26330         DB      126-TONER
03D8 40         26340         DB      244-SNDOFF
03D9 6B         26350         DB      135-TONER
03DA 3C         26360         DB      240-SNDOFF
03DB 72         26370         DB      142-TONER
03DC 38         26380         DB      236-SNDOFF
03DD 79         26390         DB      149-TONER
03DE 34         26400         DB      232-SNDOFF
03DF 80         26410         DB      156-TONER       ;Note 7 (lowest)
03E0 30         26420         DB      228-SNDOFF
004F            26430 SNDLEN  EQU     $-@SOUND
                26440 ;
                26450 ;       Process decimal assignment
                26460 ;
03E1 010000     26470 @DECHEX LD      BC,0            ;Init value to zero
03E4 7E         26480 DEC1    LD      A,(HL)          ;P/u a char
03E5 D630       26490         SUB     30H             ;Cvrt to binary
03E7 D8         26500         RET     C               ;Return if < "0"
03E8 FE0A       26510         CP      10              ;Ck for bad decimal
03EA D0         26520         RET     NC              ;Ret if not 0-9
03EB C5         26530         PUSH    BC              ;Exchange BC & HL
03EC E3         26540         EX      (SP),HL         ;  & save HL on stack
03ED 29         26550         ADD     HL,HL           ;Multiply by 10
03EE 29         26560         ADD     HL,HL
03EF 09         26570         ADD     HL,BC
03F0 29         26580         ADD     HL,HL
03F1 0600       26590         LD      B,0             ;Merge in new digit
03F3 4F         26600         LD      C,A             ;New digit to C
03F4 09         26610         ADD     HL,BC           ;  & add it in
03F5 44         26620         LD      B,H             ;Current value to BC
```

Misc. lowcore routines

```
03F6 4D         26630         LD      C,L
03F7 E1         26640         POP     HL              ;Recover HL pointer
03F8 23         26650         INC     HL
03F9 18E9       26660         JR      DEC1            ;Loop
                26670 ;
                26680 ;       Special Boot code to be moved to 4300h by IPL
                26690 ;
03FB F3         26700 BOOTCOD DI                      ;Boot stub for @IPL to
03FC AF         26710         XOR     A               ;  to move to 4300h
03FD D384       26720         OUT     (@OPREG),A
03FF C7         26730         RST     0
0005            26740 BOOTLEN EQU     $-BOOTCOD
                26750 ;
0400            03840         SUBTTL  '<Sign-on LOGO display>'
0400            03850 *GET    LOGO:3
                26760 ;RSLOGOB/ASM  3-D RS LOGO used on 6.2.0 - 1/20/84
                26770 *LIST   OFF
                27540 *LIST   ON
                03860 ;
1E00            03870         END     OVERLAY
```

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

LDOS60   @$SYS          08F0 00040
$MAIN    +@@1           0000 03870
LDOS60   +@@1           0000 00050
$MAIN    +@@2           0000 03870
LDOS60   +@@2           0000 00060
$MAIN    +@@3           0000 03870
LDOS60   +@@3           0000 00070
$MAIN    +@@4           0000 03870
LDOS60   +@@4           0000 00080
LOADER   @ABORT         1B08 16360 LOADER   12760
                                   SYSINIT4 23500
TASKER   @ADTSK         1CDA 19880 LOADER   12780
LDOS60   @BANK          0877 00090 LOADER   12960 14280
FILPOSN  @BKSP          1486 03560 LOADER   12860
LOADER   @BREAK         196F 13310 LOADER   12960
LDOS60   @BYTEIO        1300 00100 $MAIN    03290
LDOS60   @CHNIO         0689 00110 LOADER   12760
LDOS60   @CKBRKC        0553 00120 LOADER   12970
LOADER   @CKDRV         1993 13590 LOADER   12790
FILPOSN  @CKEOF         158F 05460 LOADER   12860
TASKER   @CKTSK         1CF5 20130 LOADER   12780
LOADER   @CLOSE         1999 13630 LOADER   12860
LDOS60   @CLS           0545 00130 LOADER   12970
LOADER   @CMNDI         197E 13450 LOADER   12770
                                   SYSINIT4 23680
LOADER   @CMNDR         197B 13430 LOADER   12770
LDOS60   @CTL           0623 00140 LOADER   12720
                                   SYSINIT4 21110
LDOS60   @DATE          07A8 00150 LOADER   12750
LOADER   @DBGHK         199F 13670 LOADER   15310 15450
                                   TASKER   19190
LOADER   @DCINIT        19C0 13910 LOADER   12810
LOADER   @DCRES         19C4 13930 LOADER   12810
LOADER   @DCSTAT        19B5 13850 LOADER   12810
LOADER   @DCTBYT        1A2B 14780 FILPOSN  05900 10390 11970 12040 12260 12590
LOADER   @DEBUG         19A0 13680 $MAIN    00570
                                   LOADER   12770
                                   SYSINIT4 23520
SOUND    @DECHEX        03E1 26470 LOADER   12950
FILPOSN  @DIRCYL        18F7 11960 FILPOSN  11060 11290 11640
FILPOSN  @DIRRD         18BB 11490 FILPOSN  04370 07690 09810 10050
                                   LOADER   12920 15850
FILPOSN  @DIRWR         1803 10190 FILPOSN  04510 09610 10010
                                   LOADER   12930
LDOS60   @DIV16         06E3 00160 FILPOSN  04010 04070 05930
                                   LOADER   12940
FILPOSN  @DIV8          1927 12370 LOADER   12940
LOADER   @DODIR         19AF 13780 LOADER   12790
LOADER   @DOKEY         19A9 13740
LDOS60   @DSP           0642 00170 LOADER   12710
                                   SYSINIT4 22790 22810 22870 22970 23000 23060
                                   SYSINIT4 25310
LDOS60   @DSPLY         052D 00180 LOADER   12730
                                   SYSINIT4 23020 23720 23910 24600 25200
LOADER   @ERROR         1B0F 16410 LOADER   12770
LOADER   @EXIT          1B0B 16370 LOADER   12760
                                   SYSINIT4 23610
LOADER   @FEXT          1984 13490 LOADER   12900
LOADER   @FLAGS         196A 13280 LOADER   12960
```

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

LOADER   @FNAME         199C 13650 LOADER    12910
LDOS60   @FRENCH        0000 00190 $MAIN     01480 03460
LOADER   @FSPEC         1981 13470 LOADER    12900
FILPOSN  @GATRD         1874 11010 FILPOSN   08230
                                   LOADER    12920
FILPOSN  @GATWR         1875 11020 FILPOSN   09580
                                   LOADER    12930
LDOS60   @GERMAN        0000 00200 $MAIN     01510 03430
LDOS60   @GET           0638 00210 LOADER    12710
LOADER   @GTDCB         1990 13570 LOADER    12910
LOADER   @GTDCT         1A1E 14580 LOADER    12910 14390
                                   SYSINIT4  24190
LOADER   @GTMOD         19B2 13800 LOADER    12910
LOADER   @HDFMT         19E4 14090 LOADER    12840
LDOS60   @HEX16         07BD 00220 LOADER    12950
LDOS60   @HEX8          07C2 00230 LOADER    12950 16320
LDOS60   @HEXDEC        06F6 00240 LOADER    12950
LOADER   @HIGH$         1948 13070 LOADER    12960
FILPOSN  @HITRD         1897 11240 FILPOSN   09660
FILPOSN  @HITWR         1898 11250 FILPOSN   09800
LDOS60   @HZ50          0000 00250
$MAIN    @ICNFG         0086 02870 SYSINIT4  24150
LOADER   @INIT          198D 13550 LOADER    12850
LDOS60   @INTL          0000 00260 $MAIN     01890
                                   SYSINIT4  21400 21760 21910 25360
LOADER   @IPL           1BF2 18100 LOADER    12710
LDOS60   @JCL           0630 00270
LDOS60   @KBD           0635 00280 LOADER    12730
LDOS60   @KEY           0628 00290 LOADER    12710
LDOS60   @KEYIN         0585 00300 LOADER    12730
                                   SYSINIT4  24640
$MAIN    @KITSK         0089 02920
LDOS60   @KITSK         0089 00310
TASKER   @KLTSK         1CD0 19810 LOADER    12790
LOADER   @LOAD          1B38 16770 LOADER    12900 16530
                                   SYSINIT4  23930
FILPOSN  @LOC           14B3 03870 FILPOSN   03050
                                   LOADER    12860
FILPOSN  @LOF           14DE 04250 LOADER    12870
LDOS60   @LOGER         0503 00320 LOADER    12730
LDOS60   @LOGOT         0500 00330 LOADER    12740 16340
LDOS60   @MOD2          0000 00340
LDOS60   @MOD4          FFFF 00350 $MAIN     02490
                                   FILPOSN   04170
LDOS60   @MSG           0530 00360 LOADER    12740
LDOS60   @MUL16         06C9 00370 FILPOSN   03180
                                   LOADER    12930
FILPOSN  @MUL8          190A 12110 FILPOSN   06960 10500
                                   LOADER    12930
$MAIN    @NMI           0066 00850 SYSINIT4  20340
LOADER   @OPEN          198A 13530 LOADER    12850 16810
LDOS60   @OPREG         0084 00380 TASKER    18500 18700
                                   SYSINIT4  20700
                                   SOUND     26720
LOADER   @PARAM         1987 13510 LOADER    12750
SOUND    @PAUSE         0382 25570 LOADER    12750
                                   SYSINIT4  24710
FILPOSN  @PEOF          14A2 03770 LOADER    12870
FILPOSN  @POSN          1434 03100 LOADER    12870 17740
```

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

LDOS60    @PRINT           0528 00390 LOADER   12740
LDOS60    @PRT             063D 00400 LOADER   12720
LDOS60    @PUT             0645 00410 LOADER   12720
LOADER    @RAMDIR          19AC 13760 LOADER   12790
LOADER    @RDHDR           19D8 14030 LOADER   12830
LOADER    @RDSEC           19F4 14170 FILPOSN  02130 11800 11920
                                      LOADER   12830
FILPOSN   @RDSSC           18D8 11760 FILPOSN  11110 11340 11530
                                      LOADER   12920
LOADER    @RDTRK           19E0 14070 LOADER   12830
FILPOSN   @READ            1513 04550 LOADER   12870
LOADER    @REMOVE          19A6 13720 LOADER   12850
LOADER    @RENAME          1996 13610 LOADER   12850
FILPOSN   @REW             149B 03700 LOADER   12880
TASKER    @RMTSK           1CD7 19860 LOADER   12780
TASKER    @RPTSK           1CEB 20030 LOADER   12780
                                      TASKER   19680 19820
FILPOSN   @RREAD           1473 03400 LOADER   12880
LOADER    @RSLCT           19D4 14010 LOADER   12820
$MAIN     @RST00           0000 00300
$MAIN     @RST08           0008 00340
$MAIN     @RST10           0010 00400
$MAIN     @RST18           0018 00430
$MAIN     @RST20           0020 00480
$MAIN     @RST28           0028 00530
$MAIN     @RST30           0030 00570 LOADER   16720
$MAIN     @RST38           0038 00590 SYSINIT4 24450
LDOS60    @RSTNMI          0FE9 00420 SYSINIT4 20330
LOADER    @RSTOR           19C8 13950 LOADER   12820
                                      SYSINIT4 24250
LDOS60    @RSTREG          0680 00430
LOADER    @RUN             1B1D 16500 LOADER   12900
FILPOSN   @RWRIT           13AD 02300 LOADER   12880
LOADER    @SEEK            19D0 13990 FILPOSN  02990
                                      LOADER   12820
FILPOSN   @SEEKSC          1421 02950 LOADER   12880
FILPOSN   @SKIP            1430 03050 LOADER   12890
LOADER    @SLCT            19BC 13890 LOADER   12810
SOUND     @SOUND           0392 25780 LOADER   12970
                                      SOUND    26430
LOADER    @STEPI           19CC 13970 LOADER   12820
LDOS60    @TIME            078D 00440 LOADER   12750
LDOS60    @USA             FFFF 00450 $MAIN    01540 03400
LDOS60    @VDCTL           0B99 00460 LOADER   12740
                                      SYSINIT4 22740 23890 24310 24580
LDOS60    @VDCTL3          0D38 00470
FILPOSN   @VER             1560 05080 LOADER   12890
LOADER    @VRSEC           19DC 14050 FILPOSN  02630 10250 11150 11380
                                      LOADER   12830
FILPOSN   @WEOF            14EC 04330 LOADER   12890
LOADER    @WHERE           1979 13380 LOADER   12720
FILPOSN   @WRITE           1531 04770 LOADER   12890
LOADER    @WRSEC           19E8 14110 FILPOSN  02590
                                      LOADER   12840
LOADER    @WRSSC           19EC 14130 FILPOSN  10240 11140 11370
                                      LOADER   12840
LOADER    @WRTRK           19F0 14150 LOADER   12840
LDOS60    @_VDCTL          0D42 00480
TASKER    ACTVTSK          1C49 18830 TASKER   18610
```

```
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$           0470 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 00660
LDOS60   DO_INVERT_DIS   0C8C 00670
LDOS60   DO_INVERT_ENA   0C89 00680
LDOS60   DO_INVERT_OFF   0C9B 00690
LDOS60   DO_MASK         0000 00700
LDOS60   DO_RET          0BCB 00710
LDOS60   DO_RET1         0BCC 00720
LDOS60   DO_SCROLL       0CCE 00730
LDOS60   DO_TABS         0BEA 00740
LDOS60   DSKTYP$         04C0 00750
SYSINIT4 DSPM1           210F 25290 SYSINIT4 25320
SYSINIT4 DSPMDY          2107 25220 SYSINIT4 22850
LDOS60   DTPMT$          04C2 00760 SYSINIT4 21230
LDOS60   DVREND$         0FF4 00770
LDOS60   DVRHI$          0206 00780
$MAIN    EFLAG$          006E 01330
LDOS60   ENADIS_DO_RAM   0817 00790 SYSINIT4 23440 23990
LOADER   ERRNUM          1B19 16460 LOADER   16310
LOADER   ERRSVC          1A4F 15130 LOADER   15080
LOADER   EXERR           1AD8 16000 LOADER   15860 15910
TASKER   EXITBRK         1C8E 19360 TASKER   19230
LOADER   EXOVR           1AA0 15720 LOADER   15590 15610
LOADER   EXOVR1          1AAA 15780 LOADER   15760
LOADER   EXOVR2          1AD1 15970 LOADER   15540
LOADER   EXOVR3          1ADA 16020 LOADER   15680
LOADER   EXTDBG$         19A4 13710
LDOS60   FDDINT$         000E 00800
$MAIN    FDDINT$         000E 00380
$MAIN    FEMSK$          006F 01340 SYSINIT4 20920
LDOS60   FLGTAB$         006A 00810 LOADER   13280
$MAIN    FLGTAB$         006A 00940
FILPOSN  GATRW1          188A 11140 FILPOSN  11100
FILPOSN  GATRW2          1894 11180 FILPOSN  11130
LOADER   GETADR          1BE8 18000 LOADER   17340 17410 17620 17640
LOADER   GETHI           195A 13180 LOADER   13220
LOADER   GETHILO         195E 13200 LOADER   13090
SYSINIT4 GETKB17         1FD7 23440 SYSINIT4 23470
SYSINIT4 GETKB67         204C 23970 SYSINIT4 23800
FILPOSN  GETNRN          140C 02810 FILPOSN  02310 03240 03430 03880 05470 05890
SYSINIT4 GETPARM         20AE 24550 SYSINIT4 21600 23160
LDOS60   GET_@_ROWCOL    0DAE 00820
LOADER   GODOIO          1A1C 14560 LOADER   14420
TASKER   GOTBRK          1C6A 19160 TASKER   19070
FILPOSN  GREC1           15E7 06020 FILPOSN  06230
FILPOSN  GREC2           15F8 06150 FILPOSN  06060 06280 06360
FILPOSN  GREC3           1603 06240 FILPOSN  06120
FILPOSN  GREC4           1616 06420 FILPOSN  06180
LDOS60   HERTZ$          0750 00830
LDOS60   HIGH$           040E 00840 LOADER   13170 13180
                                    TASKER   19280
                                    SYSINIT4 20730
FILPOSN  HITRW1          18AE 11370 FILPOSN  11330
FILPOSN  HITRW2          18B8 11410 FILPOSN  11360
LOADER   HKRES$          1A6C 15370
LDOS60   IFLAG$          0072 00850
$MAIN    IFLAG$          0072 01470
LDOS60   INBUF$          0420 00860 LOADER   16640
                                    SYSINIT4 23350
$MAIN    INTIM$          003C 00640 TASKER   18540
```

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

TASKER   INTLAT         00E0 18510 TASKER   18520
$MAIN    INTMSK$        003D 00700
LDOS60   INTVC$         003E 00870
$MAIN    INTVC$         003E 00750
LOADER   IOFUNC         19F6 14190 LOADER   13860 13900 13920 13940 13960 13980
                                   LOADER   14000 14020 14040 14060 14080 14100
                                   LOADER   14120 14140 14160
FILPOSN  IOREC          15CB 05890 FILPOSN  02080 02530 02970
FILPOSN  IORETZ         136A 01930 FILPOSN  01460
LDOS60   JCLCB$         0203 00880
$MAIN    JDCB$          0024 00510 FILPOSN  05170
$MAIN    JFCB$          00C0 03080
LDOS60   JLDCB$         0230 00890
$MAIN    JRET$          0026 00520 FILPOSN  05160
SYSINIT4 KB1            F401 20250 SYSINIT4 23450
SYSINIT4 KB67           F460 20260 SYSINIT4 23970
SYSINIT4 KB7            F440 20270 SYSINIT4 23450
LDOS60   KCK@           07D6 00900 TASKER   18660
$MAIN    KFLAG$         0074 01690
LDOS60   KFLAG$         0074 00910
LDOS60   KIDATA$        08FC 00920
LDOS60   KIDCB$         0208 00930 SYSINIT4 21090
LDOS60   LBANK$         0202 00940 TASKER   18400 18720
                                   SYSINIT4 20550
LOADER   LDR01          1B6F 17140 LOADER   17010 17490
LOADER   LDR02          1B72 17150 LOADER   17790
LOADER   LDR03          1B78 17180
LOADER   LDR04          1B8A 17270 LOADER   17240
LOADER   LDR05          1B66 17070 LOADER   17260
LOADER   LDR06          1B6A 17090 LOADER   17100 17610
LOADER   LDR07          1B8E 17330 LOADER   17180
LOADER   LDR08          1B95 17390 LOADER   17160
LOADER   LDR09          1B9F 17450 LOADER   17480
LOADER   LDR12          1BA8 17510 LOADER   17200
LOADER   LDR13          1BAA 17560 LOADER   17220
LOADER   LDR14          1BB2 17600 LOADER   16970
LOADER   LDR15          1BD6 17830 LOADER   17070 17090 17140 17330 17390 17450
                                   LOADER   17560 17580 17660 18000 18020
LOADER   LDR16          1BD9 17850 LOADER   17940
LOADER   LDR17          1BDB 17870 LOADER   17780 17840
$MAIN    LDRV$          0023 00500 LOADER   13870 14370
$MAIN    LF             000A 00120
$MAIN    LFLAG$         0075 01770
FILPOSN  LNKFCB@        1566 05110
LOADER   LOADER         1B56 16960 LOADER   15990 16830
FILPOSN  LOC1           14BC 03900 FILPOSN  04290
FILPOSN  LOC2           14C8 04010 FILPOSN  03960
FILPOSN  LOC3           14D9 04130 FILPOSN  03930
LOADER   LODERR         1BE6 17950 LOADER   17650 17770
$MAIN    LOW$           001E 00470 LOADER   13230 13240
$MAIN    LSVC$          000D 00370 LOADER   15090
$MAIN    MAXCOR$        2400 03270 TASKER   19260
                                   $MAIN    03730
FILPOSN  MAXCYL         18FE 12010 FILPOSN  08980
LDOS60   MAXDAY$        0401 00950 SYSINIT4 21730 22170 22290
FILPOSN  MEA1           190F 12150 FILPOSN  12190
FILPOSN  MEA2           1915 12190 FILPOSN  12170
$MAIN    MINCOR$        3000 03280
LOADER   MOD3BUF        4300 18090 LOADER   18110
```

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

                                    SYSINIT4 23760 23820
SYSINIT4 MOD4REG        2098 24430 SYSINIT4 24410
LDOS60   MODOUT$        0076 00960 SYSINIT4 20630 24100
$MAIN    MODOUT$        0076 01920
LDOS60   MONTBL$        04DC 00970 SYSINIT4 22840
FILPOSN  NEWHIT         17AF 09650 FILPOSN  08910
LDOS60   NFLAG$         0077 00980 TASKER   18380 18730
FILPOSN  NHIT1          17DE 09900 FILPOSN  09920
FILPOSN  NHIT2          17E6 09950 FILPOSN  09970
FILPOSN  NHIT3          1801 10120 FILPOSN  09750
FILPOSN  NHIT4          181F 10370 FILPOSN  09700
FILPOSN  NHIT5          1844 10600 FILPOSN  10730
FILPOSN  NHIT6          1845 10610 FILPOSN  10570 10710
FILPOSN  NHIT7          1848 10630 FILPOSN  10540
FILPOSN  NHIT8          184F 10680
FILPOSN  NHIT9          1859 10740 FILPOSN  10640
SYSINIT4 NOAUT          200D 23730 SYSINIT4 23630
SYSINIT4 NOAUT1         1FF4 23590 SYSINIT4 23560
SYSINIT4 NOFF           207F 24260 SYSINIT4 24210
TASKER   NOTASK         1CE9 20010 TASKER   18250 18250 18250 18250 18260 18260
                                    TASKER   18260 18260 18270 18270 18270 19860
                                    TASKER   20170
SYSINIT4 NOTLEAP        1EDA 21730 SYSINIT4 21690
FILPOSN  NOTOPEN        1584 05330 FILPOSN  05190
SYSINIT4 NOTSG          2069 24160 SYSINIT4 24090
FILPOSN  NSEC1          1379 02040 FILPOSN  01530 02010
FILPOSN  NSEC2          1382 02080
TASKER   NXTMSK         1C2D 18620 TASKER   19030
FILPOSN  NXTSECT        1375 02020 FILPOSN  04630
                                    LOADER   17900
TASKER   NXTVCT         1C29 18590 TASKER   18640
$MAIN    OPREG$         0078 02140 TASKER   18440 18690
                                    SYSINIT4 20670
LDOS60   OPREG$         0078 00990
LDOS60   OPREG_SV_AREA  086E 01000
LDOS60   OPREG_SV_PTR   0835 01010
FILPOSN  ORARET@        14DC 04180 LOADER   13710
$MAIN    OSRLS$         003B 00600
$MAIN    OSVER$         0085 02830
$MAIN    OVERLAY        1E00 03770 LOADER   15670
                                    SYSINIT4 24610
                                    $MAIN    03870
$MAIN    OVRLY$         0069 00890 LOADER   15400 15620
LOADER   OVRLYOLD       1A70 15390 LOADER   15660
LDOS60   PAKNAM$        0410 01020 SYSINIT4 20350
SYSINIT4 PARSDAT        20CC 24760
SYSINIT4 PARTYR         2117 25340 SYSINIT4 23010
LOADER   PAT1           1AE1 16090 LOADER   15960
LOADER   PAT1A          1AEC 16170 LOADER   16090
LOADER   PAT1B          1AEF 16180 LOADER   16170
LDOS60   PAUSE@         0382 01030
LDOS60   PCSAVE$        07AF 01040 TASKER   18340 19250
LDOS60   PDRV$          001B 01050
$MAIN    PDRV$          001B 00450
$MAIN    PHIGH$         001C 00460 SYSINIT4 20740
LOADER   POPERR         1B0E 16400 LOADER   16600
TASKER   POPREGS        1C58 18980 TASKER   18880
FILPOSN  POSN1          145E 03280 FILPOSN  03130 03170 03730 03810 03830
FILPOSN  POSN2          1461 03290 FILPOSN  03230
```

```
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
```

```
The Source   System References     SYSRES - LS-DOS 6.2         Page 00011

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

SYSINIT4 ZERDCB         1E2E 20590 SYSINIT4 20610
LDOS60   ZERO$          0401 01290 SYSINIT4 21160
FILPOSN  ZEROA@         13A0 02200 FILPOSN  02180 02290 02740

00510 Symbols declared  -  00761 References
```

NOTES:

SYS1/SYS

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 318003     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>
```

```
              01290 ;
              01300 ;        Entry from @EXIT & @CMNDI
              01310 ;
              01320 CMD3A
1EB2 7E       01330          LD      A,(HL)          ;Check for comment
1EB3 FE2E     01340          CP      '.'             ;If so go before CR
1EB5 2805     01350          JR      Z,CMD20         ;  is displayed
              01360 ;
1EB7 3E0D     01370          LD      A,CR            ;Do a line feed on
1EB9 CD4206   01380          CALL    @DSP            ;  CMNDI and @EXIT
              01390 ;
              01400 ;        Entry from @CMNDR plus the above
              01410 ;
              01420 ;        Always bring in bank 0
              01430 ;
1EBC AF       01440 CMD20    XOR     A               ;Prepare for bank-0
1EBD 47       01450          LD      B,A             ;Set function and
1EBE 4F       01460          LD      C,A             ;  bank number to 0
1EBF CD7708   01470          CALL    @BANK           ;Invoke bank 0
              01480 ;
              01490 ;        Process the command entry
              01500 ;
1EC2 CD0305   01510          CALL    @LOGER          ;Log the entry
1EC5 11E000   01520          LD      DE,CFCB$        ;Point to command FCB
1EC8 7E       01530          LD      A,(HL)          ;Jump on comment
1EC9 FE2E     01540          CP      '.'
1ECB 2838     01550          JR      Z,COMMENT
1ECD FE2A     01560          CP      '*'             ;Check if alternate CMD
1ECF 2006     01570          JR      NZ,CKNOEXC      ;  processor needed
1ED1 E5       01580          PUSH    HL
1ED2 C1       01590          POP     BC              ;Get Buffer in BC
1ED3 23       01600          INC     HL              ;Move HL past '*'
1ED4 3EFF     01610          LD      A,0FFH          ;Set up for SYS13 entry
1ED6 EF       01620          RST     40              ;  # 7, and do it
1ED7 D621     01630 CKNOEXC  SUB     '!'             ;Test for program force
1ED9 2001     01640          JR      NZ,NOEXC
1EDB 23       01650          INC     HL              ;Bump past the '!'
1EDC 32E61E   01660 NOEXC    LD      (TSTEXC+1),A
1EDF CD5D20   01670          CALL    FSPEC           ;Fetch command spec
1EE2 201D     01680          JR      NZ,WHAT         ;Jump on error
1EE4 E5       01690          PUSH    HL              ;Save terminator pointer
1EE5 3E00     01700 TSTEXC   LD      A,0             ;Test if prog force
1EE7 B7       01710          OR      A
1EE8 2808     01720          JR      Z,NOTLIB        ;Jump if starting "!"
1EEA 01241F   01730          LD      BC,LIBTBL$      ;Pt to tbl of LIB cmds
1EED CD5821   01740          CALL    @FNDPRM         ;Check for a match
1EF0 281F     01750          JR      Z,CMD4          ;Jump if it is
1EF2 21E722   01760 NOTLIB   LD      HL,DFTEXT       ;Else assume prg file, so
1EF5 CDD120   01770          CALL    FEXT            ;  default 'EXT' to CMD
1EF8 E1       01780          POP     HL              ;Rcvr terminator pointer
1EF9 3A6C00   01790          LD      A,(CFLAG$)      ;Ck LIB only execution
1EFC E610     01800          AND     10H             ;CFLAG$ bit-4
1EFE CA1D1B   01810          JP      Z,@RUN          ;The program else WHAT?
              01820 ;
              01830 ;        Process non-entry
              01840 ;
1F01 21FFFF   01850 WHAT     LD      HL,-1           ;Set to show abort
1F04 C9       01860          RET
              01870 ;
              01880 ;        Process "dot" comment
              01890 ;
```

```
1F05 3A7C00    01900 COMMENT LD      A,(SFLAG$)      ;Ret if <DO> in effect
1F08 CB6F      01910         BIT     5,A             ;  else get another
1F0A CA9D1E    01920         JP      Z,CMD2          ;  input line
1F0D 210000    01930         LD      HL,0            ;Set for no error
1F10 C9        01940         RET
               01950 ;
               01960 ;       Process LIB command
               01970 ;
1F11 E1        01980 CMD4    POP     HL              ;Rcvr terminator pointer
1F12 3EC9      01990         LD      A,0C9H          ;Turn off DEBUG
1F14 329F19    02000         LD      (@DBGHK),A
1F17 7A        02010         LD      A,D             ;Test bit 7 of high
1F18 07        02020         RLCA                    ;  order LIB address
1F19 D5        02030         PUSH    DE              ;Ret to address of
1F1A D0        02040         RET     NC              ;  vector if bit 7 = 0
1F1B D1        02050         POP     DE
1F1C 43        02060         LD      B,E             ;Else put overlay # in
1F1D 07        02070         RLCA                    ;Calculate needed library
1F1E 07        02080         RLCA                    ;  by rotating 7-5 into
1F1F C684      02090         ADD     A,84H           ;  2-0 & adding RST base
1F21 EF        02100         RST     28H
               02110 ;
               02120 ;       BOOT code brings back the ROM
               02130 ;
1F22 AF        02140 BOOTIT  XOR     A               ;SVC-0 => @IPL
1F23 EF        02150         RST     40
               02160 ;
               02170 ;       LIBRARY look-up table starts here
               02180 ;
1F24           02190 LIBTBL$ EQU     $               ;Start of library table
               02200 ;
               02210         IF      @SMALL
               02220 ;
               02230 ;       Use this table for SMALL (OEM) library
               02240 ;
               02250 ; DB 'APPEND'
               02260 ; DW LIBA!31H
               02270         DB      'ATTRIB'
               02280         DW      LIBB!51H
               02290         DB      'AUTO  '
               02300         DW      LIBB!11H
               02310 ;DB 'BOOT  '
               02320 ; DW BOOTIT
               02330 ; DB 'BUILD '
               02340 ; DW LIBB!33H
               02350 ; DB 'CAT   '
               02360 ; DW LIBA!20H
               02370 ; DB 'CLS   '
               02380 ; DW LIBA!24H
               02390         DB      'COPY  '
               02400         DW      LIBA!32H
               02410 ; DB 'CREATE'
               02420 ; DW LIBB!13H
               02430         DB      'DATE  '
               02440         DW      LIBB!15H
               02450 ; DB 'DEBUG '
               02460 ; DW LIBB!14H
               02470 ; DB 'DEVICE'
               02480 ; DW LIBA!61H
               02490         DB      'DIR   '
               02500         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 ;
```

```
                03950 ;         Routine to fetch a filespec/devicespec
                03960 ;
205D D5         03970 FSPEC     PUSH    DE              ;Save pointer to DCB
205E CD0521     03980           CALL    @PARSER         ;Parse expected command
2061 203E       03990           JR      NZ,FSP5         ;NZ=not file, ck for device
2063 FE2F       04000           CP      '/'             ;EXT separator?
2065 2007       04010           JR      NZ,FSP1
2067 12         04020           LD      (DE),A          ;File extent coming,
2068 13         04030           INC     DE              ;  get it
2069 0603       04040           LD      B,3             ;EXT is 3-chars max
206B CD0721     04050           CALL    @PAR1
206E FE2E       04060 FSP1      CP      '.'             ;Password entered?
2070 2007       04070           JR      NZ,FSP2
2072 12         04080           LD      (DE),A          ;Password coming,
2073 13         04090           INC     DE              ;  get it also
2074 CD0521     04100           CALL    @PARSER
2077 2035       04110           JR      NZ,FSP6         ;Return if error
2079 FE3A       04120 FSP2      CP      ':'             ;Drive entered?
207B 2009       04130           JR      NZ,FSP3
207D 12         04140           LD      (DE),A          ;A one-byte drive
207E 13         04150           INC     DE              ;  has been had
207F 0601       04160           LD      B,1
2081 CD0721     04170           CALL    @PAR1
2084 2028       04180           JR      NZ,FSP6         ;Return if error
2086 FE21       04190 FSP3      CP      '!'             ;Update EOF always?
2088 2004       04200           JR      NZ,FSP4
208A 12         04210           LD      (DE),A          ;Yes, slow but accurate
208B 13         04220           INC     DE              ;Inc buffer pointers
208C 23         04230           INC     HL
208D 7E         04240           LD      A,(HL)
208E 4F         04250 FSP4      LD      C,A             ;Save separator char
208F 3E03       04260           LD      A,3
2091 12         04270           LD      (DE),A          ;Stuff an ETX
2092 AF         04280           XOR     A
2093 79         04290           LD      A,C             ;P/u separator
2094 D1         04300           POP     DE              ;P/u start of DCB
2095 D5         04310           PUSH    DE
2096 01B020     04320           LD      BC,PREPTBL      ;Ck on prepositions
2099 CD5821     04330           CALL    @FNDPRM
209C D1         04340           POP     DE              ;Can use TO, ON,
209D 28BE       04350           JR      Z,FSPEC         ;  OVER, USING
209F AF         04360           XOR     A
20A0 C9         04370           RET
20A1 FE2A       04380 FSP5      CP      '*'             ;Ck on device spec
20A3 2009       04390           JR      NZ,FSP6         ;Jump if not device
20A5 12         04400           LD      (DE),A          ;  else stuff the '*'
20A6 13         04410           INC     DE
20A7 0602       04420           LD      B,2             ;Xfer two char device
20A9 CD0721     04430           CALL    @PAR1
20AC 28E0       04440           JR      Z,FSP4          ;Terminate buffer
20AE D1         04450 FSP6      POP     DE
20AF C9         04460           RET
                04470 ;
                04480 ;         Preposition table
                04490 ;
20B0 54         04500 PREPTBL   DB      'TO    '
     4F 20 20 20 20
20B6 001D       04510           DW      SBUFF$
20B8 4F         04520           DB      'ON    '
     4E 20 20 20 20
20BE 001D       04530           DW      SBUFF$
```

```
20C0 4F         04540           DB      'OVER  '
     56 45 52 20 20
20C6 001D       04550           DW      SBUFF$
20C8 55         04560           DB      'USING '
     53 49 4E 47 20
20CE 001D       04570           DW      SBUFF$
20D0 00         04580           NOP
                04590 ;
                04600 ;         Fetch default file extension
                04610 ;
20D1 D5         04620 FEXT      PUSH    DE                  ;Save FCB pointer
20D2 E5         04630           PUSH    HL                  ;Save EXT default pointer
20D3 EB         04640           EX      DE,HL               ;Exchange pointers
20D4 23         04650           INC     HL
20D5 0609       04660           LD      B,9                 ;Init for 9-char test
20D7 7E         04670 FEX1      LD      A,(HL)              ;Ret if extension start
20D8 FE2F       04680           CP      '/'                 ;  is found
20DA 280D       04690           JR      Z,FEX3
20DC 380E       04700           JR      C,FEX4              ;Jump on other separator
20DE FE3A       04710           CP      ':'                 ;Jump on digit 0-9
20E0 3804       04720           JR      C,FEX2
20E2 FE41       04730           CP      'A'                 ;Jump on special char
20E4 3806       04740           JR      C,FEX4
20E6 23         04750 FEX2      INC     HL                  ;Advance past A-Z,0-9
20E7 10EE       04760           DJNZ    FEX1
20E9 E1         04770 FEX3      POP     HL                  ;User entered file ext
20EA D1         04780           POP     DE                  ;FCB start
20EB C9         04790           RET
                04800 ;
                04810 ;         Use default extension
                04820 ;
20EC 010F00     04830 FEX4      LD      BC,15               ;Point to position past
20EF 09         04840           ADD     HL,BC               ;  the filespec
20F0 54         04850           LD      D,H
20F1 5D         04860           LD      E,L
20F2 13         04870           INC     DE                  ;Make room for '/EXT'
20F3 13         04880           INC     DE                  ;  which is 4 chars
20F4 13         04890           INC     DE
20F5 13         04900           INC     DE
20F6 03         04910           INC     BC                  ;Now move 16 bytes
20F7 EDB8       04920           LDDR
20F9 E1         04930           POP     HL                  ;Recover pointer to EXT
20FA 23         04940           INC     HL                  ;Point to 3rd char
20FB 23         04950           INC     HL
20FC 0E03       04960           LD      C,3                 ;Move in 3 chars
20FE EDB8       04970           LDDR
2100 3E2F       04980           LD      A,'/'               ;Put in the slash
2102 12         04990           LD      (DE),A
2103 D1         05000           POP     DE                  ;Point back to FCB
2104 C9         05010           RET
                05020 ;
                05030 ;         Get the code for the @PARAM SVC
                05040 ;
2105            05050 *GET      PARAM:3
                03050 ;PARAM/ASM - LS-DOS 6.2
                03060 ;
                03070 ;         Parse a field
                03080 ;         (HL) => command line
                03090 ;         (DE) => FCB area
                03100 ;         (HL) <= 1st byte past non-<A-Z, a-z, 0-9>
                03110 ;                 except 13, 3, "("
               03120 ;         Z    <= found valid field
               03130 ;         NZ   <= found invalid field
               03140 ;
2105 0608      03150 @PARSER LD      B,8               ;Set length
2107 78        03160 @PAR1   LD      A,B
2108 324621    03170         LD      (PAR6+1),A        ;Stuff length for test
210B 04        03180         INC     B
210C 7E        03190 PAR2    LD      A,(HL)
210D FE03      03200         CP      3                 ;ETX?
210F 2826      03210         JR      Z,PAR5
2111 FE0D      03220         CP      CR                ;<ENTER>?
2113 2822      03230         JR      Z,PAR5
2115 FE28      03240         CP      '('               ;Begin of parm?
2117 281E      03250         JR      Z,PAR5
2119 23        03260         INC     HL                ;Bump pointer to next
211A CD4A21    03270         CALL    TST09AZ           ;Test if 0-9,A-Z
211D 300A      03280         JR      NC,PAR3           ;Go if one of the above
211F FE61      03290         CP      'a'               ;Check on lower case
2121 3814      03300         JR      C,PAR5            ;Jump on non-alpha
2123 FE7B      03310         CP      'z'+1             ;Is it a-z?
2125 3010      03320         JR      NC,PAR5           ;Jump on non-alpha
2127 CBAF      03330         RES     5,A               ;Convert lower to upper
2129 05        03340 PAR3    DEC     B                 ;Count down
212A 2808      03350         JR      Z,PAR4
212C 12        03360         LD      (DE),A            ;Xfer the char
212D AF        03370         XOR     A                 ;Show at least 1 valid
212E 324621    03380         LD      (PAR6+1),A        ;Char was detected
2131 13        03390         INC     DE                ;Bump FCB pointer
2132 18D8      03400         JR      PAR2              ;Loop
2134 04        03410 PAR4    INC     B                 ;Here on max chars ck'd
2135 18D5      03420         JR      PAR2
2137 4F        03430 PAR5    LD      C,A               ;Save separator
2138 3E03      03440         LD      A,3               ;Stuff ETX
213A 12        03450         LD      (DE),A
               03460 ;
               03470 ;         Skip over spaces
               03480 ;
213B 79        03490         LD      A,C               ;Was separator a space?
213C FE20      03500         CP      ' '
213E 2005      03510         JR      NZ,PAR6           ;Don't skip if not
2140 BE        03520 PAR5A   CP      (HL)              ;Next char a space?
2141 23        03530         INC     HL
2142 28FC      03540         JR      Z,PAR5A           ;Loop until not
2144 2B        03550         DEC     HL                ;Backup to last non-space
               03560 ;
               03570 ;         Return status of field validity
               03580 ;
2145 3E00      03590 PAR6    LD      A,0               ;Set Z-flag if at least
2147 B7        03600         OR      A                 ;  1 valid char detected
2148 79        03610         LD      A,C               ;Recover separator char
2149 C9        03620         RET
               03630 ;
               03640 ;         Test if 0-9 or A-Z
               03650 ;
214A FE30      03660 TST09AZ CP      '0'               ;Special character?
214C D8        03670         RET     C                 ;Go if not in range
214D FE3A      03680         CP      '9'+1             ;Jump on digit 0-9
214F 3805      03690         JR      C,EXITC           ;Go if 0-9 & make NC
2151 FE41      03700         CP      'A'               ;Jump on spec char
2153 D8        03710         RET     C                 ;Go if 3B-40
2154 FE5B      03720         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
```

```
21A0 D5         04340          PUSH    DE
21A1 78         04350          LD      A,B              ;Don't if trailing length
21A2 B7         04360          OR      A                ;  is zero
21A3 2810       04370          JR      Z,FND5
21A5 13         04380 FND4     INC     DE
21A6 23         04390          INC     HL
21A7 1A         04400          LD      A,(DE)
21A8 FE03       04410          CP      3                ;ETX?
21AA 2822       04420          JR      Z,FND7
21AC FE0D       04430          CP      CR               ;Jump on <ENTER>
21AE 281E       04440          JR      Z,FND7
21B0 BE         04450          CP      (HL)             ;Match?
21B1 2016       04460          JR      NZ,FND6          ;Jump if not
21B3 10F0       04470          DJNZ    FND4             ;  else loop
21B5 D1         04480 FND5     POP     DE               ;Parm matched
21B6 E1         04490          POP     HL               ;Recover begin of parm
21B7 010600     04500 FND5A    LD      BC,6             ;Point to address field
21BA 09         04510          ADD     HL,BC
21BB 4D         04520          LD      C,L              ;Save the response-byte
21BC 44         04530          LD      B,H              ;  pointer in BC
21BD 0B         04540          DEC     BC
21BE 5E         04550          LD      E,(HL)           ;P/u parm table address
21BF 23         04560          INC     HL
21C0 56         04570          LD      D,(HL)
21C1 F1         04580          POP     AF               ;If not enhanced, change
21C2 3802       04590          JR      C,$+4            ;  pointer to bucket
21C4 061D       04600          LD      B,SBUFF$<-8      ;  so we don't alter user
21C6 E1         04610          POP     HL               ;Recover line position
21C7 AF         04620          XOR     A                ;Show found
21C8 C9         04630          RET
21C9 CD4A21     04640 FND6     CALL    TST09AZ          ;Ck if 0-9, A-Z
21CC 3005       04650          JR      NC,FND8          ;Go if in range of above
21CE 7E         04660 FND7     LD      A,(HL)           ;Loop if table has
21CF FE20       04670          CP      ' '              ;  trailing spaces
21D1 28E2       04680          JR      Z,FND5
21D3 D1         04690 FND8     POP     DE
21D4 E1         04700          POP     HL
21D5 18A8       04710          JR      FND2
                04720 ;
                04730 ;        PARAM routine
                04740 ;         (HL) => param line
                04750 ;         (DE) => parm table
                04760 ;         (DE) <= table address value
                04770 ;            C <= # of parm
                04780 ;            Z = OK
                04790 ;           NZ = parm error
                04800 ;
21D7 23         04810 PARAM0   INC     HL               ;Bump the pointer
21D8 7E         04820 PARAM    LD      A,(HL)           ;  and p/u char
21D9 FE0D       04830          CP      CR
21DB C8         04840          RET     Z                ;Return on enter
21DC FE20       04850          CP      ' '
21DE 28F7       04860          JR      Z,PARAM0         ;Loop on space
21E0 FE28       04870          CP      '('
21E2 205E       04880          JR      NZ,PARAM5        ;Jump if not left paren
21E4 1A         04890          LD      A,(DE)           ;Check if enhanced table
21E5 07         04900          RLCA
21E6 3017       04910          JR      NC,PARAM1
21E8 D5         04920          PUSH    DE               ;Save pointer to start
21E9 13         04930          INC     DE               ;Point to 1st TYPE byte
21EA E5         04940          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     05560        CP     ','              ;Comma separator?
2237 28C6     05570        JR     Z,PARAM1
2239 FE0D     05580        CP     CR
223B 2805     05590        JR     Z,PARAM5
223D FE29     05600        CP     ')'              ;Closing paren?
223F 20DF     05610        JR     NZ,ERROUT        ;Leave with ERROR
2241 23       05620        INC    HL               ;Bump line pointer
2242 AF       05630 PARAM5 XOR    A                ;Show all OK
2243 C9       05640        RET
              05650 ;
              05660 ;      Parameter assignment statement
              05670 ;
2244 23       05680 ASSIGN INC    HL               ;Advance token past '='
2245 7E       05690        LD     A,(HL)
2246 FE22     05700        CP     '"'              ;Double quote string?
2248 282A     05710        JR     Z,STRING
224A FE41     05720        CP     'A'              ;Ck on digit or
224C 3815     05730        JR     C,ASS3           ;  special character
224E CBAF     05740        RES    5,A              ;Strip l/c if present
2250 FE58     05750        CP     'X'              ;Hexadecimal?
2252 2807     05760        JR     Z,ASS1
2254 CD9222   05770        CALL   ONOFF            ;Ck on Y, N, ON, OFF
2257 28D2     05780        JR     Z,PARMSW         ;Set FLAG-SWITCH if OK
2259 18C3     05790        JR     PARAM2           ;  else error exit
225B 23       05800 ASS1   INC    HL
225C CDBA22   05810        CALL   HEXVAL           ;Ck on hex format
225F 20BD     05820        JR     NZ,PARAM2        ;Error if bad format
2261 1807     05830        JR     ASS3A            ;  else bypass & set resp
              05840 ;
              05850 ;      Is the parameter numeric or flag ?
              05860 ;
2263 FE30     05870 ASS3   CP     '0'              ;Parameter=number ?
2265 F5       05880        PUSH   AF               ;CF = 0 if number
2266 CDE103   05890        CALL   @DECHEX          ;Cvt # @ HL to bin in DE
2269 F1       05900        POP    AF
226A E3       05910 ASS3A  EX     (SP),HL          ;Get response pointer
226B 3003     05920        JR     NC,ASS4          ;Show numeric if CF=0
226D CBF6     05930        SET    6,(HL)           ;  otherwise show switch
226F 3A       05940        DB     LD___A           ;Skip next instruction
2270 CBFE     05950 ASS4   SET    7,(HL)           ;Set Numeric response bit
2272 18BA     05960        JR     PARAM4
              05970 ;
              05980 ;      Parameter string entry
              05990 ;
2274 23       06000 STRING INC    HL               ;Bypass '"'
2275 44       06010        LD     B,H              ;Save starting address
2276 4D       06020        LD     C,L
2277 7E       06030 STR1   LD     A,(HL)           ;P/u a char
2278 FE20     06040        CP     20H
227A 38A2     06050        JR     C,PARAM2         ;Exit on control char
227C 23       06060        INC    HL               ;Bump pointer
227D FE22     06070        CP     '"'              ;Closing double quote
227F 20F6     06080        JR     NZ,STR1
2281 E5       06090        PUSH   HL               ;Save current pointer
2282 ED42     06100        SBC    HL,BC            ;Calc length of string
2284 7D       06110        LD     A,L
2285 3D       06120        DEC    A                ;Adjust for INC HL
2286 FE20     06130        CP     32               ;If len > 31, set to 0
2288 3801     06140        JR     C,$+3
228A AF       06150        XOR    A
228B E1       06160        POP    HL               ;Rcvr pointer
```

```
228C E3          06170          EX      (SP),HL         ;Get response byte
228D F620        06180          OR      20H             ;Set FLAG-STRING
228F 77          06190          LD      (HL),A
2290 189C        06200          JR      PARAM4
                 06210 ;
                 06220 ;        Check for Yes, No, On, Off
                 06230 ;
2292 010000      06240 ONOFF    LD      BC,0            ;Init to FALSE
2295 D659        06250          SUB     'Y'             ;Is it Yes?
2297 2811        06260          JR      Z,ONO1          ;Jump on yes
2299 C60B        06270          ADD     A,'Y'-'N'       ;Is it No?
229B 2810        06280          JR      Z,ONO2          ;Jump on no
229D 3D          06290          DEC     A               ;Is it 'O'n or 'O'ff?
229E C0          06300          RET     NZ              ;Return if not on/off
229F 23          06310          INC     HL              ;Bump pointer to next
22A0 7E          06320          LD      A,(HL)          ;  character & p/u
22A1 CBAF        06330          RES     5,A             ;Set lower to upper
22A3 FE46        06340          CP      'F'
22A5 2806        06350          JR      Z,ONO2          ;Jump on off
22A7 FE4E        06360          CP      'N'
22A9 C0          06370          RET     NZ              ;Return if not on
22AA 01FFFF      06380 ONO1     LD      BC,-1           ;Init to true
22AD 23          06390 ONO2     INC     HL              ;Ignore trailing part
22AE 7E          06400          LD      A,(HL)          ;  of word until closing
22AF FE29        06410          CP      ')'             ;  ")" or comma separator
22B1 C8          06420          RET     Z               ;  or CR
22B2 FE0D        06430          CP      CR
22B4 C8          06440          RET     Z
22B5 FE2C        06450          CP      ','
22B7 C8          06460          RET     Z
22B8 18F3        06470          JR      ONO2            ;Loop
                 06480 ;
                 06490 ;        Process hexadecimal assignment
                 06500 ;
22BA 010000      06510 HEXVAL   LD      BC,0            ;Init value to zero
22BD 7E          06520          LD      A,(HL)          ;P/u a char
22BE FE27        06530          CP      27H             ;Must be single quote
22C0 C0          06540          RET     NZ              ;Ret if not
22C1 23          06550 HEX1     INC     HL              ;Bump past it
22C2 7E          06560          LD      A,(HL)          ;P/u possible hex digit
22C3 D630        06570          SUB     30H             ;Begin conversion
22C5 380C        06580          JR      C,HEX2          ;Jump if < "0"
22C7 FE0A        06590          CP      10              ;Ck for 0-9
22C9 3810        06600          JR      C,HEX3          ;Jump if digit is 0-9
22CB CBAF        06610          RES     5,A             ;Strip l/c if present
22CD D607        06620          SUB     7               ;  else ck A-F
22CF FE10        06630          CP      16
22D1 3808        06640          JR      C,HEX3          ;Jump if A-F
22D3 7E          06650 HEX2     LD      A,(HL)          ;Test for closing quote
22D4 FE27        06660          CP      27H
22D6 23          06670          INC     HL              ;Bump pointer
22D7 C8          06680          RET     Z               ;Ret if closing quote
22D8 2B          06690          DEC     HL              ;  else backup, set OK,
22D9 AF          06700          XOR     A               ;  then return
22DA C9          06710          RET
22DB C5          06720 HEX3     PUSH    BC              ;Exchange BC & HL
22DC E3          06730          EX      (SP),HL         ;  and save HL
22DD 29          06740          ADD     HL,HL           ;Multiply by 16
22DE 29          06750          ADD     HL,HL
22DF 29          06760          ADD     HL,HL
22E0 29          06770          ADD     HL,HL
```

```
22E1 44         06780          LD      B,H             ;Merge new digit
22E2 85         06790          ADD     A,L
22E3 4F         06800          LD      C,A
22E4 E1         06810          POP     HL              ;Recover pointer
22E5 18DA       06820          JR      HEX1            ;Loop
                05060 ;
22E7 43         05070 DFTEXT   DB      'CMD'           ;Default extension
     4D 44
                05080          IF      @MOD2
                05090 RDYMSG$  DB      LF,14,'LS-DOS Ready',CR
                05100          ELSE
22EA 0A         05110 RDYMSG$  DB      LF,14,'TRSDOS Ready',CR
     0E 54 52 53 44 4F 53 20
     52 65 61 64 79 0D
                05120          ENDIF
22F9            05130 LAST     EQU     $
                05140          IFGT    $,DIRBUF$
                05150          ERR     'Module too big'
                05160          ENDIF
23FE            05170          ORG     MAXCOR$-2
23FE F904       05180          DW      LAST-SYS1       ;Size of overlay
1E00            05190          END     SYS1
```

```
$?1             21EB $?2            21FD $A1            03B7
$A2             03B8 $A3            03B9 $CKEOF         1470
@$SYS           08F0 @@1            0000 @@2            0000
@@3             0000 @@4            0000 @ABORT         1B08
@ADTSK          1CDA @BANK          0877 @BKSP          1486
@BREAK          196F @BYTEIO        1300 @CHNIO         0689
@CKBRKC         0553 @CKDRV         1993 @CKEOF         158F
@CKTSK          1CF5 @CLOSE         1999 @CLS           0545
@CMNDI          197E @CMNDR         197B @CTL           0623
@DATE           07A8 @DBGHK         199F @DCINIT        19C0
@DCRES          19C4 @DCSTAT        19B5 @DCTBYT        1A2B
@DEBUG          19A0 @DECHEX        03E1 @DIRCYL        18F7
@DIRRD          18BB @DIRWR         1803 @DIV16         06E3
@DIV8           1927 @DODIR         19AF @DOKEY         19A9
@DSP            0642 @DSPLY         052D @ERROR         1B0F
@EXIT           1B0B @FEXT          1984 @FLAGS         196A
@FNAME          199C @FNDPRM        2158 @FRENCH        0000
@FSPEC          1981 @GATRD         1874 @GATWR         1875
@GERMAN         0000 @GET           0638 @GTDCB         1990
@GTDCT          1A1E @GTMOD         19B2 @HDFMT         19E4
@HEX16          07BD @HEX8          07C2 @HEXDEC        06F6
@HIGH$          1948 @HITRD         1897 @HITWR         1898
@HZ50           0000 @ICNFG         0086 @INIT          198D
@INTL           0000 @IPL           1BF2 @JCL           0630
@KBD            0635 @KEY           0628 @KEYIN         0585
@KITSK          0089 @KLTSK         1CD0 @LOAD          1B38
@LOC            14B3 @LOF           14DE @LOGER         0503
@LOGOT          0500 @MOD2          0000 @MOD4          FFFF
@MSG            0530 @MUL16         06C9 @MUL8          190A
@NMI            0066 @OPEN          198A @OPREG         0084
@PAR1           2107 @PARAM         1987 @PARSER        2105
@PAUSE          0382 @PEOF          14A2 @POSN          1434
@PRINT          0528 @PRT           063D @PUT           0645
@RAMDIR         19AC @RDHDR         19D8 @RDSEC         19F4
@RDSSC          18D8 @RDTRK         19E0 @READ          1513
@REMOVE         19A6 @RENAME        1996 @REW           149B
@RMTSK          1CD7 @RPTSK         1CEB @RREAD         1473
@RSLCT          19D4 @RST00         0000 @RST08         0008
@RST10          0010 @RST18         0018 @RST20         0020
@RST28          0028 @RST30         0030 @RST38         0038
@RSTNMI         0FE9 @RSTOR         19C8 @RSTREG        0680
@RUN            1B1D @RWRIT         13AD @SEEK          19D0
@SEEKSC         1421 @SKIP          1430 @SLCT          19BC
@SMALL          0000 @SOUND         0392 @STEPI         19CC
@TIME           078D @USA           FFFF @VDCTL         0B99
@VDCTL3         0D38 @VER           1560 @VRSEC         19DC
@WEOF           14EC @WHERE         1979 @WRITE         1531
@WRSEC          19E8 @WRSSC         19EC @WRTRK         19F0
@_VDCTL         0D42 ADDR_2_ROWCOL  0DF1 AFLAG$         006A
ASS1            225B ASS3           2263 ASS3A          226A
ASS4            2270 ASSIGN         2244 AUTO?          1FF1
BAR$            0201 BOOTIT         1F22 BOOTST$        439D
BREAK?          1C60 BRKVEC$        1C88 BUR$           0200
CASHK$          0A7B CFCB$          00E0 CFGFCB$        00E0
CFLAG$          006C CKMOD@         1A7F CKNOEXC        1ED7
CKOPEN@         1568 CLEANUP        1E46 CMD            1E41
CMD1A           1E97 CMD2           1E9D CMD20          1EBC
CMD30           1E3C CMD3A          1EB2 CMD4           1F11
CMDCONT         1E8E COMMENT        1F05 CONFIG$        203F
CORE$           0300 CR             000D CRTBGN$        F800
```

```
CYL_GRN        16AE D@FBYT8        1A26 DATE$          0033
DAYTBL$        04C7 DBGOFF         1E5F DBGSV$         00A0
DCBKL$         0031 DCT$           0470 DCTBYT8@       1A29
DCTFLD@        1A34 DFLAG$         006D DFTEXT         22E7
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
DTPMT$         04C2 DVREND$        0FF4 DVRHI$         0206
EFLAG$         006E ENADIS_DO_RAM  0817 ERROUT         2220
EXITC          2156 EXTDBG$        19A4 FDDINT$        000E
FEMSK$         006F FEX1           20D7 FEX2           20E6
FEX3           20E9 FEX4           20EC FEXT           20D1
FLGTAB$        006A FND1           2161 FND1A          2170
FND2           217F FND3           218B FND3A          219D
FND4           21A5 FND5           21B5 FND5A          21B7
FND6           21C9 FND7           21CE FND8           21D3
FSP1           206E FSP2           2079 FSP3           2086
FSP4           208E FSP5           20A1 FSP6           20AE
FSPEC          205D GET_@_ROWCOL   0DAE HERTZ$         0750
HEX1           22C1 HEX2           22D3 HEX3           22DB
HEXVAL         22BA HIGH$          040E HKRES$         1A6C
IFLAG$         0072 INBUF$         0420 INTIM$         003C
INTMSK$        003D INTVC$         003E JCLCB$         0203
JDCB$          0024 JFCB$          00C0 JLDCB$         0230
JRET$          0026 KCK@           07D6 KFLAG$         0074
KIDATA$        08FC KIDCB$         0208 LAST           22F9
LBANK$         0202 LDRV$          0023 LD___A         003A
LF             000A LFLAG$         0075 LIBA           8000
LIBB           A000 LIBC           C000 LIBTBL$        1F24
LNKFCB@        1566 LOW$           001E LSVC$          000D
MAXCOR$        2400 MAXDAY$        0401 MINCOR$        3000
MODOUT$        0076 MONTBL$        04DC NFLAG$         0077
NOEXC          1EDC NOTLIB         1EF2 ONO1           22AA
ONO2           22AD ONOFF          2292 OPREG$         0078
OPREG_SV_AREA  086E OPREG_SV_PTR   0835 ORARET@        14DC
OSRLS$         003B OSVER$         0085 OVRLY$         0069
PAKNAM$        0410 PAR2           210C PAR3           2129
PAR4           2134 PAR5           2137 PAR5A          2140
PAR6           2145 PARAM          21D8 PARAM0         21D7
PARAM1         21FF PARAM2         221E PARAM3         2223
PARAM4         222E PARAM5         2242 PARMSW         222B
PAUSE@         0382 PCSAVE$        07AF PDRV$          001B
PHIGH$         001C PRDCB$         0218 PREPTBL        20B0
PUTA@DE        0DCD PUT_@          0DCA PUT_@_ROWCOL   0DC6
RDYMSG$        22EA RFLAG$         007B ROWCOL_2_ADDR  0DD0
RST38@         1BFF RSTOR$         04C4 RWRIT@         13A2
S1DCB$         0238 SBUFF$         1D00 SET@EXEC       1A79
SET_SCROLL     0CF3 SFCB$          008C SFLAG$         007C
SIDCB$         0220 SODCB$         0228 SPACE4$        2142
STACK$         0380 START$         0000 STR1           2277
STRING         2274 SVCRET$        000B SVCTAB$        0100
SYS1           1E00 SYS1BGN        1E04 SYSERR$        1B13
TCB$           004E TFLAG$         007D TIME$          002D
TIMER$         002C TIMSL$         002B TIMTSK$        0713
TMPMT$         04C3 TRACE_INT      07B1 TST09AZ        214A
TSTEXC         1EE5 TYPHK$         0A8F TYPTSK$        0B26
USTOR$         0013 VFLAG$         007F WHAT           1F01
WRINT$         0080 ZERO$          0401 ZEROA@         13A0
```

```
00000 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 210802   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        01900          LD      E,L               ;Sector 0 for GAT
1EEC CDD818    01910          CALL    @RDSSC            ;Read the GAT
1EEF 2031      01920          JR      NZ,CKDR7          ;Jump on error
1EF1 2ACC1D    01930          LD      HL,(SBUFF$+0CCH)          ;P/u excess tracks
1EF4 3E22      01940          LD      A,22H             ;Add offset
1EF6 85        01950          ADD     A,L
1EF7 FD7706    01960          LD      (IY+6),A          ;Max track # to DCT
1EFA FDCB04AE  01970          RES     5,(IY+4)          ;Set to side 0
1EFE CB6C      01980          BIT     5,H               ;Test double sided
1F00 2804      01990          JR      Z,CKDR3           ;Jump if only single
1F02 FDCB04EE  02000          SET     5,(IY+4)          ;Set for side 2
1F06 F1        02010 CKDR3    POP     AF                ;Recover FDC status
1F07 07        02020 CKDR3A   RLCA                      ;Shift write prot to 7
1F08 FDB603    02030          OR      (IY+3)            ;Merge Soft WP bit
1F0B E680      02040          AND     80H               ;Strip all but 7
1F0D 326E20    02050          LD      (OPNCB9+1),A      ;Save WP status for OPNCB
1F10 87        02060          ADD     A,A               ;Write prot to carry flg
               02070 ;
1F11           02080 CKDR4    EQU     $
1F11 FB        02090          EI
1F12 D1        02100          POP     DE
1F13 E1        02110          POP     HL
1F14 FDE1      02120 CKDR5    POP     IY
1F16 C9        02130          RET
1F17 7C        02140 INDEX    LD      A,H
1F18 B5        02150          OR      L
1F19 2807      02160          JR      Z,CKDR7
1F1B 2B        02170          DEC     HL
1F1C CDD419    02180          CALL    @RSLCT            ;Check for index pulse
1F1F CB4F      02190          BIT     1,A               ;Test index
1F21 C9        02200          RET
1F22 F1        02210 CKDR7    POP     AF
               02220 ;
1F23 B7        02230 CKDR7A   OR      A                 ;Set NZ ret
1F24 18EB      02240          JR      CKDR4             ;  and exit
               02250 ;
               02260 ;        OPEN a device
               02270 ;        Device Control Blocks are from X'0208' - X'02FF'
               02280 ;
1F26 CD591E    02290 DEVOPEN CALL     GETDCB            ;Find the DCB named
1F29 C0        02300          RET     NZ                ;  in the IX pointer
               02310 ;
               02320 ;        Found the needed Device Control Block
               02330 ;
1F2A 44        02340 DEV4     LD      B,H               ;Xfer dcb vector to BC
1F2B 4D        02350          LD      C,L
1F2C DDE5      02360          PUSH    IX                ;User DCB to HL
1F2E E1        02370          POP     HL
1F2F 3610      02380          LD      (HL),10H          ;Show routed
1F31 23        02390          INC     HL
1F32 71        02400          LD      (HL),C            ;Stuff dcb vector
1F33 23        02410          INC     HL
1F34 70        02420          LD      (HL),B
1F35 23        02430          INC     HL
1F36 AF        02440          XOR     A                 ;Zero next 3 bytes
1F37 77        02450          LD      (HL),A
1F38 23        02460          INC     HL
1F39 77        02470          LD      (HL),A
1F3A 23        02480          INC     HL
1F3B 77        02490          LD      (HL),A
1F3C 23        02500          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
```

```
2033 36C9                LD      (HL),0C9H       ;  off DEBUG
2035 32A320     OPEN16   LD      (OPNCB1+1),A    ;Stuff access level
2038 E1                  POP     HL              ;Ptr to direc record
2039 C1                  POP     BC              ;P/u DEC and drive
                ;
                ;        Routine to open up the fcb from the directory
                ;        HL => directory record in SBUFF$
                ;        BC => DEC and drive used for directory read/write
                ;        IX => pointer to File Control Block
                ;
203A FDE5       OPNCB    PUSH    IY              ;Save IY
203C E5                  PUSH    HL              ;Transfer direc record
203D FDE1                POP     IY              ;  ptr to IY
203F C5                  PUSH    BC              ;Save DEC and drive
2040 CD9320              CALL    OPNCB0          ;Create the opened FCB
2043 C1                  POP     BC
2044 212420              LD      HL,OPEN14+1     ;If from LOAD, don't do
2047 CB46                BIT     0,(HL)          ;  any further checks
2049 2804                JR      Z,OPNEX1
204B AF                  XOR     A
204C FDE1       OPNEX    POP     IY
204E C9                  RET
204F FDCB016E   OPNEX1   BIT     5,(IY+1)        ;If file already open
2053 280F                JR      Z,OPNCB8        ;  then set read-only
2055 FDE1                POP     IY              ;  & return "file open...
2057 DD7E01     OPNEX2   LD      A,(IX+1)        ;P/u current attributes
205A E6F8                AND     0F8H            ;Mask off current prot
205C F605                OR      5               ;  & replace with READ
205E DD7701              LD      (IX+1),A        ;Reset access to READ
2061 3E29                LD      A,41            ;Set "file already open"
2063 C9                  RET
                ;
                ;        If access level is > read, set file open flag in
                ;        the directory & note close authority in the FCB.
                ;
2064 DD7E01     OPNCB8   LD      A,(IX+1)        ;P/u FCB access level
2067 E607                AND     7               ;Mask off other junk
2069 FE05                CP      5               ;Ck READ, EXEC, NONE
206B 3017                JR      NC,OPNCB10      ;Go if one of the above
206D 3E00       OPNCB9   LD      A,0             ;P/u CKDRV status
206F 07                  RLCA                    ;Was drive write prot?
2070 381C                JR      C,FRCREAD       ;CF = WP
2072 FDCB01EE            SET     5,(IY+1)        ;Set file open in direc
2076 3A7700              LD      A,(NFLAG$)      ;P/u Nflag
2079 CB47                BIT     0,A             ;Check for function ON
207B C40318              CALL    NZ,@DIRWR       ;Write the directory
207E 20CC                JR      NZ,OPNEX
2080 DDCB00F6            SET     6,(IX+0)        ;Set close authority
                ;
                ;        Ck if passed LRL matches directory
                ;
2084 DD7E09     OPNCB10  LD      A,(IX+9)        ;P/u LRL from FCB &
2087 FDBE04              CP      (IY+4)          ;  compare with directory
208A 3E2A                LD      A,42            ;Init "LRL open fault
208C 18BE                JR      OPNEX
                ;
                ;        Disk write protected - Change access to READ
                ;
208E CD5720     FRCREAD  CALL    OPNEX2          ;Change access to READ
2091 18F1                JR      OPNCB10
                ;
```

```
               04950 ;         This routine creates the open file control block
               04960 ;
2093 EB        04970 OPNCB0  EX      DE,HL
2094 DDE5      04980         PUSH    IX              ;Transfer fcb pointer
2096 E1        04990         POP     HL
2097 1A        05000         LD      A,(DE)          ;Get DIR+0
2098 E620      05010         AND     20H             ;Keep "PDS" bit & show
209A F680      05020         OR      80H             ;  FCB as open
209C 77        05030         LD      (HL),A          ;Shove into FCB+0
209D 23        05040         INC     HL
209E 3AFE22    05050         LD      A,(LREC$)       ;P/u lrl
20A1 B7        05060         OR      A               ;Test for 0 (256)
20A2 3E00      05070 OPNCB1  LD      A,0             ;Now start byte 2 with
20A4 2802      05080         JR      Z,OPNCB2        ;  that set by "OPEN16"
20A6 F680      05090         OR      80H             ;Show sector or byte I/O
20A8 F620      05100 OPNCB2  OR      20H             ;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 F600      05170 OPNCB3  OR      0
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 0
20B0 23        05220         INC     HL
20B1 D5        05230         PUSH    DE              ;Put address of disk I/O
20B2 110000    05240 OPNCB4  LD      DE,0            ;  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 0 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 70        05340         LD      (HL),B          ;FCB+7 with DEC
20BF 23        05350         INC     HL
20C0 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+10 & FCB+11
20CE 23        05480         INC     HL              ;  with zero for NRN
20CF 77        05490         LD      (HL),A
20D0 23        05500         INC     HL
20D1 CBE3      05510         SET     4,E             ;Point to file EOF
20D3 010200    05520         LD      BC,2            ;Move ERN
20D6 EB        05530         EX      DE,HL
20D7 EDB0      05540         LDIR                    ;  and zero BC reg
20D9 EB        05550         EX      DE,HL
```

```
20DA 3E05     05560          LD      A,5              ;Max 5 extents
20DC F5       05570          PUSH    AF
20DD 1A       05580 OPNCB5   LD      A,(DE)           ;Move starting track
20DE 77       05590          LD      (HL),A
20DF 23       05600          INC     HL
20E0 13       05610          INC     DE
20E1 1A       05620          LD      A,(DE)           ;Move grans & offset
20E2 77       05630          LD      (HL),A
20E3 23       05640          INC     HL
20E4 E61F     05650          AND     1FH              ;Strip out grans
20E6 3C       05660          INC     A                ;Bump for zero offset
              05670 ;
              05680 ;        Add reg A to reg pair BC
              05690 ;
20E7 81       05700          ADD     A,C              ;Add previous count
20E8 4F       05710          LD      C,A              ;Update C
20E9 3001     05720          JR      NC,$+3           ;Go if no carry to B
20EB 04       05730          INC     B
20EC F1       05740          POP     AF               ;Recover counter
20ED 3D       05750          DEC     A                ;Decrement loop
20EE C8       05760          RET     Z                ;Done if moved in 5
20EF F5       05770          PUSH    AF
20F0 13       05780          INC     DE
20F1 1A       05790          LD      A,(DE)           ;Test for end of extents
20F2 FEFE     05800          CP      0FEH             ;Extent in use?
20F4 3006     05810          JR      NC,OPNCB6        ;Jump if not
20F6 71       05820          LD      (HL),C           ;Stuff # of cumulative
20F7 23       05830          INC     HL               ;  grans to this
20F8 70       05840          LD      (HL),B           ;  allocation into FCB
20F9 23       05850          INC     HL
20FA 18E1     05860          JR      OPNCB5           ;Loop for next
              05870 ;
              05880 ;        Unused extents - put X'FFFF' in remaining fields
              05890 ;
20FC F1       05900 OPNCB6   POP     AF               ;Recover counter
20FD 07       05910          RLCA                     ;Make times 4 and
20FE 07       05920          RLCA                     ;  fill remaining
20FF 47       05930          LD      B,A              ;  extent bytes with
2100 36FF     05940 OPNCB7   LD      (HL),0FFH        ;  0FFH
2102 23       05950          INC     HL
2103 10FB     05960          DJNZ    OPNCB7
2105 C9       05970          RET
              05980 ;
              05990 ;        INIT a file
              06000 ;         HL => the address of a 256-byte buffer
              06010 ;         DE => the address of a 32-byte FCB
              06020 ;          B => the logical record length (LREC)
              06030 ;
2106 CD6615   06040 INIT     CALL    LNKFCB@          ;Link to FCB
2109 32A320   06050          LD      (OPNCB1+1),A     ;Start FCB+1 with 0
210C E5       06060          PUSH    HL
210D 217C00   06070          LD      HL,SFLAG$        ;Reset called by RUN bit
2110 CB96     06080          RES     2,(HL)
2112 E1       06090          POP     HL
2113 CD441F   06100          CALL    OPEN1            ;Can we "OPEN" the file?
2116 C8       06110          RET     Z                ;Return if file existing
2117 FE18     06120          CP      24               ;Return if error not
2119 C0       06130          RET     NZ               ;  "file not found"
211A 3E10     06140          LD      A,10H            ;Set dir rec to show
211C 32FA22   06150          LD      (DIR$INIT),A     ;  assigned
211F 3A771F   06160          LD      A,(OPEN2+1)      ;P/u the drive entry
```

```
2122 4F        06170          LD      C,A
2123 3C        06180          INC     A               ;Jump if a drive entry
2124 F5        06190          PUSH    AF
2125 2001      06200          JR      NZ,INIT1        ;  was made
2127 4F        06210          LD      C,A
2128 F1        06220 INIT1    POP     AF              ;Stack integrity
2129 CD7A1E    06230          CALL    CKDRV           ;Is this drive available?
212C 200C      06240          JR      NZ,INIT2        ;Jump if not
212E 380A      06250          JR      C,INIT2         ;  or if write protected
2130 CD9718    06260          CALL    @HITRD          ;Read hash index table
2133 C0        06270          RET     NZ              ;Return if read error
2134 CDEE21    06280          CALL    SPRHIT          ;Locate spare entry
2137 281F      06290          JR      Z,INIT4         ;Jump if space
2139 AF        06300          XOR     A               ;Set status of CKDRV=Z
213A F5        06310 INIT2    PUSH    AF              ;Save last CKDRV status
213B CD9E1F    06320          CALL    TESTDRV
213E 38E8      06330          JR      C,INIT1         ;Loop if not at end
2140 3A771F    06340          LD      A,(OPEN2+1)     ;If drivespec not entered
2143 3C        06350          INC     A               ;  then "directory full
2144 2003      06360          JR      NZ,INIT2A
2146 F1        06370          POP     AF              ;Stack integrity
2147 1805      06380          JR      ERR26
2149 F1        06390 INIT2A   POP     AF              ;  else if no drive then
214A 2008      06400          JR      NZ,ERR32        ;  "illegal drive...
214C 3803      06410          JR      C,ERR15         ;  else if write protect
214E 3E1A      06420 ERR26    LD      A,26            ;  "directory space full"
2150 01        06430          DB      1
2151 3E0F      06440 ERR15    LD      A,15            ;"write protect...
2153 01        06450          DB      1
2154 3E20      06460 ERR32    LD      A,32            ;"Illegal drive...
2156 B7        06470          OR      A
2157 C9        06480          RET
               06490 ;
               06500 ;        Found a spare HIT entry position
               06510 ;
2158 45        06520 INIT4    LD      B,L             ;Save DEC
2159 3AE022    06530          LD      A,(FILEHASH)    ;P/u filespec hash
215C 77        06540          LD      (HL),A          ;  & store in HIT
215D CD9818    06550          CALL    @HITWR          ;Write updated HIT
2160 CCBB18    06560          CALL    Z,@DIRRD        ;Read that dir record
2163 C0        06570          RET     NZ              ;Return if read error
2164 E5        06580          PUSH    HL
2165 C5        06590          PUSH    BC
2166 EB        06600          EX      DE,HL
2167 010500    06610          LD      BC,5            ;Move 1st 5 bytes into
216A 21FA22    06620          LD      HL,DIR$INIT     ;  directory record
216D EDB0      06630          LDIR
216F 0E11      06640          LD      C,17            ;Move filename & password
2171 21E922    06650          LD      HL,NAME$EXT     ;  info into directory
2174 EDB0      06660          LDIR
2176 EB        06670          EX      DE,HL
2177 060A      06680          LD      B,10            ;Put X'FFFF' into 5 ext's
2179 CD0021    06690 INIT5    CALL    OPNCB7          ;4 for the ext's & 1 for
217C C1        06700          POP     BC              ;  starting info
217D CD0318    06710          CALL    @DIRWR          ;Write updated directory
2180 E1        06720          POP     HL
2181 C0        06730          RET     NZ              ;Return if write error
2182 CD3A20    06740          CALL    OPNCB           ;  else open the fcb
2185 37        06750          SCF                     ;Indicate new file
2186 C9        06760          RET
               06770 ;
```

```
                06780 ;         Xfer the file spec to system buffer area
                06790 ;
2187 0613       06800 XFRSPEC LD      B,19
2189 11E122     06810         LD      DE,PSWDBUF
218C 3E20       06820         LD      A,20H           ;Blank out the filename
218E 12         06830 XSPEC1  LD      (DE),A          ;  field in system buffer
218F 13         06840         INC     DE
2190 10FC       06850         DJNZ    XSPEC1
2192 3EFF       06860         LD      A,0FFH          ;Set drive to X'FF' for
2194 32771F     06870         LD      (OPEN2+1),A     ;  checking user entry
2197 1EE9       06880         LD      E,NAME$EXT&0FFH ;Xfer file name
2199 CDD221     06890         CALL    XSPEC8
219C 4F         06900         LD      C,A
219D 78         06910         LD      A,B
219E D608       06920         SUB     8               ;Any valid chars found?
21A0 2003       06930         JR      NZ,XSPEC3       ;Jump if valid name
                06940 ;
                06950 ;         Filename was invalid format
                06960 ;
21A2 F613       06970         OR      19              ;"illegal file name"
21A4 C9         06980         RET
                06990 ;
                07000 ;         Continue to check file spec
                07010 ;
21A5 79         07020 XSPEC3  LD      A,C
21A6 FE2F       07030         CP      '/'             ;Ext entered?
21A8 1EF1       07040         LD      E,FILE$EXT&0FFH
21AA 0603       07050         LD      B,3
21AC CCD421     07060         CALL    Z,XSPEC8A       ;Xfer the ext
21AF FE2E       07070         CP      '.'             ;Password entered?
21B1 1EE1       07080         LD      E,PSWDBUF&0FFH
21B3 CCD221     07090         CALL    Z,XSPEC8        ;Xfer the password
21B6 FE3A       07100         CP      ':'             ;Drive entered?
21B8 200D       07110         JR      NZ,XSPEC6
21BA 7E         07120         LD      A,(HL)          ;P/u drive #
21BB D630       07130         SUB     '0'             ;Convert to binary
21BD 32771F     07140         LD      (OPEN2+1),A     ;Stuff drive #
21C0 E6F8       07150         AND     0F8H            ;Must be <0-7>
21C2 3E20       07160         LD      A,32            ;"illegal drive #"
21C4 C0         07170         RET     NZ              ;Return error if out
21C5 23         07180         INC     HL              ;  of range
21C6 7E         07190         LD      A,(HL)          ;Does filespec end in
21C7 D621       07200 XSPEC6  SUB     21H             ;  exclamation point?
21C9 3E08       07210         LD      A,8             ;Init to set bit 3 of
21CB 2801       07220         JR      Z,XSPEC7        ;  FCB+1 & jump if "!"
21CD AF         07230         XOR     A               ;  else reset if not
21CE 32AB20     07240 XSPEC7  LD      (OPNCB3+1),A
21D1 C9         07250         RET
                07260 ;
                07270 ;
                07280 ;
21D2 0608       07290 XSPEC8  LD      B,8
21D4 7E         07300 XSPEC8A LD      A,(HL)          ;P/u a filespec character
21D5 23         07310         INC     HL              ;  & 1st test for A-Z
21D6 1809       07320         JR      XSPEC10
21D8 7E         07330 XSPEC9  LD      A,(HL)          ;P/u a filespec character
21D9 23         07340         INC     HL              ;Advance to next one
21DA FE30       07350         CP      '0'             ;Check for 0-9
21DC D8         07360         RET     C
21DD FE3A       07370         CP      '9'+1
21DF 3806       07380         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 D0        07420         RET     NC
21E7 12        07430 XSPEC11 LD      (DE),A         ;Character is valid
21E8 13        07440         INC     DE             ;Advance to next one
21E9 10ED      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 3E07      07550         LD      A,7            ;Get highest # sector
21F0 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 0 offset
21F9 AA        07620         XOR     D              ;Recover # heads
21FA 07        07630         RLCA                   ;  into bits 0-2
21FB 07        07640         RLCA
21FC 07        07650         RLCA
21FD 3C        07660         INC     A              ;Bump for 0 offset
21FE CD0A19    07670         CALL    @MUL8          ;Multiply sectors x heads
2201 5F        07680         LD      E,A            ;Now check double bit
2202 3E04      07690         LD      A,4
2204 CD2B1A    07700         CALL    @DCTBYT
2207 CB6F      07710         BIT     5,A            ;Set if 2-sided
2209 7B        07720         LD      A,E
220A 2801      07730         JR      Z,ONESID       ;Go if not set else
220C 87        07740         ADD     A,A            ;  double value
220D D1        07750 ONESID  POP     DE
220E D602      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 2E01      07850         LD      L,1            ;Start after DIR slot
221B 2C        07860 GSHLOOP INC     L              ;Step to next
221C 2002      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 FE00      07920 GSH3    CP      0              ;Cp with # of dir sectors
2225 7D        07930         LD      A,L
2226 3805      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         08000           RET     Z               ;Done if so
2230 18E9       08010           JR      GSHLOOP         ;Try next
                08020 ;
                08030 ;         Routine to rename a filespec/devspec
                08040 ;
2232 3E18       08050 REN0      LD      A,18H
2234 320620     08060           LD      (WASMAT),A
2237 B7         08070           OR      A               ;Denote "file not in dir
2238 C9         08080           RET
2239 CD6615     08090 RENAME    CALL    LNKFCB@         ;Save regs & link to IX
223C DD7E00     08100           LD      A,(IX+0)        ;If a device, use the
223F D62A       08110           SUB     '*'             ;  "device" routine
2241 2879       08120           JR      Z,RENDEV
2243 FEA8       08130           CP      'R'!80H-'*'     ;Special open condition?
2245 28EB       08140           JR      Z,REN0          ;Go if so
2247 E5         08150           PUSH    HL              ;Save new pointer
2248 217C00     08160           LD      HL,SFLAG$       ;Set don't test flags
224B CBC6       08170           SET     0,(HL)
224D CD441F     08180           CALL    OPEN1           ;Open the "old" spec
2250 E1         08190           POP     HL
2251 C0         08200           RET     NZ              ;Exit on error
2252 DD7E01     08210           LD      A,(IX+1)        ;Make sure user has
2255 E607       08220           AND     7               ;  permission to rename
2257 FE03       08230           CP      3
2259 3804       08240           JR      C,REN1
225B 3E25       08250           LD      A,25H           ;"Illegal access...
225D B7         08260           OR      A
225E C9         08270           RET
                08280 ;
                08290 ;         User has access to rename - locate drivespec
                08300 ;
225F E5         08310 REN1      PUSH    HL              ;Save start
2260 7E         08320 REN2      LD      A,(HL)          ;P/u char of new spec
2261 23         08330           INC     HL
2262 FE0D       08340           CP      CR
2264 2808       08350           JR      Z,REN3          ;Go on ENTER
2266 FE03       08360           CP      3
2268 2804       08370           JR      Z,REN3          ;Go on ETX
226A FE3A       08380           CP      ':'
226C 20F2       08390           JR      NZ,REN2         ;Loop on colon
226E 2B         08400 REN3      DEC     HL              ;Backup to where the
226F 363A       08410           LD      (HL),':'        ;  colon should go
2271 23         08420           INC     HL              ;  & force the drivespec
2272 DD7E06     08430           LD      A,(IX+6)        ;  to the same as "old"
2275 4F         08440           LD      C,A             ;Keep drive spec in C
2276 E607       08450           AND     7
2278 C630       08460           ADD     A,'0'           ;Make it an ASCII digit
227A 77         08470           LD      (HL),A
227B 23         08480           INC     HL
227C 360D       08490           LD      (HL),CR
227E DD4607     08500           LD      B,(IX+7)        ;Get DEC
2281 DDE1       08510           POP     IX              ;Put "new" FCB into IX
2283 C5         08520           PUSH    BC              ;  & save DEC & drive
2284 217C00     08530           LD      HL,SFLAG$       ;Set don't test flags
2287 CBC6       08540           SET     0,(HL)
2289 CD441F     08550           CALL    OPEN1           ;Open the "new" spec
228C C1         08560           POP     BC
228D 2004       08570           JR      NZ,REN4         ;Should error here
228F 3E13       08580 REN3A     LD      A,19            ;  or else return
2291 B7         08590           OR      A               ;  if "new" is existing
2292 C9         08600           RET                     ;  & we opened it
```

```
2293 FE18      08610 REN4    CP      24              ;If not "file not found"
2295 C0        08620         RET     NZ              ;  then is error
2296 CDBB18    08630         CALL    @DIRRD          ;Read "old's" directory
2299 C0        08640         RET     NZ
229A C5        08650         PUSH    BC              ;Save drive spec
229B 54        08660         LD      D,H             ;Xfer buffer hi
229C 7D        08670         LD      A,L
229D C605      08680         ADD     A,5             ;Pt to filename field
229F 5F        08690         LD      E,A             ;Set buffer lo
22A0 21E922    08700         LD      HL,NAME$EXT     ;Point to where the
22A3 010B00    08710         LD      BC,11           ;  new name is stored
22A6 EDB0      08720         LDIR                    ;Move in new name
22A8 C1        08730         POP     BC
22A9 CD0318    08740         CALL    @DIRWR          ;Rewrite the directory
22AC CC9718    08750         CALL    Z,@HITRD        ;Read the HIT
22AF C0        08760         RET     NZ
22B0 54        08770         LD      D,H             ;Set the buffer hi
22B1 58        08780         LD      E,B             ;Set the exact HIT lo
22B2 21E922    08790         LD      HL,NAME$EXT     ;This doesn't change C
22B5 CD1E1E    08800         CALL    HASHNAME        ;Hash the new name
22B8 12        08810         LD      (DE),A          ;Stuff code into HIT
22B9 C39818    08820         JP      @HITWR          ;Rewrite & exit
               08830 ;
               08840 ;       Routine to rename a device
               08850 ;
22BC E5        08860 RENDEV  PUSH    HL              ;Save new pointer
22BD CD591E    08870         CALL    GETDCB          ;Locate old in tables
22C0 DDE1      08880         POP     IX              ;Recover pointer to "new"
22C2 C0        08890         RET     NZ              ;Back if not in tables
22C3 7D        08900         LD      A,L
22C4 FE31      08910         CP      DCBKL$          ;Ck if protected device
22C6 3E28      08920         LD      A,40            ;"Protected system device
22C8 D8        08930         RET     C
22C9 DD7E00    08940         LD      A,(IX+0)        ;"new" must be a device
22CC FE2A      08950         CP      '*'
22CE 20BF      08960         JR      NZ,REN3A        ;"illegal file name...
22D0 E5        08970         PUSH    HL              ;Save address of "old"
22D1 CD591E    08980         CALL    GETDCB          ;Ck if "new" is unused
22D4 E1        08990         POP     HL              ;Rcvr address of "old"
22D5 28B8      09000         JR      Z,REN3A
22D7 010600    09010         LD      BC,6            ;Point to name field
22DA 09        09020         ADD     HL,BC           ;  of "old" device
22DB 73        09030         LD      (HL),E          ;Stuff new name into
22DC 23        09040         INC     HL              ;  device control block
22DD 72        09050         LD      (HL),D
22DE AF        09060         XOR     A               ;Set Z-flag
22DF C9        09070         RET
               09080 ;
               09090 ;       Parameter storage area
               09100 ;
0001           09110 FILEHASH        DS      1
0008           09120 PSWDBUF DS      8
0008           09130 NAME$EXT        DS      8
0003           09140 FILE$EXT        DS      3
0002           09150 PW$HASH1        DS      2
0002           09160 PW$HASH2        DS      2
22F8 0000      09170         DW      0               ;ERN init
22FA 00        09180 DIR$INIT        DB      0,0,0,0
     00 00 00
0001           09190 LREC$   DS      1
22FF           09200 LAST    EQU     $
```

```
               09210          IFGT     $,DIRBUF$
               09220          ERR      'Module too big'
               09230          ENDIF
23FE           09240          ORG      MAXCOR$-2
23FE FF04      09250          DW       LAST-SYS2       ;Overlay length
               09260 ;
1E00           09270          END      SYS2
```

```
$A1             03B7 $A2             03B8 $A3             03B9
$CKEOF          1470 @$SYS           08F0 @@1             0000
@@2             0000 @@3             0000 @@4             0000
@ABORT          1B08 @ADTSK          1CDA @BANK           0877
@BKSP           1486 @BREAK          196F @BYTEIO         1300
@CHNIO          0689 @CKBRKC         0553 @CKDRV          1993
@CKEOF          158F @CKTSK          1CF5 @CLOSE          1999
@CLS            0545 @CMNDI          197E @CMNDR          197B
@CTL            0623 @DATE           07A8 @DBGHK          199F
@DCINIT         19C0 @DCRES          19C4 @DCSTAT         19B5
@DCTBYT         1A2B @DEBUG          19A0 @DECHEX         03E1
@DIRCYL         18F7 @DIRRD          18BB @DIRWR          1803
@DIV16          06E3 @DIV8           1927 @DODIR          19AF
@DOKEY          19A9 @DSP            0642 @DSPLY          052D
@ERROR          1B0F @EXIT           1B0B @FEXT           1984
@FLAGS          196A @FNAME          199C @FRENCH         0000
@FSPEC          1981 @GATRD          1874 @GATWR          1875
@GERMAN         0000 @GET            0638 @GTDCB          1990
@GTDCT          1A1E @GTMOD          19B2 @HDFMT          19E4
@HEX16          07BD @HEX8           07C2 @HEXDEC         06F6
@HIGH$          1948 @HITRD          1897 @HITWR          1898
@HZ50           0000 @ICNFG          0086 @INIT           198D
@INTL           0000 @IPL            1BF2 @JCL            0630
@KBD            0635 @KEY            0628 @KEYIN          0585
@KITSK          0089 @KLTSK          1CD0 @LOAD           1B38
@LOC            14B3 @LOF            14DE @LOGER          0503
@LOGOT          0500 @MOD2           0000 @MOD4           FFFF
@MSG            0530 @MUL16          06C9 @MUL8           190A
@NMI            0066 @OPEN           198A @OPREG          0084
@PARAM          1987 @PAUSE          0382 @PEOF           14A2
@POSN           1434 @PRINT          0528 @PRT            063D
@PUT            0645 @RAMDIR         19AC @RDHDR          19D8
@RDSEC          19F4 @RDSSC          18D8 @RDTRK          19E0
@READ           1513 @REMOVE         19A6 @RENAME         1996
@REW            149B @RMTSK          1CD7 @RPTSK          1CEB
@RREAD          1473 @RSLCT          19D4 @RST00          0000
@RST08          0008 @RST10          0010 @RST18          0018
@RST20          0020 @RST28          0028 @RST30          0030
@RST38          0038 @RSTNMI         0FE9 @RSTOR          19C8
@RSTREG         0680 @RUN            1B1D @RWRIT          13AD
@SEEK           19D0 @SEEKSC         1421 @SKIP           1430
@SLCT           19BC @SOUND          0392 @STEPI          19CC
@TIME           078D @USA            FFFF @VDCTL          0B99
@VDCTL3         0D38 @VER            1560 @VRSEC          19DC
@WEOF           14EC @WHERE          1979 @WRITE          1531
@WRSEC          19E8 @WRSSC          19EC @WRTRK          19F0
@_VDCTL         0D42 ADDR_2_ROWCOL   0DF1 AFLAG$          006A
AUTO?           1FF1 BAR$            0201 BOOTST$         439D
BREAK?          1C60 BRKVEC$         1C88 BUR$            0200
CASHK$          0A7B CDCNT           1ED4 CFCB$           00E0
CFGFCB$         00E0 CFLAG$          006C CKDR1           1EC9
CKDR2           1ED6 CKDR2A          1EDF CKDR2B          1EE4
CKDR3           1F06 CKDR3A          1F07 CKDR4           1F11
CKDR5           1F14 CKDR7           1F22 CKDR7A          1F23
CKDRV           1E7A CKDRV1          1E9D CKDRV1A         1EA7
CKMOD@          1A7F CKOPEN@         1568 CONFIG$         203F
CORE$           0300 CR              000D CRTBGN$         F800
CYL_GRN         16AE D@FBYT8         1A26 DATE$           0033
DAYTBL$         04C7 DBGSV$          00A0 DCBKL$          0031
DCT$            0470 DCTBYT8@        1A29 DCTFLD@         1A34
```

```
DEV1            1E62 DEV2            1E72 DEV4            1F2A
DEVOPEN         1F26 DFLAG$          006D DIR$INIT        22FA
DIRBUF$         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
DTPMT$          04C2 DVREND$         0FF4 DVRHI$          0206
EFLAG$          006E ENADIS_DO_RAM   0817 ERR15           2151
ERR26           214E ERR32           2154 EXTDBG$         19A4
FDDINT$         000E FEMSK$          006F FILE$EXT        22F1
FILEHASH        22E0 FLGTAB$         006A FRCREAD         208E
GETDCB          1E59 GET_@_ROWCOL    0DAE GSH3            2223
GSHLOOP         221B GSHOK           222D GSHTRY          2220
GTDCB           1E5F HASHNAME        1E1E HASHPSWD        1E2E
HERTZ$          0750 HIGH$           040E HKRES$          1A6C
HNAME1          1E21 HNAME2          1E2A HPSWD1          1E38
IFLAG$          0072 INBUF$          0420 INDEX           1F17
INIT            2106 INIT1           2128 INIT2           213A
INIT2A          2149 INIT4           2158 INIT5           2179
INTIM$          003C INTMSK$         003D INTRON          1EC3
INTVC$          003E JCLCB$          0203 JDCB$           0024
JFCB$           00C0 JLDCB$          0230 JRET$           0026
KCK@            07D6 KFLAG$          0074 KIDATA$         08FC
KIDCB$          0208 LAST            22FF LBANK$          0202
LDRV$           0023 LFLAG$          0075 LNKFCB@         1566
LOW$            001E LREC$           22FE LSVC$           000D
MAXCOR$         2400 MAXDAY$         0401 MINCOR$         3000
MODOUT$         0076 MONTBL$         04DC NAME$EXT        22E9
NFLAG$          0077 ONESID          220D OPEN            1F41
OPEN1           1F44 OPEN10          1FBE OPEN11          1FD1
OPEN12          2018 OPEN13          201E OPEN14          2023
OPEN15          2030 OPEN16          2035 OPEN2           1F76
OPEN3           1F7D OPEN4           1F86 OPEN5           1F92
OPEN6           1F95 OPEN7           1F9A OPEN8           1FA9
OPEN9           1FB3 OPNCB           203A OPNCB0          2093
OPNCB1          20A2 OPNCB10         2084 OPNCB2          20A8
OPNCB3          20AA OPNCB4          20B2 OPNCB5          20DD
OPNCB6          20FC OPNCB7          2100 OPNCB8          2064
OPNCB9          206D OPNEX           204C OPNEX1          204F
OPNEX2          2057 OPREG$          0078 OPREG_SV_AREA   086E
OPREG_SV_PTR    0835 ORARET@         14DC OSRLS$          003B
OSVER$          0085 OVRLY$          0069 PAKNAM$         0410
PAUSE@          0382 PCSAVE$         07AF PDRV$           001B
PHIGH$          001C PRDCB$          0218 PSWDBUF         22E1
PUTA@DE         0DCD PUT_@           0DCA PUT_@_ROWCOL    0DC6
PW$HASH1        22F4 PW$HASH2        22F6 REN0            2232
REN1            225F REN2            2260 REN3            226E
REN3A           228F REN4            2293 RENAME          2239
RENDEV          22BC RFLAG$          007B ROWCOL_2_ADDR   0DD0
RST38@          1BFF RSTOR$          04C4 RWRIT@          13A2
S1DCB$          0238 SBUFF$          1D00 SET@EXEC        1A79
SET_SCROLL      0CF3 SFCB$           008C SFLAG$          007C
SIDCB$          0220 SODCB$          0228 SPACE4$         2142
SPRHIT          21EE STACK$          0380 START$          0000
SVCRET$         000B SVCTAB$         0100 SYS2            1E00
SYSERR$         1B13 TCB$            004E TESTDRV         1F9E
TFLAG$          007D TIME$           002D TIMER$          002C
TIMSL$          002B TIMTSK$         0713 TMPMT$          04C3
TRACE_INT       07B1 TYPHK$          0A8F TYPTSK$         0B26
USEPWD          2002 USTOR$          0013 VFLAG$          007F
```

```
WASMAT          2006 WRINT$          0080 XFRSPEC         2187
XSPEC1          218E XSPEC10         21E1 XSPEC11         21E7
XSPEC3          21A5 XSPEC6          21C7 XSPEC7          21CE
XSPEC8          21D2 XSPEC8A         21D4 XSPEC9          21D8
ZERO$           0401 ZEROA@          13A0
00000 Total errors
```

NOTES:

NOTES:

SYS3/SYS

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.

```
              00100 ;SYS3/ASM - LS-DOS 6.2
0000          00110         TITLE   <SYS3 - LS-DOS 6.2>
              00120 ;
              00130 *LIST   OFF                     ;Get SYS0/EQU
              00150 *LIST   ON
000A          00160 LF      EQU     10
000D          00170 CR      EQU     13
              00180 ;
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 SYS3    AND     70H
1E02 C8       00240         RET     Z               ;Back on zero entry
1E03 FE10     00250         CP      10H
1E05 2806     00260         JR      Z,CLOSE         ;Jump if close
1E07 FE20     00270         CP      20H
1E09 CA2420   00280         JP      Z,FNAME         ;Jump if filespec recover
1E0C C9       00290         RET
1E0D 1A       00300 CLOSE   LD      A,(DE)          ;Test for device
1E0E CB7F     00310         BIT     7,A
1E10 CA5D20   00320         JP      Z,CLOSDEV       ;Jump if closing device
1E13 CD6815   00330         CALL    CKOPEN@         ;Test for open file
1E16 DD4E06   00340         LD      C,(IX+6)        ;P/u drive #
              00350 ;
              00360 ;       Special MINI check drive routine
              00370 ;
1E19 FDE5     00380         PUSH    IY              ;Save IY
1E1B CD1E1A   00390         CALL    @GTDCT          ;Pick up DCT for drive
1E1E CDD419   00400 CKAGN   CALL    @RSLCT          ;Wait until not busy
1E21 C28820   00410         JP      NZ,HOLDUP       ;Go to error handler
1E24 FDCB035E 00420         BIT     3,(IY+3)        ;If hard drive, bypass
1E28 2024     00430         JR      NZ,SAWBLK
1E2A FDCB0466 00440         BIT     4,(IY+4)        ;If "ALIEN" by pass
1E2E 201E     00450         JR      NZ,SAWBLK
1E30 FDCB047E 00460         BIT     7,(IY+4)        ;Ck if CKDRV inhibit
1E34 2018     00470         JR      NZ,SAWBLK       ;Go if so
              00480 ;
              00490 ;       Test for diskette in drive (no index)
              00500 ;
1E36 D5       00510         PUSH    DE
1E37 FD5605   00520         LD      D,(IY+5)        ;P/u current track
1E3A 1E00     00530         LD      E,0             ;Set to sector 0
1E3C CDD019   00540         CALL    @SEEK           ;Do a command
1E3F D1       00550         POP     DE
1E40 0630     00560         LD      B,30H           ;Set up count (short)
1E42 CDD419   00570 BLACK   CALL    @RSLCT          ;Check for index pulse
1E45 CB4F     00580         BIT     1,A             ;Test index
1E47 2805     00590         JR      Z,SAWBLK        ;Saw black, seems OK
1E49 10F7     00600         DJNZ    BLACK
1E4B C38820   00610         JP      HOLDUP          ;Close fault handler
              00620 ;
              00630 ;       Diskette is there, let's continue
              00640 ;
1E4E FDE1     00650 SAWBLK  POP     IY              ;Restore IY
1E50 DD4607   00660         LD      B,(IX+7)        ;P/u DEC of fpde
1E53 CDBB18   00670         CALL    @DIRRD          ;Read the directory
```

```
1E56 C0        00680          RET     NZ               ;Quit if error there
1E57 CB66      00690          BIT     4,(HL)           ;Ck for killed file
1E59 C8        00700          RET     Z                ;Quit if killed file
1E5A E5        00710          PUSH    HL
1E5B C5        00720          PUSH    BC
1E5C CDA213    00730          CALL    RWRIT@           ;Write last buffer?
1E5F C1        00740          POP     BC
1E60 E1        00750          POP     HL
1E61 C0        00760          RET     NZ               ;Ret on i/o error
1E62 DDCB0076  00770          BIT     6,(IX+0)         ;If user does not have
1E66 CACD1F    00780          JP      Z,RCVN0          ;  close authority...
1E69 2C        00790          INC     L                ;  else reset possible
1E6A CBAE      00800          RES     5,(HL)           ;  file-open bit in DIR+1
1E6C 2C        00810          INC     L                ;Determine if the EOF
1E6D 2C        00820          INC     L                ;  byte has changed
1E6E DD7E08    00830          LD      A,(IX+8)         ;P/u EOF byte offset
1E71 E5        00840          PUSH    HL               ;Save ptr to DIR+3
1E72 BE        00850          CP      (HL)
1E73 2014      00860          JR      NZ,CLOS1         ;Go if moved
1E75 3E11      00870          LD      A,11H
1E77 85        00880          ADD     A,L
1E78 6F        00890          LD      L,A
1E79 DD7E0C    00900          LD      A,(IX+12)        ;P/u lo-order ERN
1E7C BE        00910          CP      (HL)
1E7D 200A      00920          JR      NZ,CLOS1         ;Go if moved
1E7F 2C        00930          INC     L
1E80 DD7E0D    00940          LD      A,(IX+13)        ;P/u hi-order ERN
1E83 BE        00950          CP      (HL)
1E84 2003      00960          JR      NZ,CLOS1         ;Go if moved
1E86 F1        00970          POP     AF
1E87 181A      00980          JR      CLOS2            ;Didn't move
               00990 ;
               01000 ;        Routine to change the 3-byte EOF marker
               01010 ;
1E89 E1        01020 CLOS1    POP     HL               ;Pop DIR+3
1E8A DD7E08    01030          LD      A,(IX+8)         ;Xfer the eof offset
1E8D 77        01040          LD      (HL),A
1E8E 3E11      01050          LD      A,11H
1E90 85        01060          ADD     A,L
1E91 6F        01070          LD      L,A
1E92 DD7E0C    01080          LD      A,(IX+12)        ;  and the ERN from the FB
1E95 77        01090          LD      (HL),A
1E96 2C        01100          INC     L
1E97 DD7E0D    01110          LD      A,(IX+13)        ;  to the DIR entry
1E9A 77        01120          LD      (HL),A
1E9B DDCB0056  01130          BIT     2,(IX+0)         ;If file was updated
1E9F 2010      01140          JR      NZ,CLOS3         ;  then update mod date
1EA1 1834      01150          JR      CLOS5            ;  else don't
               01160 ;
               01170 ;        Three-byte EOF marker did not change
               01180 ;
1EA3 DDCB0056  01190 CLOS2    BIT     2,(IX+0)         ;If file was updated
1EA7 2008      01200          JR      NZ,CLOS3         ;  then update mod date
1EA9 DDCB0076  01210          BIT     6,(IX+0)         ;If close authority then
1EAD 2028      01220          JR      NZ,CLOS5         ;  write back the DIR
1EAF 182C      01230          JR      CLOS6            ;  else continue
               01240 ;
               01250 ;        Routine to insert packed date into directory
               01260 ;
1EB1 E5        01270 CLOS3    PUSH    HL               ;Save ptr to DIR+21
1EB2 7D        01280          LD      A,L              ;Pt to start of dir rec
```

```
1EB3 E6E0     01290          AND    0E0H
1EB5 6F       01300          LD     L,A
1EB6 2C       01310          INC    L              ;Pt to DIR+1
1EB7 CBF6     01320          SET    6,(HL)         ;Set the MOD flag
1EB9 113300   01330          LD     DE,DATE$       ;Point to year
1EBC 1A       01340          LD     A,(DE)         ;If year = 0, then date
1EBD B7       01350          OR     A              ;  is 00/00/00
1EBE 2802     01360          JR     Z,$+4
1EC0 D650     01370          SUB    80             ;Offset from 1980
1EC2 C5       01380          PUSH   BC
1EC3 47       01390          LD     B,A            ;Year-80 -> regB
1EC4 13       01400          INC    DE             ;Point to day
1EC5 1A       01410          LD     A,(DE)         ;Shift day into 3-7 &
1EC6 07       01420          RLCA                  ;  merge the year into
1EC7 07       01430          RLCA                  ;  the lo-order bits
1EC8 07       01440          RLCA
1EC9 B0       01450          OR     B
1ECA 2C       01460          INC    L
1ECB 77       01470          LD     (HL),A         ;Store day/year
1ECC 2D       01480          DEC    L
1ECD 13       01490          INC    DE             ;Point to month
1ECE 1A       01500          LD     A,(DE)
1ECF 47       01510          LD     B,A
1ED0 7E       01520          LD     A,(HL)         ;P/u dir byte
1ED1 E6F0     01530          AND    0F0H           ;Strip old month
1ED3 B0       01540          OR     B              ;Merge month &
1ED4 77       01550          LD     (HL),A         ;  update the field
1ED5 C1       01560          POP    BC
1ED6 E1       01570 CLOS4    POP    HL             ;Rcvr DIR+21
1ED7 E5       01580 CLOS5    PUSH   HL
1ED8 CD0318   01590          CALL   @DIRWR         ;Write back DIR entry
1EDB E1       01600          POP    HL
1EDC C0       01610          RET    NZ
1EDD 2C       01620 CLOS6    INC    L              ;Pt to DIR+22 which is
1EDE E5       01630          PUSH   HL             ;  the 1st extent
1EDF 7D       01640          LD     A,L
1EE0 D615     01650          SUB    15H            ;Backup to DIR+1
1EE2 6F       01660          LD     L,A
1EE3 CB7E     01670          BIT    7,(HL)         ;Test if created
1EE5 E1       01680          POP    HL
1EE6 C2CD1F   01690          JP     NZ,RCVN0       ;Bypass if created
1EE9 110000   01700          LD     DE,0           ;Init gran counter
1EEC 7E       01710 CLOS7    LD     A,(HL)         ;P/u cyl indicator
1EED 2C       01720          INC    L              ;Pt to gran alloc
1EEE FEFE     01730          CP     0FEH           ;Extent in use?
1EF0 300C     01740          JR     NC,CLOS8       ;Jump if spare or FXDE
1EF2 7E       01750          LD     A,(HL)         ;P/u granule allocation
1EF3 2C       01760          INC    L              ;Pt to next extent
1EF4 E61F     01770          AND    1FH            ;Strip off # of grans &
1EF6 3C       01780          INC    A              ;  adjust for zero offset
1EF7 83       01790          ADD    A,E            ;Accumulate the number of
1EF8 5F       01800          LD     E,A            ;  grans in this extent
1EF9 30F1     01810          JR     NC,CLOS7       ;Any previous quantity
1EFB 14       01820          INC    D
1EFC 18EE     01830          JR     CLOS7
1EFE 200B     01840 CLOS8    JR     NZ,CLOS9       ;Found all grans in this
1F00 46       01850          LD     B,(HL)         ;  extent, ck for FXDE
1F01 CDBB18   01860          CALL   @DIRRD
1F04 C0       01870          RET    NZ
1F05 7D       01880          LD     A,L            ;Point to extents in FXDE
1F06 C616     01890          ADD    A,16H
```

```
1F08 6F        01900           LD     L,A
1F09 18E1      01910           JR     CLOS7          ;Go to continue count
               01920 ;
               01930 ;         Routine to determine need to deallocate
               01940 ;
1F0B E5        01950 CLOS9     PUSH   HL             ;Save ptr to last extent
1F0C DD6E0C    01960           LD     L,(IX+12)      ;P/u ending record #
1F0F DD660D    01970           LD     H,(IX+13)
1F12 3E08      01980           LD     A,8            ;Get # sectors/gran
1F14 CD2B1A    01990           CALL   @DCTBYT
1F17 E61F      02000           AND    1FH            ;Remove other data
1F19 F5        02010           PUSH   AF             ;Save the #
1F1A 85        02020           ADD    A,L            ;Round up to next
1F1B 6F        02030           LD     L,A            ;  higher gran
1F1C 3001      02040           JR     NC,CLOS10
1F1E 24        02050           INC    H
1F1F F1        02060 CLOS10    POP    AF             ;Rcvr # sectors/gran
1F20 3C        02070           INC    A              ;Adjust for division
1F21 CDE306    02080           CALL   @DIV16         ;Calculate # grans in use
1F24 AF        02090           XOR    A              ;Subtract the # of grans
1F25 EB        02100           EX     DE,HL          ;  used from the # of
1F26 ED52      02110           SBC    HL,DE          ;  grans allocated in the
1F28 EB        02120           EX     DE,HL          ;  directory, and move DE
1F29 E1        02130           POP    HL             ;Rcvr ptr to last extent
1F2A CACD1F    02140           JP     Z,RCVN0        ;Jump if same quantity
1F2D DACD1F    02150           JP     C,RCVN0        ;Jump if now more
               02160 ;
               02170 ;         Need to deallocate space
               02180 ;
1F30 CD7418    02190           CALL   @GATRD         ;Read GAT
1F33 C0        02200           RET    NZ
1F34 1841      02210           JR     BAKUP          ;B/u to last used extent
1F36 D5        02220 CLOS11    PUSH   DE             ;Sv count of excess grans
1F37 7E        02230           LD     A,(HL)         ;P/u alloc info
1F38 E6E0      02240           AND    0E0H           ;Get starting relative
1F3A 07        02250           RLCA                  ;  gran into reg-E
1F3B 07        02260           RLCA
1F3C 07        02270           RLCA
1F3D 5F        02280           LD     E,A
1F3E 7E        02290           LD     A,(HL)         ;# of contiguous grans
1F3F E61F      02300           AND    1FH            ;Remove unneeded data
1F41 83        02310           ADD    A,E            ;Calculate ending
1F42 5F        02320           LD     E,A            ;  relative gran #
1F43 3E08      02330           LD     A,8            ;P/u the # of grans
1F45 CD2B1A    02340           CALL   @DCTBYT        ;  per cylinder
1F48 07        02350           RLCA
1F49 07        02360           RLCA
1F4A 07        02370           RLCA
1F4B E607      02380           AND    7              ;Move into bits 0-2
1F4D 3C        02390           INC    A              ;Adjust for zero offset
1F4E 57        02400           LD     D,A            ;Save count
1F4F 3E04      02410           LD     A,4
1F51 CD2B1A    02420           CALL   @DCTBYT
1F54 CB6F      02430           BIT    5,A            ;2-sided disk?
1F56 7A        02440           LD     A,D            ;Rcvr count
1F57 2801      02450           JR     Z,$+3          ;Bypass if 1-sided
1F59 07        02460           RLCA                  ;Double count
1F5A CD2719    02470           CALL   @DIV8          ;A=quotient, E=remainder
1F5D 2D        02480           DEC    L              ;Pt to starting cylinder
1F5E 86        02490           ADD    A,(HL)         ;Bump cyl pointer by how
1F5F 57        02500           LD     D,A            ;  many excessive cyls to
```

```
1F60 E5       02510          PUSH    HL              ;  start from the rear
1F61 C5       02520          PUSH    BC
1F62 2623     02530          LD      H,DIRBUF$<-8    ;Pt to that cyl's GAT
1F64 6A       02540          LD      L,D
1F65 46       02550          LD      B,(HL)          ;P/u the GAT allocation
1F66 7B       02560          LD      A,E
1F67 CD7B20   02570          CALL    CALCBIT         ;Deallocate a gran
1F6A 70       02580          LD      (HL),B          ;Repl GAT byte
1F6B C1       02590          POP     BC
1F6C E1       02600          POP     HL
1F6D 2C       02610          INC     L               ;Repoint to alloc info
1F6E 35       02620          DEC     (HL)            ;Reduce by 1 gran
1F6F 7E       02630          LD      A,(HL)          ;Get info on contig gran
1F70 3C       02640          INC     A               ;Adj for zero offset
1F71 E61F     02650          AND     1FH             ;Strip unneeded
1F73 D1       02660          POP     DE              ;Rcvr excess gran count
1F74 1B       02670          DEC     DE              ;  and count down
1F75 2043     02680          JR      NZ,CLOS12       ;Go if extent still used
1F77 36FF     02690 BAKUP    LD      (HL),0FFH       ;  else extent is spare
1F79 2D       02700          DEC     L
1F7A 36FF     02710          LD      (HL),0FFH
1F7C 2D       02720          DEC     L
1F7D 7D       02730          LD      A,L             ;Chack if backed all the
1F7E E61F     02740          AND     1FH             ;  way thru this entry
1F80 FE15     02750          CP      15H
1F82 2036     02760          JR      NZ,CLOS12       ;Go if not
1F84 AD       02770          XOR     L               ;Deallocate this FXDE
1F85 6F       02780          LD      L,A
1F86 CB7E     02790          BIT     7,(HL)          ;Was it the FPDE?
1F88 2830     02800          JR      Z,CLOS12        ;Bypass if FPDE
1F8A 3600     02810          LD      (HL),0          ;Show dir is spare
1F8C CD0318   02820          CALL    @DIRWR          ;Write back
1F8F C0       02830          RET     NZ
1F90 78       02840          LD      A,B             ;P/u deallocated DEC
1F91 E6E0     02850          AND     0E0H
1F93 3C       02860          INC     A               ;Pt to DIR+1
1F94 6F       02870          LD      L,A
1F95 7E       02880          LD      A,(HL)          ;P/u previous DEC
1F96 32A51F   02890          LD      (STUFDEC+1),A   ;Save in instruction
1F99 CD9718   02900          CALL    @HITRD          ;Read the HIT
1F9C C0       02910          RET     NZ
1F9D 68       02920          LD      L,B             ;Point to deallocated HIT
1F9E 3600     02930          LD      (HL),0          ;Deallocate space in HIT
1FA0 CD9818   02940          CALL    @HITWR          ;Write back
1FA3 C0       02950          RET     NZ
1FA4 0600     02960 STUFDEC  LD      B,0             ;P/u previous DEC
1FA6 CDBB18   02970          CALL    @DIRRD          ;Read its dir entry
1FA9 C0       02980          RET     NZ
1FAA 78       02990          LD      A,B
1FAB F61F     03000          OR      1FH             ;Pt to end of entry
1FAD 6F       03010          LD      L,A
1FAE 36FF     03020          LD      (HL),0FFH       ;Erase pointer
1FB0 2D       03030          DEC     L               ;  to deallocated FXDE
1FB1 36FF     03040          LD      (HL),0FFH
1FB3 2D       03050          DEC     L               ;Point to previous extent
1FB4 E5       03060          PUSH    HL              ;Save pointer
1FB5 CD0318   03070          CALL    @DIRWR          ;Write back
1FB8 E1       03080          POP     HL
1FB9 C0       03090          RET     NZ
1FBA 7A       03100 CLOS12   LD      A,D             ;Loop if still more to
1FBB B3       03110          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 DD36002A   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             03B7 $A2             03B8 $A3             03B9
$CKEOF          1470 @$SYS           08F0 @@1             0000
@@2             0000 @@3             0000 @@4             0000
@ABORT          1B08 @ADTSK          1CDA @BANK           0877
@BKSP           1486 @BREAK          196F @BYTEIO         1300
@CHNIO          0689 @CKBRKC         0553 @CKDRV          1993
@CKEOF          158F @CKTSK          1CF5 @CLOSE          1999
@CLS            0545 @CMNDI          197E @CMNDR          197B
@CTL            0623 @DATE           07A8 @DBGHK          199F
@DCINIT         19C0 @DCRES          19C4 @DCSTAT         19B5
@DCTBYT         1A2B @DEBUG          19A0 @DECHEX         03E1
@DIRCYL         18F7 @DIRRD          18BB @DIRWR          1803
@DIV16          06E3 @DIV8           1927 @DODIR          19AF
@DOKEY          19A9 @DSP            0642 @DSPLY          052D
@ERROR          1B0F @EXIT           1B0B @FEXT           1984
@FLAGS          196A @FNAME          199C @FRENCH         0000
@FSPEC          1981 @GATRD          1874 @GATWR          1875
@GERMAN         0000 @GET            0638 @GTDCB          1990
@GTDCT          1A1E @GTMOD          19B2 @HDFMT          19E4
@HEX16          07BD @HEX8           07C2 @HEXDEC         06F6
@HIGH$          1948 @HITRD          1897 @HITWR          1898
@HZ50           0000 @ICNFG          0086 @INIT           198D
@INTL           0000 @IPL            1BF2 @JCL            0630
@KBD            0635 @KEY            0628 @KEYIN          0585
@KITSK          0089 @KLTSK          1CD0 @LOAD           1B38
@LOC            14B3 @LOF            14DE @LOGER          0503
@LOGOT          0500 @MOD2           0000 @MOD4           FFFF
@MSG            0530 @MUL16          06C9 @MUL8           190A
@NMI            0066 @OPEN           198A @OPREG          0084
@PARAM          1987 @PAUSE          0382 @PEOF           14A2
@POSN           1434 @PRINT          0528 @PRT            063D
@PUT            0645 @RAMDIR         19AC @RDHDR          19D8
@RDSEC          19F4 @RDSSC          18D8 @RDTRK          19E0
@READ           1513 @REMOVE         19A6 @RENAME         1996
@REW            149B @RMTSK          1CD7 @RPTSK          1CEB
@RREAD          1473 @RSLCT          19D4 @RST00          0000
@RST08          0008 @RST10          0010 @RST18          0018
@RST20          0020 @RST28          0028 @RST30          0030
@RST38          0038 @RSTNMI         0FE9 @RSTOR          19C8
@RSTREG         0680 @RUN            1B1D @RWRIT          13AD
@SEEK           19D0 @SEEKSC         1421 @SKIP           1430
@SLCT           19BC @SOUND          0392 @STEPI          19CC
@TIME           078D @USA            FFFF @VDCTL          0B99
@VDCTL3         0D38 @VER            1560 @VRSEC          19DC
@WEOF           14EC @WHERE          1979 @WRITE          1531
@WRSEC          19E8 @WRSSC          19EC @WRTRK          19F0
@_VDCTL         0D42 ABRT            20A1 ADDR_2_ROWCOL   0DF1
AFLAG$          006A AUTO?           1FF1 BAKUP           1F77
BAR$            0201 BLACK           1E42 BOOTST$         439D
BREAK?          1C60 BRKVEC$         1C88 BUR$            0200
CALC1           2085 CALCBIT         207B CASHK$          0A7B
CFCB$           00E0 CFGFCB$         00E0 CFLAG$          006C
CKAGN           1E1E CKMOD@          1A7F CKOPEN@         1568
CLOS1           1E89 CLOS10          1F1F CLOS11          1F36
CLOS12          1FBA CLOS13          1FC9 CLOS2           1EA3
CLOS3           1EB1 CLOS4           1ED6 CLOS5           1ED7
CLOS6           1EDD CLOS7           1EEC CLOS8           1EFE
CLOS9           1F0B CLOSDEV         205D CLOSE           1E0D
CONFIG$         203F CORE$           0300 CR              000D
CRTBGN$         F800 CYL_GRN         16AE D@FBYT8         1A26
```

```
DATE$           0033 DAYTBL$        04C7 DBGSV$          00A0
DCBKL$          0031 DCT$           0470 DCTBYT8@        1A29
DCTFLD@         1A34 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 DTPMT$          04C2
DVREND$         0FF4 DVRHI$         0206 EFLAG$          006E
ENADIS_DO_RAM   0817 EXTDBG$        19A4 FDDINT$         000E
FEMSK$          006F FLGTAB$        006A FNAM1           2052
FNAM2           205A FNAME          2024 GET_@_ROWCOL    0DAE
HERTZ$          0750 HIGH$          040E HKRES$          1A6C
HOLDUP          2088 HOLDUP$        20AE IFLAG$          0072
INBUF$          0420 INTIM$         003C INTMSK$         003D
INTVC$          003E JCLCB$         0203 JDCB$           0024
JFCB$           00C0 JLDCB$         0230 JRET$           0026
KCK@            07D6 KFLAG$         0074 KIDATA$         08FC
KIDCB$          0208 LAST           20F6 LBANK$          0202
LDRV$           0023 LF             000A LFLAG$          0075
LNKFCB@         1566 LOW$           001E LSVC$           000D
MAXCOR$         2400 MAXDAY$        0401 MINCOR$         3000
MODOUT$         0076 MONTBL$        04DC NFLAG$          0077
ONESID          2044 OPREG$         0078 OPREG_SV_AREA   086E
OPREG_SV_PTR    0835 ORARET@        14DC OSRLS$          003B
OSVER$          0085 OVRLY$         0069 PAKNAM$         0410
PAUSE@          0382 PCSAVE$        07AF PDRV$           001B
PHIGH$          001C PRDCB$         0218 PUTA@DE         0DCD
PUT_@           0DCA PUT_@_ROWCOL   0DC6 RCVN0           1FCD
RCVN1           1FF3 RCVN2          1FFD RCVN3           200D
RCVN4           2017 RCVN5          201B RCVNAM          1FE0
RFLAG$          007B ROWCOL_2_ADDR  0DD0 RST38@          1BFF
RSTOR$          04C4 RWRIT@         13A2 S1DCB$          0238
SAWBLK          1E4E SBUFF$         1D00 SET@EXEC        1A79
SET_SCROLL      0CF3 SFCB$          008C SFLAG$          007C
SIDCB$          0220 SODCB$         0228 SPACE4$         2142
STACK$          0380 START$         0000 STUFDEC         1FA4
SVCRET$         000B SVCTAB$        0100 SYS3            1E00
SYSERR$         1B13 TCB$           004E TFLAG$          007D
TIME$           002D TIMER$         002C TIMSL$          002B
TIMTSK$         0713 TMPMT$         04C3 TRACE_INT       07B1
TRYNOW          20A9 TYPHK$         0A8F TYPTSK$         0B26
USTOR$          0013 VFLAG$         007F WAITING         2093
WRINT$          0080 ZERO$          0401 ZEROA@          13A0

1E00 is the transfer address
00000 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 B0
               00540 ;                write fault on disk drive
1E45 0A        00550 MSG15     DB     10,22,19+80H
     16 93
               00560 ;                write protected disk
1E48 17        00570 MSG16     DB     23,24,26,25+80H
     18 1A 99
               00580 ;                illegal logical file number
1E4C 10        00590 MSG17     DB     16,9,2+80H
     09 82
               00600 ;                directory read error
1E4F 10        00610 MSG18     DB     16,10,2+80H
     0A 82
               00620 ;                directory write error
1E52 17        00630 MSG19     DB     23,26,41+0C0H
     1A E9
               00640 ;                illegal file name
1E55 22        00650 MSG20     DB     34,9,2+80H
     09 82
               00660 ;                gat read error
1E58 22        00670 MSG21     DB     34,10,2+80H
     0A 82
               00680 ;                gat write error
1E5B 23        00690 MSG22     DB     35,9,2+80H
     09 82
               00700 ;                hit read error
1E5E 23        00710 MSG23     DB     35,10,2+80H
     0A 82
               00720 ;                hit write error
1E61 1A        00730 MSG24     DB     26,12,45,16+0C0H
     0C 2D D0
               00740 ;                file not in directory
1E65 1A        00750 MSG25     DB     26,46,49+0C0H
     2E F1
               00760 ;                file access denied
1E68 01        00770 MSG26     DB     1,16,39,51+0C0H
     10 27 F3
               00780 ;                directory space full
1E6C 13        00790 MSG27     DB     19,39,47+80H
     27 AF
               00800 ;                disk space full
1E6F 1C        00810 MSG28     DB     28,29,26,32+80H
     1D 1A A0
               00820 ;                end of file encountered
1E73 1B        00830 MSG29     DB     27,25,30,29,31+80H
     19 1E 1D 9F
               00840 ;                record number out of range
1E78 10        00850 MSG30     DB     16,47,52,26+80H
     2F 34 9A
               00860 ;                directory full - can't extend file
1E7C 32        00870 MSG31     DB     50,12,44+0C0H
     0C EC
               00880 ;                program not found
1E7F 17        00890 MSG32     DB     23,48,25+0C0H
     30 D9
               00900 ;                illegal drive number
1E82 01        00910 MSG33     DB     1,42,39,51+0C0H
     2A 27 F3
               00920 ;                no device space available
1E86 26        00930 MSG34     DB     38,26,43,2+80H
     1A 2B 82
```

```
              00940 ;                load file format error
1E8A 11       00950 MSG35   DB       17,21+80H
     95
              00960 ;                memory fault
1E8C 0D       00970 MSG36   DB       13,38,9,40,17+80H
     26 09 28 91
              00980 ;                attempted to load read only memory
1E91 17       00990 MSG37   DB       23,46,13,22,26+80H
     2E 0D 16 9A
              01000 ;                illegal access attempted to protected file
1E96 1A       01010 MSG38   DB       26,12,53+0C0H
     0C F5
              01020 ;                file not open
1E99 2A       01030 MSG39   DB       42,45,54+80H
     2D B6
              01040 ;                device in use
1E9C 16       01050 MSG40   DB       22,15,42+80H
     0F AA
              01060 ;                protected system device
1E9F 1A       01070 MSG41   DB       26,57,53!0C0H
     39 F5
              01080 ;                file already open
1EA2 18       01090 MSG42   DB       24,27,58,53,21!0C0H
     1B 3A 35 D5
              01100 ;                logical record length open fault
1EA7 38       01110 MSG43   DB       56,20,2!80H
     14 82
              01120 ;                SVC parameter error
1EAA 14       01130 MSG44   DB       20,2!80H
     82
              01140 ;                Parameter error
1EAC 25       01150 MSG45   DB       37,2,33+80H
     02 A1
              01160 ;                unknown error code
1EAF E670     01170 BEGIN   AND      70H                ;What's the entry?
1EB1 C8       01180         RET      Z                  ;Back on zero
1EB2 F5       01190         PUSH     AF
1EB3 3A0D00   01200         LD       A,(LSVC$)          ;Grab the last SVC
1EB6 325E20   01210         LD       (SVSVC+1),A        ; and store for later
1EB9 F1       01220         POP      AF
1EBA 22461F   01230         LD       (EXTEND+1),HL      ;Value if extended error
1EBD E3       01240         EX       (SP),HL            ;Grab return address
1EBE 22BF1F   01250         LD       (ERR7+1),HL        ; & stuff it
1EC1 E1       01260         POP      HL
1EC2 F1       01270         POP      AF                 ;Pop off the error code
1EC3 E3       01280         EX       (SP),HL            ;Get user ret address
1EC4 22241F   01290         LD       (USRET+1),HL       ; for long dsply
1EC7 E3       01300         EX       (SP),HL
1EC8 E5       01310         PUSH     HL                 ;Save regs
1EC9 D5       01320         PUSH     DE
1ECA C5       01330         PUSH     BC
1ECB 2A0B00   01340         LD       HL,(SVCRET$)       ;Grab last SVC return
1ECE 227220   01350         LD       (SVRET+1),HL       ; and save for dsply
1ED1 47       01360         LD       B,A
1ED2 3A7C00   01370         LD       A,(SFLAG$)         ;Test expanded-error
1ED5 E640     01380         AND      40H                ; flag bit in system flag
1ED7 A8       01390         XOR      B
1ED8 A0       01400         AND      B
1ED9 47       01410         LD       B,A                ;Xfer the result to B
1EDA F5       01420         PUSH     AF                 ; & save for later
1EDB E63F     01430         AND      3FH                ;Strip all but error #
```

```
1EDD 4F        01440         LD      C,A             ;Place error code -> C
1EDE 216C00    01450         LD      HL,CFLAG$       ;If system error suppress
1EE1 CB76      01460         BIT     6,(HL)          ;  flag is set, don't
1EE3 C2B91F    01470         JP      NZ,ERR6A        ;  display error message.
1EE6 CB7E      01480         BIT     7,(HL)          ;If error-to-buffer is
1EE8 2005      01490         JR      NZ,ERR0         ;  set, put to user buf
1EEA 11001D    01500         LD      DE,SBUFF$
1EED 1806      01510         JR      ERR0A           ;Branch around force
1EEF CBF0      01520 ERR0    SET     6,B             ;Force buffer to abbrev
1EF1 F1        01530         POP     AF
1EF2 CBF7      01540         SET     6,A
1EF4 F5        01550         PUSH    AF
1EF5 CB70      01560 ERR0A   BIT     6,B             ;Expanded error display?
1EF7 0600      01570         LD      B,0
1EF9 2044      01580         JR      NZ,ERR2         ;Jump if abbreviated
1EFB C5        01590         PUSH    BC
1EFC 21C320    01600         LD      HL,ERRMSG       ;Pt to "< ERRCOD =...
1EFF 0E11      01610         LD      C,MLEN          ;  & move to buffer
1F01 EDB0      01620         LDIR
1F03 C1        01630         POP     BC
1F04 EB        01640         EX      DE,HL           ;Buffer ptr to HL
1F05 79        01650         LD      A,C             ;Error code to A
1F06 362F      01660         LD      (HL),2FH        ;Init for digit conv
1F08 34        01670 ERR1    INC     (HL)            ;Bump ASCII digit
1F09 D60A      01680         SUB     10              ;  count by 10
1F0B 30FB      01690         JR      NC,ERR1         ;Keep bumping 10's digit
1F0D 2C        01700         INC     L               ;Bump buffer ptr
1F0E C63A      01710         ADD     A,3AH           ;Convert rmndr to unit's
1F10 77        01720         LD      (HL),A          ;  & place in buffer
1F11 2C        01730         INC     L               ;Bump to next pos
1F12 362C      01740         LD      (HL),','        ;Stuff a comma & bump
1F14 2C        01750         INC     L
1F15 3620      01760         LD      (HL),' '        ;  & a space
1F17 2C        01770         INC     L
1F18 EB        01780         EX      DE,HL           ;Buffer ptr back to DE
1F19 C5        01790         PUSH    BC
1F1A 21D420    01800         LD      HL,ERRMSG1      ;"Returns to X'"
1F1D 010D00    01810         LD      BC,M1LEN
1F20 EDB0      01820         LDIR
1F22 EB        01830         EX      DE,HL           ;HL back to buffer
1F23 110000    01840 USRET   LD      DE,$-$          ;User ret address
1F26 CDBD07    01850         CALL    @HEX16
1F29 3E27      01860         LD      A,27H           ;"'"
1F2B 77        01870         LD      (HL),A
1F2C 23        01880         INC     HL
1F2D 360A      01890         LD      (HL),LF         ;End the line
1F2F 23        01900         INC     HL
1F30 C1        01910         POP     BC
1F31 CB71      01920         BIT     6,C             ;Extended error?
1F33 2009      01930         JR      NZ,ERR6         ;Go if not
1F35 362A      01940         LD      (HL),'*'        ;Make long msg look nice
1F37 23        01950         INC     HL
1F38 362A      01960         LD      (HL),'*'
1F3A 23        01970         INC     HL
1F3B 3620      01980         LD      (HL),' '
1F3D 23        01990         INC     HL
1F3E EB        02000 ERR6    EX      DE,HL           ;DE back to nxt buff line
1F3F 79        02010 ERR2    LD      A,C
1F40 FE3F      02020         CP      63              ;"Extended error"?
1F42 2023      02030         JR      NZ,ERR2A
               02040 ;
               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 21001D     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$FCB DB      'Open FCB, Drive=n, DEC=   ',CR
     70 65 6E 20 46 43 42 2C
     20 44 72 69 76 65 3D 6E
     2C 20 44 45 43 3D 20 20
     20 0D
2128 4F          04320 OPN$DCB DB      'Open DCB, Device=*xx',CR
     70 65 6E 20 44 43 42 2C
     20 44 65 76 69 63 65 3D
     2A 78 78 0D
213D 55          04330 UNK$TYP DB      'Unknown FCB/DCB',CR
     6E 6B 6E 6F 77 6E 20 46
     43 42 2F 44 43 42 0D
214D 4C          04340 SVC$NUM DB      'Last SVC = nnn',3
     61 73 74 20 53 56 43 20
     3D 20 6E 6E 6E 03
215C 2C          04350 SVC$RET DB      ', Returned to X',27H,'xxxx',27H,CR
     20 52 65 74 75 72 6E 65
     64 20 74 6F 20 58 27 78
     78 78 78 27 0D
                 04360 ;
0005             04370 LILBUF  DS      5
2177 00          04380         DB      0
                 04390 ;
                 04400 ;       Table points to low-order bytes of messages
                 04410 ;
2178 03          04420 CODTAB  DB      MSG0&0FFH,MSG1&0FFH,MSG2&0FFH,MSG3&0FFH
     05 0A 0E
217C 12          04430         DB      MSG4&0FFH,MSG5&0FFH,MSG6&0FFH
     16 1C
217F 21          04440         DB      MSG7&0FFH,MSG8&0FFH,MSG9&0FFH
     26 29
2182 2E          04450         DB      MSG10&0FFH,MSG11&0FFH,MSG12&0FFH,MSG13&0FFH
     32 36 3A
2186 40          04460         DB      MSG14&0FFH,MSG15&0FFH,MSG16&0FFH,MSG17&0FFH
     45 48 4C
218A 4F          04470         DB      MSG18&0FFH,MSG19&0FFH,MSG20&0FFH,MSG21&0FFH
     52 55 58
218E 5B          04480         DB      MSG22&0FFH,MSG23&0FFH,MSG24&0FFH,MSG25&0FFH
     5E 61 65
2192 68          04490         DB      MSG26&0FFH,MSG27&0FFH,MSG28&0FFH,MSG29&0FFH
     6C 6F 73
2196 78          04500         DB      MSG30&0FFH,MSG31&0FFH,MSG32&0FFH,MSG33&0FFH
     7C 7F 82
219A 86          04510         DB      MSG34&0FFH,MSG35&0FFH,MSG36&0FFH,MSG37&0FFH
     8A 8C 91
219E 96          04520         DB      MSG38&0FFH,MSG39&0FFH,MSG40&0FFH,MSG41&0FFH
     99 9C 9F
21A2 A2          04530         DB      MSG42&0FFH,MSG43&0FFH,MSG44&0FFH,MSG45&0FFH
     A7 AA AC
                 04540 ;
                 04550 ;       Word dictionary
                 04560 ;
21A6 D2          04570 WORDS   DB      'R'!80H                 ;Start table with bit 7
21A7 6E          04580         DB      'n','o'!80H             ;1
     EF
21A9 65          04590         DB      'erro','r'!80H          ;2
     72 72 6F F2
21AE EF          04600         DB      'o'!80H                 ;3 extra word
21AF 70          04610         DB      'parit','y'!80H         ;4
     61 72 69 74 F9
21B5 64          04620         DB      'durin','g'!80H         ;5
     75 72 69 6E E7
21BB 68          04630         DB      'heade','r'!80H         ;6
     65 61 64 65 F2
```

```
21C1 64         04640          DB        'dat','a'!80H           ;7
     61 74 E1
21C5 73         04650          DB        'see','k'!80H           ;8
     65 65 EB
21C9 72         04660          DB        'rea','d'!80H           ;9
     65 61 E4
21CD 77         04670          DB        'writ','e'!80H          ;10
     72 69 74 E5
21D2 6C         04680          DB        'los','t'!80H           ;11
     6F 73 F4
21D6 6E         04690          DB        'no','t'!80H            ;12
     6F F4
21D9 61         04700          DB        'attempted t','o'!80H   ;13
     74 74 65 6D 70 74 65 64
     20 74 EF
21E5 6C         04710          DB        'locked/delete','d'!80H ;14
     6F 63 6B 65 64 2F 64 65
     6C 65 74 65 E4
21F3 73         04720          DB        'syste','m'!80H         ;15
     79 73 74 65 ED
21F9 64         04730          DB        'director','y'!80H      ;16
     69 72 65 63 74 6F 72 F9
2202 6D         04740          DB        'memor','y'!80H         ;17
     65 6D 6F 72 F9
2208 6F         04750          DB        'o','n'!80H             ;18
     EE
220A 64         04760          DB        'dis','k'!80H           ;19
     69 73 EB
220E 70         04770          DB        'paramete','r'!80H      ;20
     61 72 61 6D 65 74 65 F2
2217 66         04780          DB        'faul','t'!80H          ;21
     61 75 6C F4
221C 70         04790          DB        'protecte','d'!80H      ;22
     72 6F 74 65 63 74 65 E4
2225 69         04800          DB        'illega','l'!80H        ;23
     6C 6C 65 67 61 EC
222C 6C         04810          DB        'logica','l'!80H        ;24
     6F 67 69 63 61 EC
2233 6E         04820          DB        'numbe','r'!80H         ;25
     75 6D 62 65 F2
2239 66         04830          DB        'fil','e'!80H           ;26
     69 6C E5
223D 72         04840          DB        'recor','d'!80H         ;27
     65 63 6F 72 E4
2243 65         04850          DB        'en','d'!80H            ;28
     6E E4
2246 6F         04860          DB        'o','f'!80H             ;29
     E6
2248 6F         04870          DB        'ou','t'!80H            ;30
     75 F4
224B 72         04880          DB        'rang','e'!80H          ;31
     61 6E 67 E5
2250 65         04890          DB        'encountere','d'!80H    ;32
     6E 63 6F 75 6E 74 65 72
     65 E4
225B 63         04900          DB        'cod','e'!80H           ;33
     6F 64 E5
225F 47         04910          DB        'GA','T'!80H            ;34
     41 D4
2262 48         04920          DB        'HI','T'!80H            ;35
     49 D4
```

```
2265 F9         04930           DB      'y'!80H                 ;36
2266 75         04940           DB      'unknow','n'!80H        ;37
     6E 6B 6E 6F 77 EE
226D 6C         04950           DB      'loa','d'!80H           ;38
     6F 61 E4
2271 73         04960           DB      'spac','e'!80H          ;39
     70 61 63 E5
2276 6F         04970           DB      'onl','y'!80H           ;40
     6E 6C F9
227A 6E         04980           DB      'nam','e'!80H           ;41
     61 6D E5
227E 64         04990           DB      'devic','e'!80H         ;42
     65 76 69 63 E5
2284 66         05000           DB      'forma','t'!80H         ;43
     6F 72 6D 61 F4
228A 66         05010           DB      'foun','d'!80H          ;44
     6F 75 6E E4
228F 69         05020           DB      'i','n'!80H             ;45
     EE
2291 61         05030           DB      'acces','s'!80H         ;46
     63 63 65 73 F3
2297 66         05040           DB      'ful','l'!80H           ;47
     75 6C EC
229B 64         05050           DB      'driv','e'!80H          ;48
     72 69 76 E5
22A0 64         05060           DB      'denie','d'!80H         ;49
     65 6E 69 65 E4
22A6 70         05070           DB      'progra','m'!80H        ;50
     72 6F 67 72 61 ED
22AD 61         05080           DB      'availabl','e'!80H      ;51
     76 61 69 6C 61 62 6C E5
22B6 2D         05090           DB      '- can''t exten','d'!80H        ;52
     20 63 61 6E 27 74 20 65
     78 74 65 6E E4
22C4 6F         05100           DB      'ope','n'!80H           ;53
     70 65 EE
22C8 75         05110           DB      'us','e'!80H            ;54
     73 E5
22CB 6F         05120           DB      'o','r'!80H             ;55
     F2
22CD 53         05130           DB      'SV','C'!80H            ;56
     56 C3
22D0 61         05140           DB      'alread','y'!80H        ;57
     6C 72 65 61 64 F9
22D7 6C         05150           DB      'lengt','h'!80H         ;58
     65 6E 67 74 E8
22DD            05160 LAST      EQU     $
                05170           IFGT    $,DIRBUF$
                05180           ERR     'Module too big'
                05190           ENDIF
23FE            05200           ORG     MAXCOR$-2
23FE DD04       05210           DW      LAST-SYS4       ;Overlay length
                05220 ;
1E00            05230           END     SYS4
```

```
$A1             03B7 $A2            03B8 $A3            03B9
$CKEOF          1470 @$SYS          08F0 @@1            0000
@@2             0000 @@3            0000 @@4            0000
@ABORT          1B08 @ADTSK         1CDA @BANK          0877
@BKSP           1486 @BREAK         196F @BYTEIO        1300
@CHNIO          0689 @CKBRKC        0553 @CKDRV         1993
@CKEOF          158F @CKTSK         1CF5 @CLOSE         1999
@CLS            0545 @CMNDI         197E @CMNDR         197B
@CTL            0623 @DATE          07A8 @DBGHK         199F
@DCINIT         19C0 @DCRES         19C4 @DCSTAT        19B5
@DCTBYT         1A2B @DEBUG         19A0 @DECHEX        03E1
@DIRCYL         18F7 @DIRRD         18BB @DIRWR         1803
@DIV16          06E3 @DIV8          1927 @DODIR         19AF
@DOKEY          19A9 @DSP           0642 @DSPLY         052D
@ERROR          1B0F @EXIT          1B0B @FEXT          1984
@FLAGS          196A @FNAME         199C @FRENCH        0000
@FSPEC          1981 @GATRD         1874 @GATWR         1875
@GERMAN         0000 @GET           0638 @GTDCB         1990
@GTDCT          1A1E @GTMOD         19B2 @HDFMT         19E4
@HEX16          07BD @HEX8          07C2 @HEXDEC        06F6
@HIGH$          1948 @HITRD         1897 @HITWR         1898
@HZ50           0000 @ICNFG         0086 @INIT          198D
@INTL           0000 @IPL           1BF2 @JCL           0630
@KBD            0635 @KEY           0628 @KEYIN         0585
@KITSK          0089 @KLTSK         1CD0 @LOAD          1B38
@LOC            14B3 @LOF           14DE @LOGER         0503
@LOGOT          0500 @MOD2          0000 @MOD4          FFFF
@MSG            0530 @MUL16         06C9 @MUL8          190A
@NMI            0066 @OPEN          198A @OPREG         0084
@PARAM          1987 @PAUSE         0382 @PEOF          14A2
@POSN           1434 @PRINT         0528 @PRT           063D
@PUT            0645 @RAMDIR        19AC @RDHDR         19D8
@RDSEC          19F4 @RDSSC         18D8 @RDTRK         19E0
@READ           1513 @REMOVE        19A6 @RENAME        1996
@REW            149B @RMTSK         1CD7 @RPTSK         1CEB
@RREAD          1473 @RSLCT         19D4 @RST00         0000
@RST08          0008 @RST10         0010 @RST18         0018
@RST20          0020 @RST28         0028 @RST30         0030
@RST38          0038 @RSTNMI        0FE9 @RSTOR         19C8
@RSTREG         0680 @RUN           1B1D @RWRIT         13AD
@SEEK           19D0 @SEEKSC        1421 @SKIP          1430
@SLCT           19BC @SOUND         0392 @STEPI         19CC
@TIME           078D @USA           FFFF @VDCTL         0B99
@VDCTL3         0D38 @VER           1560 @VRSEC         19DC
@WEOF           14EC @WHERE         1979 @WRITE         1531
@WRSEC          19E8 @WRSSC         19EC @WRTRK         19F0
@_VDCTL         0D42 ADDR_2_ROWCOL  0DF1 AFLAG$         006A
AUTO?           1FF1 BAR$           0201 BEGIN          1EAF
BOOTST$         439D BREAK?         1C60 BRKVEC$        1C88
BUR$            0200 CASHK$         0A7B CFCB$          00E0
CFGFCB$         00E0 CFLAG$         006C CHKASC         2083
CKASC1          208D CKASC2         2092 CKMOD@         1A7F
CKOPEN@         1568 CODTAB         2178 CONFIG$        203F
CORE$           0300 CR             000D CRTBGN$        F800
CYL_GRN         16AE D@FBYT8        1A26 DATE$          0033
DAYTBL$         04C7 DBGSV$         00A0 DCBKL$         0031
DCBUNK          1FF8 DCT$           0470 DCTBYT8@       1A29
DCTFLD@         1A34 DEV$EQ         20E1 DEV$NAM        20EB
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 DSPC1           1FDE DSPC1A          1FF2
DSPC2          1FFF DSPC3           201D DSPC3A          2033
DSPSPEC        1FC4 DTPMT$          04C2 DVREND$         0FF4
DVRHI$         0206 ED1             2097 EDEC            2094
EFLAG$         006E ENADIS_DO_RAM   0817 ERR0            1EEF
ERR0A          1EF5 ERR1            1F08 ERR2            1F3F
ERR2A          1F67 ERR3            1F6E ERR5            1F75
ERR6           1F3E ERR6A           1FB9 ERR7            1FBE
ERRMSG         20C3 ERRMSG1         20D4 EXT$ERR         20A3
EXTDBG$        19A4 EXTEND          1F45 FDDINT$         000E
FEMSK$         006F FILE$EQ         20EE FLGTAB$         006A
GET_@_ROWCOL   0DAE HERTZ$          0750 HIGH$           040E
HKRES$         1A6C IFLAG$          0072 INBUF$          0420
INTIM$         003C INTMSK$         003D INTVC$          003E
JCLCB$         0203 JDCB$           0024 JFCB$           00C0
JLDCB$         0230 JRET$           0026 KCK@            07D6
KFLAG$         0074 KIDATA$         08FC KIDCB$          0208
LAST           22DD LBANK$          0202 LDRV$           0023
LF             000A LFLAG$          0075 LILBUF          2172
LNKFCB@        1566 LOW$            001E LP1             1F7D
LP2            1F85 LSVC$           000D M1LEN           000D
M2LEN          0020 MAXCOR$         2400 MAXDAY$         0401
MINCOR$        3000 MLEN            0011 MODOUT$         0076
MONTBL$        04DC MSG0            1E03 MSG1            1E05
MSG10          1E2E MSG11           1E32 MSG12           1E36
MSG13          1E3A MSG14           1E40 MSG15           1E45
MSG16          1E48 MSG17           1E4C MSG18           1E4F
MSG19          1E52 MSG2            1E0A MSG20           1E55
MSG21          1E58 MSG22           1E5B MSG23           1E5E
MSG24          1E61 MSG25           1E65 MSG26           1E68
MSG27          1E6C MSG28           1E6F MSG29           1E73
MSG3           1E0E MSG30           1E78 MSG31           1E7C
MSG32          1E7F MSG33           1E82 MSG34           1E86
MSG35          1E8A MSG36           1E8C MSG37           1E91
MSG38          1E96 MSG39           1E99 MSG4            1E12
MSG40          1E9C MSG41           1E9F MSG42           1EA2
MSG43          1EA7 MSG44           1EAA MSG45           1EAC
MSG5           1E16 MSG6            1E1C MSG7            1E21
MSG8           1E26 MSG9            1E29 NFLAG$          0077
OPN$DCB        2128 OPN$FCB         210D OPREG$          0078
OPREG_SV_AREA  086E OPREG_SV_PTR    0835 ORARET@         14DC
OSRLS$         003B OSVER$          0085 OVRLY$          0069
PAKNAM$        0410 PAUSE@          0382 PCSAVE$         07AF
PDRV$          001B PHIGH$          001C PRDCB$          0218
PUTA@DE        0DCD PUT_@           0DCA PUT_@_ROWCOL    0DC6
RCVSPEC        2038 RFLAG$          007B ROWCOL_2_ADDR   0DD0
RSPC5          2052 RSPC6           2057 RST38@          1BFF
RSTOR$         04C4 RWRIT@          13A2 S1DCB$          0238
SBUFF$         1D00 SET@EXEC        1A79 SET_SCROLL      0CF3
SFCB$          008C SFLAG$          007C SIDCB$          0220
SODCB$         0228 SPACE4$         2142 STACK$          0380
START$         0000 SVC$NUM         214D SVC$RET         215C
SVCRET$        000B SVCTAB$         0100 SVRET           2071
SVSVC          205D SYS4            1E00 SYSERR$         1B13
TCB$           004E TFLAG$          007D TIME$           002D
TIMER$         002C TIMSL$          002B TIMTSK$         0713
TMPMT$         04C3 TRACE_INT       07B1 TYPHK$          0A8F
TYPTSK$        0B26 UNK$TYP         213D USRET           1F23
```

```
USTOR$         0013 VFLAG$        007F WORDS          21A6
WRINT$         0080 ZERO$         0401 ZEROA@         13A0

1E00 is the transfer address
00000 Total errors
```

NOTES:

SYS5/SYS

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 SYS0/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 = 0, return
1E02 C8           03310          RET     Z
1E03 F1           03320          POP     AF              ;Discard return to SYS0
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         03550           LD      SP,HL
1E26 2ABE00     03560           LD      HL,(PCREG)
1E29 2B         03570           DEC     HL
1E2A 7E         03580           LD      A,(HL)          ;P/u the byte at PC
1E2B FEF7       03590           CP      0F7H            ;  & check for breakpoint
1E2D 2003       03600           JR      NZ,$?1          ;Go if not a breakpoint
1E2F 22BE00     03610           LD      (PCREG),HL
                03620 ;
                03630 ;         This next routine picks up the data stored in the
                03640 ;         instruction storage areas used to hold the
                03650 ;         address & byte of the inserted RST's used to
                03660 ;         control the single step mode. If the address
                03670 ;         save area is zero, then an RST was not inserted.
                03680 ;         Two areas are needed because DEBUG inserts
                03690 ;         RST 48's at both CALL origin & destination.
                03700 ;
1E32 21A000     03710 $?1       LD      HL,SAVONE
1E35 0602       03720           LD      B,2             ;Set up loop for 2 areas
1E37 AF         03730 $?2       XOR     A               ;Clear register A & flags
1E38 5E         03740           LD      E,(HL)          ;P/u the next 2 bytes
1E39 77         03750           LD      (HL),A          ;  (where an address
1E3A 23         03760           INC     HL              ;  would be stored) while
1E3B 56         03770           LD      D,(HL)          ;  simultaneously setting
1E3C 77         03780           LD      (HL),A          ;  the save area to zero
1E3D 23         03790           INC     HL
1E3E 7B         03800           LD      A,E             ;Ck if the area was zero
1E3F B2         03810           OR      D
1E40 2807       03820           JR      Z,$?3           ;If zero, no RST entry
1E42 1A         03830           LD      A,(DE)          ;Address save <> zero,
1E43 FEF7       03840           CP      0F7H            ;  ck byte for RST 48
1E45 2002       03850           JR      NZ,$?3
1E47 7E         03860           LD      A,(HL)          ;  Was RST 48, restore
1E48 12         03870           LD      (DE),A          ;  the program byte
1E49 23         03880 $?3       INC     HL
1E4A 10EB       03890           DJNZ    $?2             ;Loop thru 2 save areas
1E4C ED7BBC00   03900 CMND      LD      SP,(SPREG)      ;Set up the stack
1E50 CDD51E     03910           CALL    WRREGS          ;  & display normal CRT
1E53 210010     03920           LD      HL,16<8!0       ;Move cursor to 16,0
1E56 0603       03930           LD      B,3             ;Command
1E58 3E0F       03940           LD      A,15            ;Svc @VDCTL
1E5A EF         03950           RST     28H             ;Set cursor
1E5B CDC921     03960           CALL    INPUT@          ;Get command
1E5E FE67       03970           CP      'g'             ;Goto AAAA,(BBBB(,CCCC))
1E60 CA821F     03980           JP      Z,CMD_G
1E63 214C1E     03990           LD      HL,CMND         ;Set up a return branch
1E66 E5         04000           PUSH    HL
1E67 FE73       04010           CP      's'             ;Set CRT to full screen?
1E69 2832       04020           JR      Z,CMD_S
1E6B FE3B       04030           CP      ';'             ;Inc CRT one page?
1E6D 2842       04040           JR      Z,CMD_INC
1E6F FE2D       04050           CP      '-'             ;Dec CRT one page?
1E71 2856       04060           JR      Z,CMD_DEC
1E73 FE6F       04070           CP      'o'             ;Out to DOS
1E75 2857       04080           JR      Z,CMD_O
1E77 FE63       04090           CP      'c'             ;Single step with CALL?
1E79 2806       04100           JR      Z,CMD_C
1E7B FE64       04110           CP      'd'             ;Display AAAA <space>
1E7D 282C       04120           JR      Z,CMD_D
1E7F FE69       04130           CP      'i'             ;Single step?
1E81 CA8B20     04140 CMD_C     JP      Z,CMD_CI
1E84 FE61       04150           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 ;
```

```
               04770 ;        Register display routine
               04780 ;
               04790 WRREGS
1ED5 3E1C      04800          LD     A,1CH           ;Home the cursor
1ED7 CD4206    04810          CALL   @DSP
               04820          IF     @MOD4
1EDA 3E0F      04830          LD     A,15            ;Turn off the cursor
1EDC CD4206    04840          CALL   @DSP
               04850          ENDIF
1EDF 3AA100    04860          LD     A,(SAVTWO)      ;0 = Normal display mode
1EE2 B7        04870          OR     A               ;<> 0 = Full display mode
1EE3 2068      04880          JR     NZ,FULDSP       ;No reg display if FULL
1EE5 21A800    04890          LD     HL,AFREG        ;Pt to register save area
1EE8 E5        04900          PUSH   HL
1EE9 21561F    04910          LD     HL,REGTBL       ;Pt to reg symbol table
1EEC 060C      04920          LD     B,12            ;Init for 12 registers
1EEE CD3522    04930 $?8      CALL   WR3BYT          ;Write 3-character symbol
1EF1 E3        04940          EX     (SP),HL         ;Exchange reg save ptr
1EF2 5E        04950          LD     E,(HL)          ;Place reg value -> DE
1EF3 23        04960          INC    HL
1EF4 56        04970          LD     D,(HL)
1EF5 23        04980          INC    HL              ;Place next reg save
1EF6 E5        04990          PUSH   HL              ;  pointer on the stack
1EF7 EB        05000          EX     DE,HL           ;Reg value -> HL
1EF8 3E3D      05010          LD     A,'='
1EFA CD4206    05020          CALL   @DSP
1EFD CD3122    05030          CALL   WRSPA@
1F00 7C        05040          LD     A,H             ;Write hi-order byte
1F01 CD2E22    05050          CALL   WRHEX
1F04 7D        05060          LD     A,L             ;Write lo-order byte
1F05 CD2E22    05070          CALL   WRHEX
1F08 78        05080          LD     A,B             ;Get loop counter &
1F09 E60B      05090          AND    0BH             ;  ck if 12 => AF pair
1F0B FE08      05100          CP     08H             ;  or if 8 => AF' pair
1F0D 201C      05110          JR     NZ,NOFLG        ;Bypass if not flag reg
1F0F 4D        05120          LD     C,L             ;Transfer 'F' reg to C &
1F10 C5        05130          PUSH   BC              ;  save the loop counter
1F11 217A1F    05140          LD     HL,FLGTBL       ;Pt to flag synbol table
1F14 0608      05150          LD     B,8             ;Init for 8 bits
1F16 CB21      05160 $?9      SLA    C               ;Shift a bit into carry
1F18 7E        05170          LD     A,(HL)          ;P/u flag table character
1F19 3802      05180          JR     C,$?10          ;Use table char if bit on
1F1B 3E2D      05190          LD     A,'-'           ;  else use a dash
1F1D CD4206    05200 $?10     CALL   @DSP
1F20 23        05210          INC    HL              ;Next flag table char
1F21 10F3      05220          DJNZ   $?9             ;Loop for 8 flag bits
1F23 C1        05230          POP    BC              ;Get main loop counter
1F24 3EFD      05240          LD     A,61+0C0H       ;Tab 60 to put cursor
1F26 CD4206    05250          CALL   @DSP            ;  on next line
1F29 1803      05260          JR     $?11
1F2B CD7321    05270 NOFLG    CALL   WRMEM
1F2E E1        05280 $?11     POP    HL              ;Get next reg save ptr
1F2F E3        05290          EX     (SP),HL         ;Exc with next reg symbol
1F30 10BC      05300          DJNZ   $?8             ;Loop end
1F32 E1        05310          POP    HL              ;Get reg save ptr (fini)
1F33 2AA600    05320          LD     HL,(DSPADR)     ;P/u memory disp address
1F36 0604      05330          LD     B,4             ;Init for 4 lines
1F38 3EC6      05340 $?12     LD     A,6+0C0H        ;Tab 6 spaces
1F3A CD4206    05350          CALL   @DSP
1F3D CD1522    05360          CALL   WR2HEX@         ;Write the memory address
1F40 CD3122    05370          CALL   WRSPA@
```

```
1F43 CD7321     05380          CALL    WRMEM           ;Write a line of memory
1F46 10F0       05390          DJNZ    $?12            ;Loop until 4 or 16
1F48 3E1F       05400          LD      A,1FH           ;Clear to end-of-frame
1F4A C34206     05410          JP      @DSP
1F4D 2AA600     05420 FULDSP   LD      HL,(DSPADR)     ;P/u display address
1F50 2E00       05430          LD      L,0             ;Round to multiple of 256
1F52 0610       05440          LD      B,16            ;Init for 16 lines
1F54 18E2       05450          JR      $?12
                05460 ;
                05470 ;        Register symbol table
                05480 ;
1F56 61         05490 REGTBL   DB      'af bc de hl af''bc''de''hl''ix iy sp pc '
     66 20 62 63 20 64 65 20
     68 6C 20 61 66 27 62 63
     27 64 65 27 68 6C 27 69
     78 20 69 79 20 73 70 20
     70 63 20
                05500 ;
                05510 ;        Flag register bit symbol table
                05520 ;
1F7A 53         05530 FLGTBL   DB      'SZ1H1PNC'
     5A 31 48 31 50 4E 43
                05540 ;
                05550 ;        Command G - Go to memory address NNNN,
                05560 ;         Optional breakpoints
                05570 ;
1F82 0602       05580 CMD_G    LD      B,2             ;Init for maximum of
1F84 11A500     05590          LD      DE,NXTBYT       ;  two breakpoints
1F87 CDE421     05600          CALL    HEXIN@          ;Get exec address
1F8A 2803       05610          JR      Z,$?13          ;Go on end
1F8C 22BE00     05620          LD      (PCREG),HL      ;  else save new start
1F8F 380A       05630 $?13     JR      C,$?14          ;Go if <ENTER> used
1F91 CDE421     05640          CALL    HEXIN@          ;Get a breakpoint
1F94 F5         05650          PUSH    AF
1F95 C4C51F     05660          CALL    NZ,$?17         ;Set if brkpt entered
1F98 F1         05670          POP     AF
1F99 10F4       05680          DJNZ    $?13
                05690 $?14
1F9B AF         05700          XOR     A
1F9C 329F19     05710          LD      (@DBGHK),A      ;Init DEBUG on
                05720 ;
                05730 ;        This next section of code picks up the register
                05740 ;        save area, pushes the save area onto the stack,
                05750 ;        then pops out into the correct reg assignments.
                05760 ;
1F9F 21BD00     05770 $?15     LD      HL,REGSAV       ;End of reg save area
1FA2 060B       05780          LD      B,11            ;Init for 11 regs
1FA4 56         05790 $?16     LD      D,(HL)
1FA5 2B         05800          DEC     HL
1FA6 5E         05810          LD      E,(HL)
1FA7 2B         05820          DEC     HL
1FA8 D5         05830          PUSH    DE
1FA9 10F9       05840          DJNZ    $?16
1FAB F1         05850          POP     AF              ;Now pop the registers
1FAC C1         05860          POP     BC
1FAD D1         05870          POP     DE
1FAE E1         05880          POP     HL
1FAF 08         05890          EX      AF,AF'
1FB0 D9         05900          EXX
1FB1 F1         05910          POP     AF
1FB2 C1         05920          POP     BC
```

```
1FB3 D1         05930          POP     DE
1FB4 E1         05940          POP     HL
1FB5 08         05950          EX      AF,AF'
1FB6 D9         05960          EXX
1FB7 DDE1       05970          POP     IX
1FB9 FDE1       05980          POP     IY
1FBB E1         05990          POP     HL
1FBC F9         06000          LD      SP,HL
1FBD 2ABE00     06010          LD      HL,(PCREG)        ;Init the branch address
1FC0 E5         06020          PUSH    HL
1FC1 2AAE00     06030          LD      HL,(HLREG)
1FC4 C9         06040          RET                       ;Go to branch
                06050 ;
                06060 ;        This next routine will insert an RST 48 inst into
                06070 ;        the target of a single-step or breakpoint
                06080 ;        providing the target address is a RAM location.
                06090 ;        If it is, the target byte and its address are
                06100 ;        saved in one of the instruction save areas.
                06110 ;        If the target address is ROM or nonexistent, a
                06120 ;        branch to command INPUT routine is taken instead
                06130 ;        of the pending operation.
                06140 ;
1FC5 7E         06150 $?17     LD      A,(HL)            ;Save byte of next inst
1FC6 12         06160          LD      (DE),A
1FC7 1B         06170          DEC     DE
1FC8 3EF7       06180          LD      A,0F7H            ;Insert RST 48 into
1FCA 77         06190          LD      (HL),A            ;  next INST address
1FCB BE         06200          CP      (HL)              ;Ck if RAM/ROM/no memory
1FCC C2321E     06210          JP      NZ,$?1            ;Go to command if not RAM
1FCF 7C         06220          LD      A,H               ;Is RAM, save address of
1FD0 12         06230          LD      (DE),A            ;  insertion into buffer
1FD1 1B         06240          DEC     DE                ;  pointed to by DE, DE-1
1FD2 7D         06250          LD      A,L
1FD3 12         06260          LD      (DE),A
1FD4 1B         06270          DEC     DE
1FD5 C9         06280          RET
                06290 ;
                06300 ;        Commands A & H - Modify address NNNN to XX
                06310 ;         <SPACE> increments address
                06320 ;
1FD6 32A000     06330 CMD_AH   LD      (SAVONE),A        ;Save entry condition
1FD9 2AA300     06340          LD      HL,(NXTADR)       ;Default to current mod addr
1FDC CDE421     06350          CALL    HEXIN@
1FDF 22A300     06360 $?18     LD      (NXTADR),HL       ;Adjust addr for mod
1FE2 D8         06370          RET     C                 ;Return on <ENTER>
1FE3 E5         06380          PUSH    HL
1FE4 CDD51E     06390          CALL    WRREGS
1FE7 21000D     06400          LD      HL,13<8!0         ;Cursor to 13,0
1FEA 0603       06410          LD      B,3
1FEC 3E0F       06420          LD      A,15              ;SVC @VDCTL set cursor
1FEE EF         06430          RST     28H
1FEF 2AA300     06440          LD      HL,(NXTADR)       ;P/u mod address again
1FF2 CD1522     06450          CALL    WR2HEX@           ;Write the address & save
1FF5 E5         06460          PUSH    HL                ;  the mod addr again
1FF6 21000E     06470          LD      HL,14<8!0         ;Cursor to 14,0
1FF9 0603       06480          LD      B,3
1FFB 3E0F       06490          LD      A,15              ;SVC @VDCTL set cursor
1FFD EF         06500          RST     28H
1FFE E1         06510          POP     HL                ;Recover mod addr
1FFF CD1320     06520          CALL    AHDSP
2002 3E2D       06530          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
```

```
206B 23         07150           INC     HL              ;Match, ck 3rd reg symbol
206C 7E         07160           LD      A,(HL)          ;P/u the 3rd table symbol
206D BB         07170           CP      E               ;  & compare with input
206E 20F2       07180           JR      NZ,$?23         ;-> next if no match
2070 3E18       07190           LD      A,18H           ;Convert counter to index
2072 90         07200           SUB     B               ;  into reg save area
2073 90         07210           SUB     B
2074 4F         07220           LD      C,A             ;Index into BC
2075 0600       07230           LD      B,0
2077 21A800     07240           LD      HL,AFREG        ;Start of reg save area
207A 09         07250           ADD     HL,BC           ;Add index to get pointer
207B E5         07260           PUSH    HL              ;Save the pointer
207C 3E1E       07270           LD      A,1EH           ;Erase to end-of-line
207E CD4206     07280           CALL    @DSP
2081 D1         07290           POP     DE              ;Recover pointer
2082 CDE421     07300           CALL    HEXIN@          ;Read in the new value
2085 C8         07310           RET     Z               ;No update if none
2086 EB         07320           EX      DE,HL           ;Exchg value/pointer
2087 73         07330           LD      (HL),E          ;Insert new value into
2088 23         07340           INC     HL              ;  register save area
2089 72         07350           LD      (HL),D
208A C9         07360           RET
                07370 ;
                07380 ;         Command I - Step one instruction at a time
                07390 ;
208B F5         07400 CMD_CI    PUSH    AF              ;Save whether I or C
208C ED5BBE00   07410           LD      DE,(PCREG)      ;Point to inst address
2090 1A         07420           LD      A,(DE)          ;  & get it
2091 215721     07430           LD      HL,XY_TAB       ;IX,IY Table
2094 FEDD       07440           CP      0DDH            ;Is inst an IX?
2096 280E       07450           JR      Z,$?25
2098 FEFD       07460           CP      0FDH            ;Is inst an IY?
209A 280A       07470           JR      Z,$?25
209C 211F21     07480           LD      HL,OP_TAB       ;All X IX, IY, & ED
209F FEED       07490           CP      0EDH            ;Is inst an ED?
20A1 2006       07500           JR      NZ,$?26
20A3 215021     07510           LD      HL,ED_TAB       ;ED Table
20A6 13         07520 $?25      INC     DE              ;Get next byte for
20A7 1A         07530           LD      A,(DE)          ;  IX, IY, and ED inst
20A8 1B         07540           DEC     DE              ;Reset ptr to 1st byte
20A9 4F         07550 $?26      LD      C,A             ;Inst byte to reg C
                07560 ;
                07570 ;         This next section of code determines the length
                07580 ;         of all instructions and whether they
                07590 ;         are CALLs, JumPs, or RETurns.
                07600 ;
20AA 7E         07610 $?27      LD      A,(HL)          ;P/u table value &
20AB A1         07620           AND     C               ;  strip off certain bits
20AC 23         07630           INC     HL              ;Pt to table code
20AD BE         07640           CP      (HL)            ;If a match, the inst is
20AE 23         07650           INC     HL              ;  fully decoded as to
20AF 2806       07660           JR      Z,$?28          ;  length & type by the
20B1 23         07670           INC     HL              ;  next byte
20B2 7E         07680           LD      A,(HL)          ;Ck for table end
20B3 FE05       07690           CP      5
20B5 30F3       07700           JR      NC,$?27
20B7 7E         07710 $?28      LD      A,(HL)          ;Get control/length byte
20B8 47         07720           LD      B,A             ;  into reg B
20B9 E60F       07730           AND     0FH             ;Strip off the control
20BB 6F         07740           LD      L,A             ;Put length into reg L
20BC 2600       07750           LD      H,0             ;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
               08370 ;        selected bits and include others. The result is
               08380 ;        compared to the next table byte (test byte) for
               08390 ;        a match. If matched, then the inst byte has been
               08400 ;        identified as to its class & length. The 3rd byte
               08410 ;        denotes the class and length as follows:
               08420 ; High order nybble
               08430 ;          0 = Regular instruction
               08440 ;          1 = JP (HL) instruction
               08450 ;          2 = JP (IX) or JP (IY) instruction
               08460 ;          3 = JR or DJNZ instructions
               08470 ;          4 = JP instructions
               08480 ;          5 = RET instructions
               08490 ;          6 = CALL instructions
               08500 ;          7 = RST instructions
               08510 ; Low order nybble = the length
               08520 ;        The last byte of each table is the length of
               08530 ;        all other instructions.
               08540 ;
               08550 ;        Table for regular instructions (no IX, IY, ED)
               08560 ;
211F C7        08570 OP_TAB   DB      0C7H,0C0H,51H   ;C8, D8, E8, F8
     C0 51
2122 FF        08580          DB      0FFH,0C9H,51H   ;C9
     C9 51
2125 FF        08590          DB      0FFH,0E9H,11H   ;E9
     E9 11
2128 CF        08600          DB      0CFH,01H,03H    ;01, 11, 21, 31
     01 03
212B E7        08610          DB      0E7H,22H,3      ;22, 2A, 32, 3A
     22 03
212E C7C2      08620          DW      0C2C7H          ;C2, CA, D2, DA, E2, EA,
2130 43        08630          DB      43H             ;  F2, FA
2131 FF        08640          DB      0FFH,0C3H,43H   ;C3
     C3 43
2134 C7C4      08650          DW      0C4C7H          ;C4, CC, D4, DC, E4, EC,
2136 63        08660          DB      63H             ;  F4, FC
2137 FF        08670          DB      0FFH,0CDH,63H   ;CD
     CD 63
213A C706      08680          DW      06C7H           ;06, 0E, 16, 1E, 26, 2E,
213C 02        08690          DB      02H             ;  36, 3E
213D F7        08700          DB      0F7H,0D3H,02    ;D3, DB
     D3 02
2140 C7C6      08710          DW      0C6C7H          ;C6, CE, D6, DE, E6, EE,
2142 02        08720          DB      02H             ;  F6, FE
2143 FF        08730          DB      0FFH,0CBH,2     ;All CB instructions
     CB 02
2146 F7        08740          DB      0F7H,10H,32H    ;10, 18
     10 32
2149 E7        08750          DB      0E7H,20H,32H    ;20, 28, 30, 38
     20 32
214C C7        08760          DB      0C7H,0C7H,71H   ;RST instructions
     C7 71
214F 01        08770          DB      1               ;All others are 1-byte
               08780 ;
               08790 ;        Next table is for ED - extended instructions
               08800 ;
2150 C7        08810 ED_TAB   DB      0C7H,43H,04H    ;43, 4B, 53, 5B, 73, 7B
     43 04
2153 F7        08820          DB      0F7H,45H,52H    ;45, 4D
     45 52
2156 02        08830          DB      2               ;All other ED are 2-byte
```

```
                 08840 ;
                 08850 ;       IX, IY Index instructions table
                 08860 ;
2157 FE          08870 XY_TAB  DB      0FEH,34H,03      ;34, 35
     34 03
215A C0          08880         DB      0C0H,40H,03      ;4X, 5X, 6X, 7X (X = 0-F)
     40 03
215D C0          08890         DB      0C0H,80H,03      ;8X, 9X, AX, BX (X= 0-F)
     80 03
2160 FF          08900         DB      0FFH,21H,04      ;21
     21 04
2163 FF          08910         DB      0FFH,22H,04      ;22
     22 04
2166 FF          08920         DB      0FFH,2AH,04      ;2A
     2A 04
2169 FF          08930         DB      0FFH,36H,04      ;36
     36 04
216C FF          08940         DB      0FFH,0CBH,04     ;CB
     CB 04
216F FF          08950         DB      0FFH,0E9H,22H    ;E9
     E9 22
2172 02          08960         DB      02H              ;All others are 2-bytes
                 08970 ;
                 08980 ;       Routine to display memory on CRT screen
                 08990 ;
2173 C5          09000 WRMEM   PUSH    BC               ;Save main counter 4/16
2174 3E3D        09010         LD      A,'='
2176 CD4206      09020         CALL    @DSP
2179 3C          09030         INC     A                ;'>'
217A CD4206      09040         CALL    @DSP
217D 0610        09050         LD      B,16             ;Init for 16 lines
217F E5          09060         PUSH    HL               ;Save memory pointer
2180 CDA421      09070 $?36    CALL    GRPHIC           ;Ck if need graphic bars
2183 CD1122      09080         CALL    WR1HEX@          ;Call on HEX display only
2186 10F8        09090         DJNZ    $?36             ;Loop until full line
2188 E1          09100         POP     HL               ;Rcvr memory pointer
                 09110 ;
                 09120 ;       Now write the line in ASCII
                 09130 ;
2189 CD3122      09140         CALL    WRSPA@
218C 0610        09150         LD      B,16
218E CDC321      09160 $?37    CALL    $?41             ;Space after 8th
2191 7E          09170         LD      A,(HL)           ;P/u the byte -> reg A
2192 FE20        09180         CP      20H              ;Repl controls with '.'
2194 3804        09190         JR      C,$?38
2196 FEC0        09200         CP      0C0H             ;Tabs/specials with '.'
2198 3802        09210         JR      C,$?39
219A 3E2E        09220 $?38    LD      A,'.'
219C CD4206      09230 $?39    CALL    @DSP
219F 23          09240         INC     HL               ;Bump memory address
21A0 10EC        09250         DJNZ    $?37
21A2 C1          09260         POP     BC               ;Get line counter
21A3 C9          09270         RET
                 09280 ;
                 09290 ;       This routine determines if the vertical graphic
                 09300 ;       bars should be surrounding the current character
                 09310 ;
21A4 ED5BA300    09320 GRPHIC  LD      DE,(NXTADR)      ;P/u modification address
21A8 13          09330         INC     DE               ;  & increment it
21A9 E5          09340         PUSH    HL               ;Save current memory
21AA AF          09350         XOR     A                ;  display address
```

```
21AB ED52     09360          SBC    HL,DE           ;Ck if mod addr=disp addr
              09370          IF     @MOD4
21AD 3E95     09380          LD     A,95H           ;Graphic left bar
              09390          ENDIF
              09400          IF     @MOD2
              09410          LD     A,15H
              09420          ENDIF
21AF 280E     09430          JR     Z,$?40          ;Insert graphic if equal
21B1 CDC321   09440          CALL   $?41            ;Not =, insert space if
21B4 23       09450          INC    HL              ;  between pos 8 & 9
21B5 7D       09460          LD     A,L             ;Result is zero if next
21B6 B4       09470          OR     H               ;  char address is also
              09480                                 ;  the display address
21B7 E1       09490          POP    HL              ;Get current mem disp adr
              09500          IF     @MOD4
21B8 3EAA     09510          LD     A,0AAH          ;Graphic right bar output
21BA CA4206   09520          JP     Z,@DSP          ;Go if yes
21BD 1808     09530          JR     $?42            ;  else continue
              09540          ENDIF
              09550          IF     @MOD2
              09560          JR     NZ,$?42         ;Go if not
              09570          XOR    A               ;  lead in
              09580          CALL   @DSP            ;Init video lead in
              09590          LD     A,15H
              09600          JP     @DSP            ;  and display
              09610          ENDIF
21BF          09620 $?40     EQU    $
              09630          IF     @MOD2
              09640          PUSH   AF
              09650          XOR    A
              09660          CALL   @DSP            ;Lead in code
              09670          POP    AF              ;Restore
              09680          ENDIF
21BF CD4206   09690          CALL   @DSP            ;Display char
21C2 E1       09700          POP    HL              ;Recover current display
21C3 78       09710 $?41     LD     A,B             ;  address & output a
21C4 FE08     09720          CP     8               ;  space if between the
21C6 C0       09730          RET    NZ              ;  8th & 9th bytes
21C7 1868     09740 $?42     JR     WRSPA@          ;  else just return
              09750 ;
              09760 ;        This routine will return with zero flag set
              09770 ;        on entry of a comma or a SPACE. Entry of <ENTER>
              09780 ;        will set carry flag and return
              09790 ;
21C9 D5       09800 INPUT@   PUSH   DE
21CA CD2806   09810 $?43     CALL   @KEY
21CD FE0D     09820          CP     0DH             ;ENTER?
21CF 2810     09830          JR     Z,$?44
21D1 FE20     09840          CP     20H             ;Get another char if
21D3 38F5     09850          JR     C,$?43          ;  entry was control
21D5 CBEF     09860 INPUC@   SET    5,A             ;Cvrt UC to lc
21D7 CD4206   09870          CALL   @DSP            ;Not control, disp it
21DA D1       09880          POP    DE
21DB FE2C     09890          CP     ','             ;Return with zero flag
21DD C8       09900          RET    Z               ;  set if a comma
21DE FE20     09910          CP     ' '             ;Return with zero flag
21E0 C9       09920          RET                    ;  set if <SPACE>
21E1 D1       09930 $?44     POP    DE
21E2 37       09940          SCF                    ;<ENTER> will set
21E3 C9       09950          RET                    ;  the carry flag
              09960 ;
              09970 ;         This routine will read in digits
              09980 ;         and convert them to binary
              09990 ;
21E4 CDC921   10000 HEXIN@    CALL    INPUT@          ;Get char and return on
21E7 C8       10010           RET     Z               ;  SPACE, COMMA, or ENTER
21E8 210000   10020           LD      HL,0            ;Init value to zero
21EB CD0022   10030 $?45      CALL    CVB             ;Convert to binary if ok
21EE DA4C1E   10040           JP      C,CMND          ;  else back on bad digit
21F1 29       10050           ADD     HL,HL           ;Multiply current value
21F2 29       10060           ADD     HL,HL           ;  by 16 and insert the
21F3 29       10070           ADD     HL,HL           ;  new digit into the
21F4 29       10080           ADD     HL,HL           ;  lo-order nybble of L
21F5 B5       10090           OR      L
21F6 6F       10100           LD      L,A
21F7 CDC921   10110           CALL    INPUT@          ;Get another character
21FA 20EF     10120           JR      NZ,$?45         ;Go if not separator
21FC 1F       10130           RRA                     ;Force <ENTER> to set
21FD CE81     10140           ADC     A,81H           ;  the carry flag
21FF C9       10150           RET
              10160 ;
              10170 ;         Routine to convert expected ASCII hex digit to
              10180 ;         its binary value. Set Carry-flag on bad digit
              10190 ;
2200 D630     10200 CVB       SUB     '0'             ;Convert digit to binary
2202 D8       10210           RET     C               ;Error if < '0'
2203 C6C9     10220           ADD     A,0C9H          ;Ck for > F (46H-30H=16H)
              10230                                   ;  (16H + E9H = FFH)
2205 D8       10240           RET     C               ;Error if > ASCII 'F'
2206 C606     10250           ADD     A,6             ;(E9H-EFH) to (EFH-05H)
2208 3803     10260           JR      C,ATOF          ;Carry denotes was <A-F>
220A C627     10270           ADD     A,27H           ;(EFH-FFH) to (F6H-06H)
220C D8       10280           RET     C               ;Error if (3AH-3FH/:-?)
220D C60A     10290 ATOF      ADD     A,0AH           ;(00D-06D) to (10D-16D)
              10300                                   ;  or (F6H-FFH) to (0-9)
220F B7       10310           OR      A               ;Set zero flag on zero
2210 C9       10320           RET
              10330 ;
              10340 ;         Routine to write one byte as two hex digits
              10350 ;
2211 7E       10360 WR1HEX@   LD      A,(HL)
2212 23       10370           INC     HL
2213 1805     10380           JR      CV2HEX@
              10390 ;
              10400 ;         Routine to write 2 bytes (HL) as 4 hex digits
              10410 ;
2215 7C       10420 WR2HEX@   LD      A,H
2216 CD1A22   10430           CALL    CV2HEX@
2219 7D       10440           LD      A,L
              10450 ;
              10460 ;         Routine converts a byte to 2 hex digits
              10470 ;
221A F5       10480 CV2HEX@   PUSH    AF              ;Save the byte in A
221B 1F       10490           RRA                     ;Move hi-order
221C 1F       10500           RRA                     ;  into lo-order
221D 1F       10510           RRA
221E 1F       10520           RRA
221F CD2322   10530           CALL    $?46            ;Strip off hi-order
              10540                                   ;  & convert to ASCII
2222 F1       10550           POP     AF              ;Recover the byte
2223 E60F     10560 $?46      AND     0FH             ;Strip off hi-order
              10570                                   ;  & 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
22A0 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 D0       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 2008     11400          JR      NZ,QUERY
22AF 2ABE00   11410          LD      HL,(PCREG)        ;Get current PC location
22B2 23       11420          INC     HL                ;  and increment it
22B3 22BE00   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 2024     11500          JR      NZ,DISKIO
22BB 3E1E     11510          LD      A,1EH             ;Clear to end of line
22BD CD4206   11520          CALL    @DSP
22C0 CDE421   11530          CALL    HEXIN@            ;Get port number
22C3 C8       11540          RET     Z                 ;Back if no value
22C4 4D       11550          LD      C,L
22C5 3807     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
22D0 CD4206   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 D630     11760 DISKIO   SUB     30H               ;Cvrt drive to binary
22E1 FE08     11770          CP      8                 ;Check on max drive
22E3 30F6     11780          JR      NC,EXTDBG         ;Exit if not <0-7>
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,0                 ;Reset sector # to 0
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,0                 ;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),0              ;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$-2
               13020         ERR     'Module too big'
               13030         ENDIF
23FE           13040         ORG     MAXCOR$-2
23FE BD05      13050         DW      LAST-SYS5         ;Overlay size
               00180 ;
1E00           00190         END     SYS5
```

```
$?1             1E32 $?10            1F1D $?11            1F2E
$?12            1F38 $?13            1F8F $?14            1F9B
$?15            1F9F $?16            1FA4 $?17            1FC5
$?18            1FDF $?19            200F $?2             1E37
$?20            2057 $?21            205C $?22            2061
$?23            2062 $?24            2066 $?25            20A6
$?26            20A9 $?27            20AA $?28            20B7
$?28A           20F1 $?29            20F6 $?3             1E49
$?30            20F9 $?31            20FC $?32            2102
$?33            210B $?34            2117 $?35            211A
$?36            2180 $?37            218E $?38            219A
$?39            219C $?4             1EB4 $?40            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             03B7 $A2             03B8 $A3             03B9
$CKEOF          1470 @$SYS           08F0 @@1             0000
@@2             0000 @@3             0000 @@4             0000
@ABORT          1B08 @ADTSK          1CDA @BANK           0877
@BKSP           1486 @BREAK          196F @BYTEIO         1300
@CHNIO          0689 @CKBRKC         0553 @CKDRV          1993
@CKEOF          158F @CKTSK          1CF5 @CLOSE          1999
@CLS            0545 @CMNDI          197E @CMNDR          197B
@CTL            0623 @DATE           07A8 @DBGHK          199F
@DCINIT         19C0 @DCRES          19C4 @DCSTAT         19B5
@DCTBYT         1A2B @DEBUG          19A0 @DECHEX         03E1
@DIRCYL         18F7 @DIRRD          18BB @DIRWR          1803
@DIV16          06E3 @DIV8           1927 @DODIR          19AF
@DOKEY          19A9 @DSP            0642 @DSPLY          052D
@ERROR          1B0F @EXIT           1B0B @FEXT           1984
@FLAGS          196A @FNAME          199C @FRENCH         0000
@FSPEC          1981 @GATRD          1874 @GATWR          1875
@GERMAN         0000 @GET            0638 @GTDCB          1990
@GTDCT          1A1E @GTMOD          19B2 @HDFMT          19E4
@HEX16          07BD @HEX8           07C2 @HEXDEC         06F6
@HIGH$          1948 @HITRD          1897 @HITWR          1898
@HZ50           0000 @ICNFG          0086 @INIT           198D
@INTL           0000 @IPL            1BF2 @JCL            0630
@KBD            0635 @KEY            0628 @KEYIN          0585
@KITSK          0089 @KLTSK          1CD0 @LOAD           1B38
@LOC            14B3 @LOF            14DE @LOGER          0503
@LOGOT          0500 @MOD2           0000 @MOD4           FFFF
@MSG            0530 @MUL16          06C9 @MUL8           190A
@NMI            0066 @OPEN           198A @OPREG          0084
@PARAM          1987 @PAUSE          0382 @PEOF           14A2
@POSN           1434 @PRINT          0528 @PRT            063D
@PUT            0645 @RAMDIR         19AC @RDHDR          19D8
@RDSEC          19F4 @RDSSC          18D8 @RDTRK          19E0
@READ           1513 @REMOVE         19A6 @RENAME         1996
@REW            149B @RMTSK          1CD7 @RPTSK          1CEB
@RREAD          1473 @RSLCT          19D4 @RST00          0000
@RST08          0008 @RST10          0010 @RST18          0018
@RST20          0020 @RST28          0028 @RST30          0030
@RST38          0038 @RSTNMI         0FE9 @RSTOR          19C8
@RSTREG         0680 @RUN            1B1D @RWRIT          13AD
@SEEK           19D0 @SEEKSC         1421 @SKIP           1430
@SLCT           19BC @SOUND          0392 @STEPI          19CC
@TIME           078D @USA            FFFF @VDCTL          0B99
@VDCTL3         0D38 @VER            1560 @VRSEC          19DC
```

```
@WEOF          14EC @WHERE        1979 @WRITE          1531
@WRSEC         19E8 @WRSSC        19EC @WRTRK          19F0
@_VDCTL        0D42 ADDR_2_ROWCOL 0DF1 AFLAG$          006A
AFREG          00A8 AHDSP         2013 AHGET           2029
ATOF           220D AUTO?         1FF1 BAR$            0201
BLO1           2257 BLO2          226C BLO3            2276
BLO4           227B BLOCK         223F 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
CMND           1E4C CONFIG$       203F CORE$           0300
CRTBGN$        F800 CV2HEX@       221A CVB             2200
CYL_GRN        16AE D@FBYT8       1A26 DATE$           0033
DAYTBL$        04C7 DBGSV$        00A0 DCBKL$          0031
DCT$           0470 DCTBYT8@      1A29 DCTFLD@         1A34
DFLAG$         006D DIRBUF$       2300 DIS1            22FB
DIS10          2396 DIS11         239D DIS12           23A2
DIS2           2305 DIS3          231B DIS4            2328
DIS6           2346 DIS7          2353 DIS8            235F
DIS8A          2368 DIS9          237B DISKIO          22DF
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 EXTDBG        22DB EXTDBG$         19A4
FDDINT$        000E FEMSK$        006F FIL1            22A2
FILL           228B FLGTAB$       006A FLGTBL          1F7A
FULDSP         1F4D GETASC@       2031 GET_@_ROWCOL    0DAE
GRPHIC         21A4 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 JUMP            22AB
KCK@           07D6 KFLAG$        0074 KIDATA$         08FC
KIDCB$         0208 LAST          23BD LBANK$          0202
LDRV$          0023 LFLAG$        0075 LNKFCB@         1566
LOW$           001E LSVC$         000D MAXCOR$         2400
MAXDAY$        0401 MINCOR$       3000 MODOUT$         0076
MONTBL$        04DC NFLAG$        0077 NOFLG           1F2B
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 PUTA@DE         0DCD
PUT_@          0DCA PUT_@_ROWCOL  0DC6 QUE1            22CE
QUERY          22B7 REGSAV        00BD REGTBL          1F56
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
SYS5           1E00 SYSERR$       1B13 TCB$          004E
TFLAG$         007D TIME$         002D TIMER$        002C
TIMSL$         002B TIMTSK$       0713 TMPMT$        04C3
TRACE_INT      07B1 TYP3          2024 TYP4          2026
TYPHK$         0A8F TYPTSK$       0B26 USTOR$        0013
VFLAG$         007F WR1HEX@       2211 WR2HEX@       2215
WR3BYT         2235 WRHEX         222E WRINT$        0080
WRMEM          2173 WRREGS        1ED5 WRSPA@        2231
XY_TAB         2157 ZERO$         0401 ZEROA@        13A0

1E00 is the transfer address
00000 Total errors
```

NOTES:

NOTES:

SYS9/SYS

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 015602      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 CD4206     01910          CALL    @DSP
1F1F F1         01920          POP     AF
1F20 3815       01930 VER1     JR      C,VER2            ;Jump if <ENTER> used prev.
1F22 2AA300     01940          LD      HL,(NXTADR)       ;2nd default current mod addr
1F25 CDE421     01950          CALL    HEXIN@            ;Prompt new 2nd start
1F28 22A300     01960          LD      (NXTADR),HL
1F2B 200A       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 CD4206     02010          CALL    @DSP
1F36 F1         02020          POP     AF
1F37 210000     02030 VER2     LD      HL,0              ;Default length to verify
1F3A 380A       02040          JR      C,VER3            ;Go if <ENTER> sued prev
1F3C CDE421     02050          CALL    HEXIN@            ;Get new length
1F3F 2005       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 2AA600     02120          LD      HL,(DSPADR)       ;Set up for compare
1F4B ED5BA300   02130          LD      DE,(NXTADR)
1F4F 1A         02140 VER4     LD      A,(DE)
1F50 BE         02150          CP      (HL)              ;Compare the two locations
1F51 2007       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 0B         02190          DEC     BC
1F56 78         02200          LD      A,B
1F57 B1         02210          OR      C
1F58 20F5       02220          JR      NZ,VER4
1F5A ED53A300   02230 VER5     LD      (NXTADR),DE       ;Store non-match or end of
1F5E 22A600     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'-'0'
1F64 2052       02300          JR      NZ,PRINT
1F66 2AA300     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 22A300     02350          LD      (NXTADR),HL
1F71 2012       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 CD4206     02400          CALL    @DSP
1F7C F1         02410          POP     AF
1F7D 3AA500     02420          LD      A,(NXTBYT)        ;Get next default
1F80 6F         02430          LD      L,A
1F81 3AA200     02440          LD      A,(SAVTWO+1)
1F84 67         02450          LD      H,A
1F85 380F       02460 WOR1     JR      C,WOR2            ;Go if <ENTER>
1F87 CDE421     02470          CALL    HEXIN@            ;Get next value
1F8A 280A       02480          JR      Z,WOR2            ;Go if default
```

```
1F8C 7D       02490        LD      A,L             ;Store new value
1F8D 32A500   02500        LD      (NXTBYT),A
1F90 7C       02510        LD      A,H
1F91 32A200   02520        LD      (SAVTWO+1),A
1F94 1803     02530        JR      WOR3
1F96 CD1522   02540 WOR2   CALL    WR2HEX@         ;Display value
1F99 2AA300   02550 WOR3   LD      HL,(NXTADR)     ;Start looking here
1F9C 010000   02560        LD      BC,0            ;Init count to 64K
1F9F 3AA500   02570 WOR4   LD      A,(NXTBYT)
1FA2 EDB1     02580        CPIR                    ;Find first match
1FA4 C0       02590        RET     NZ              ;Return if none
1FA5 3AA200   02600        LD      A,(SAVTWO+1)    ;Get 2nd half of word
1FA8 BE       02610        CP      (HL)            ;Is a match?
1FA9 20F4     02620        JR      NZ,WOR4         ;Continue if not
1FAB 2B       02630        DEC     HL
1FAC 2B       02640        DEC     HL              ;Pt 1 byte before
1FAD 22A300   02650        LD      (NXTADR),HL     ;  and save that address
1FB0 7D       02660        LD      A,L
1FB1 E6C0     02670        AND     0C0H
1FB3 6F       02680        LD      L,A
1FB4 22A600   02690        LD      (DSPADR),HL     ;New display start
1FB7 C9       02700        RET
              02710 ;
              02720 ;      'p'rint aaaa,bbbb - print memory
              02730 ;
1FB8 FE40     02740 PRINT  CP      'p'-30H         ;If command is not 'P',
1FBA C0       02750 PRI1   RET     NZ              ;  back to SYS5
1FBB CDE421   02760        CALL    HEXIN@          ;Get start
1FBE C8       02770        RET     Z               ;Back if no start addr
1FBF E5       02780        PUSH    HL
1FC0 CDE421   02790        CALL    HEXIN@          ;Get end
1FC3 E3       02800        EX      (SP),HL
1FC4 C1       02810        POP     BC              ;Start in HL, end in BC
1FC5 C8       02820        RET     Z               ;Back if no end addr
1FC6 7D       02830        LD      A,L             ;Round to multiple of 16
1FC7 E6F0     02840        AND     0F0H
1FC9 6F       02850        LD      L,A
1FCA 3E0D     02860        LD      A,0DH           ;Send 2 blank lines to
1FCC CD3D06   02870        CALL    @PRT            ;  the printer
1FCF CD3D06   02880        CALL    @PRT
1FD2 E5       02890 PRI2   PUSH    HL              ;Routine to write HL
1FD3 7C       02900        LD      A,H             ;  as 4 hex digits
1FD4 1F       02910        RRA
1FD5 1F       02920        RRA
1FD6 1F       02930        RRA
1FD7 1F       02940        RRA
1FD8 E60F     02950        AND     0FH
1FDA C690     02960        ADD     A,90H
1FDC 27       02970        DAA
1FDD CE40     02980        ADC     A,40H
1FDF 27       02990        DAA
1FE0 CD3D06   03000        CALL    @PRT            ;1st one done
1FE3 7C       03010        LD      A,H
1FE4 E60F     03020        AND     0FH
1FE6 C690     03030        ADD     A,90H
1FE8 27       03040        DAA
1FE9 CE40     03050        ADC     A,40H
1FEB 27       03060        DAA
1FEC CD3D06   03070        CALL    @PRT            ;2nd one done
1FEF 7D       03080        LD      A,L
1FF0 1F       03090        RRA
```

```
1FF1 1F         03100           RRA
1FF2 1F         03110           RRA
1FF3 1F         03120           RRA
1FF4 E60F       03130           AND     0FH
1FF6 C690       03140           ADD     A,90H
1FF8 27         03150           DAA
1FF9 CE40       03160           ADC     A,40H
1FFB 27         03170           DAA
1FFC CD3D06     03180           CALL    @PRT            ;3rd one done
1FFF 7D         03190           LD      A,L
2000 E60F       03200           AND     0FH
2002 C690       03210           ADD     A,90H
2004 27         03220           DAA
2005 CE40       03230           ADC     A,40H
2007 27         03240           DAA
2008 CD3D06     03250           CALL    @PRT            ;4th one done
200B 3E20       03260           LD      A,20H           ;  & 2 spaces
200D CD3D06     03270           CALL    @PRT
2010 CD3D06     03280           CALL    @PRT
2013 1802       03290           JR      PRI4
2015 18BB       03300 PRI3      JR      PRI2
                03310 ;
                03320 ;         Write a byte in hex
                03330 ;
2017 7E         03340 PRI4      LD      A,(HL)
2018 1F         03350           RRA
2019 1F         03360           RRA
201A 1F         03370           RRA
201B 1F         03380           RRA
201C E60F       03390           AND     0FH
201E C690       03400           ADD     A,90H
2020 27         03410           DAA
2021 CE40       03420           ADC     A,40H
2023 27         03430           DAA
2024 CD3D06     03440           CALL    @PRT            ;Output it
2027 7E         03450           LD      A,(HL)
2028 E60F       03460           AND     0FH
202A C690       03470           ADD     A,90H
202C 27         03480           DAA
202D CE40       03490           ADC     A,40H
202F 27         03500           DAA
2030 CD3D06     03510           CALL    @PRT            ;Output it
2033 3E20       03520           LD      A,20H
2035 CD3D06     03530           CALL    @PRT            ;  & a space
2038 23         03540           INC     HL              ;Pt to next byte
2039 7D         03550           LD      A,L             ;Test multiple of 16
203A E60F       03560           AND     0FH
203C 2809       03570           JR      Z,PRI5
203E E603       03580           AND     3               ;Space on multiple of 4
2040 3E20       03590           LD      A,20H
2042 CC3D06     03600           CALL    Z,@PRT
2045 18D0       03610           JR      PRI4
2047 3E20       03620 PRI5      LD      A,20H           ;Space at end of 16
2049 CD3D06     03630           CALL    @PRT
204C E1         03640           POP     HL
204D 7E         03650 PRI6      LD      A,(HL)          ;Print in ASCII if
204E FE20       03660           CP      20H             ;  printable; else
2050 3804       03670           JR      C,PRI7          ;  convert to '.'
2052 FE80       03680           CP      80H
2054 3802       03690           JR      C,PRI8
2056 3E2E       03700 PRI7      LD      A,'.'
```

```
2058 CD3D06     03710 PRI8      CALL    @PRT
205B 23         03720           INC     HL              ;Loop until 16 chars
205C 7D         03730           LD      A,L
205D E60F       03740           AND     0FH
205F 20EC       03750           JR      NZ,PRI6
2061 3E0D       03760           LD      A,0DH           ;  then a new line
2063 CD3D06     03770           CALL    @PRT
2066 E5         03780           PUSH    HL
2067 7D         03790           LD      A,L             ;Check if HL is 0000
2068 B4         03800           OR      H
2069 2003       03810           JR      NZ,PRI9         ;  is OK > continue
206B E1         03820           POP     HL
206C 1806       03830           JR      PRI10           ;Get OUT now
206E AF         03840 PRI9      XOR     A               ;Ck on finished
206F ED42       03850           SBC     HL,BC
2071 E1         03860           POP     HL
2072 38A1       03870           JR      C,PRI3
2074 3E0D       03880 PRI10     LD      A,0DH           ;3 new lines if done
2076 CD3D06     03890           CALL    @PRT
2079 CD3D06     03900           CALL    @PRT
207C C33D06     03910           JP      @PRT
207F            03920 LAST      EQU     $
                03930           IFGT    $,DIRBUF$
                03940           ERR     'Module too big'
                03950           ENDIF
23FE            03960           ORG     MAXCOR$-2
23FE 7F02       03970           DW      LAST-SYS9       ;Overlay size
                03980 ;
1E00            03990           END     SYS9
```

```
$?1            1E32 $?10           1F1D $?11           1F2E
$?12           1F38 $?13           1F8F $?14           1F9B
$?15           1F9F $?16           1FA4 $?17           1FC5
$?18           1FDF $?19           200F $?2            1E37
$?20           2057 $?21           205C $?22           2061
$?23           2062 $?24           2066 $?25           20A6
$?26           20A9 $?27           20AA $?28           20B7
$?28A          20F1 $?29           20F6 $?3            1E49
$?30           20F9 $?31           20FC $?32           2102
$?33           210B $?34           2117 $?35           211A
$?36           2180 $?37           218E $?38           219A
$?39           219C $?4            1EB4 $?40           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            03B7 $A2            03B8 $A3            03B9
$CKEOF         1470 @$SYS          08F0 @@1            0000
@@2            0000 @@3            0000 @@4            0000
@ABORT         1B08 @ADTSK         1CDA @BANK          0877
@BKSP          1486 @BREAK         196F @BYTEIO        1300
@CHNIO         0689 @CKBRKC        0553 @CKDRV         1993
@CKEOF         158F @CKTSK         1CF5 @CLOSE         1999
@CLS           0545 @CMNDI         197E @CMNDR         197B
@CTL           0623 @DATE          07A8 @DBGHK         199F
@DCINIT        19C0 @DCRES         19C4 @DCSTAT        19B5
@DCTBYT        1A2B @DEBUG         19A0 @DECHEX        03E1
@DIRCYL        18F7 @DIRRD         18BB @DIRWR         1803
@DIV16         06E3 @DIV8          1927 @DODIR         19AF
@DOKEY         19A9 @DSP           0642 @DSPLY         052D
@ERROR         1B0F @EXIT          1B0B @FEXT          1984
@FLAGS         196A @FNAME         199C @FRENCH        0000
@FSPEC         1981 @GATRD         1874 @GATWR         1875
@GERMAN        0000 @GET           0638 @GTDCB         1990
@GTDCT         1A1E @GTMOD         19B2 @HDFMT         19E4
@HEX16         07BD @HEX8          07C2 @HEXDEC        06F6
@HIGH$         1948 @HITRD         1897 @HITWR         1898
@HZ50          0000 @ICNFG         0086 @INIT          198D
@INTL          0000 @IPL           1BF2 @JCL           0630
@KBD           0635 @KEY           0628 @KEYIN         0585
@KITSK         0089 @KLTSK         1CD0 @LOAD          1B38
@LOC           14B3 @LOF           14DE @LOGER         0503
@LOGOT         0500 @MOD2          0000 @MOD4          FFFF
@MSG           0530 @MUL16         06C9 @MUL8          190A
@NMI           0066 @OPEN          198A @OPREG         0084
@PARAM         1987 @PAUSE         0382 @PEOF          14A2
@POSN          1434 @PRINT         0528 @PRT           063D
@PUT           0645 @RAMDIR        19AC @RDHDR         19D8
@RDSEC         19F4 @RDSSC         18D8 @RDTRK         19E0
@READ          1513 @REMOVE        19A6 @RENAME        1996
@REW           149B @RMTSK         1CD7 @RPTSK         1CEB
@RREAD         1473 @RSLCT         19D4 @RST00         0000
@RST08         0008 @RST10         0010 @RST18         0018
@RST20         0020 @RST28         0028 @RST30         0030
@RST38         0038 @RSTNMI        0FE9 @RSTOR         19C8
@RSTREG        0680 @RUN           1B1D @RWRIT         13AD
@SEEK          19D0 @SEEKSC        1421 @SKIP          1430
@SLCT          19BC @SOUND         0392 @STEPI         19CC
@TIME          078D @USA           FFFF @VDCTL         0B99
@VDCTL3        0D38 @VER           1560 @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$          002B TIMTSK$         0713 TMPMT$          04C3
TRACE_INT       07B1 TYP1            1EE6 TYP2            1EEB
TYP5            1F01 TYPE            1ED3 TYPHK$          0A8F
TYPTSK$         0B26 USTOR$          0013 VER1            1F20
VER2            1F37 VER3            1F46 VER4            1F4F
VER5            1F5A VERIFY          1F07 VFLAG$          007F
WOR1            1F85 WOR2            1F96 WOR3            1F99
WOR4            1F9F WORD            1F62 WR1HEX@         2211
WR2HEX@         2215 WRINT$          0080 WRSPA@          2231
XY_TAB          2157 ZERO$           0401 ZEROA@          13A0

1E00 is the transfer address
00000 Total errors
```

NOTES:

SYS10/SYS

SYS10/SYS will kill a file or a device. It contains code for the SVC @REMOV.

```
                00100 ;SYS10/ASM - LS-DOS 6.2
0000            00110          TITLE   <SYS10 - LS-DOS 6.2>
                00120 ;
000D            00130 CR       EQU     13
                00140 ;
                00150 *LIST    OFF                     ;Get SYS0/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 SYS10    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 C0       00680          RET     NZ               ;Ret if read/write error
1E53 110000   00690 EXTINFO  LD      DE,0             ;P/u last extent info
              00700 ;
              00710 ;        If extended directory record in use,
              00720 ;        D -> DEC of FXDE record
              00730 ;        E -> FE if FXDE, FF if extent unused
              00740 ;
1E56 42       00750          LD      B,D              ;Ck for FXDE in use
1E57 7B       00760          LD      A,E
1E58 FEFE     00770          CP      0FEH             ;X'FE' => FXDE in use
1E5A 28C8     00780          JR      Z,REMOV2         ;Jump if FXDE in use
1E5C CD7518   00790          CALL    @GATWR           ;  else write the GAT
1E5F C0       00800          RET     NZ               ;Ret if write error
1E60 DDE5     00810          PUSH    IX               ;Transfer FCB address
1E62 E1       00820          POP     HL               ;  to HL & zero out FCB
1E63 0620     00830          LD      B,32             ;Init for 32 byte field
1E65 AF       00840          XOR     A                ;Zero the accumulator
1E66 77       00850 ZERLP1   LD      (HL),A           ;Go for it!
1E67 23       00860          INC     HL
1E68 10FC     00870          DJNZ    ZERLP1
1E6A C9       00880          RET
              00890 ;
              00900 ;        REMOVE will only close a logical device
              00910 ;
1E6B FE10     00920 CLOSDCB  CP      10H              ;Is this an open DCB?
1E6D 3E26     00930          LD      A,38             ;Init "file not open
1E6F C0       00940          RET     NZ
1E70 CD6615   00950          CALL    LNKFCB@          ;Link to DCB (DE->IX)
1E73 DD4E06   00960          LD      C,(IX+6)         ;Get device name
1E76 DD4607   00970          LD      B,(IX+7)
1E79 DD36002A 00980          LD      (IX+0),'*'       ;Stuff device indicator
1E7D DD7101   00990          LD      (IX+1),C         ;Stuff 1st char of name
1E80 DD7002   01000          LD      (IX+2),B         ;Stuff 2nd char of name
1E83 DD360303 01010          LD      (IX+3),3         ;Terminate with ETX
1E87 AF       01020          XOR     A
1E88 C9       01030          RET
              01040 ;
              01050 ;        Deallocate an extent
              01060 ;
1E89 E5       01070 RMVEXT   PUSH    HL
1E8A C5       01080          PUSH    BC
1E8B 3E08     01090          LD      A,8              ;P/u the # of grans per
1E8D CD2B1A   01100          CALL    @DCTBYT          ;  cylinder into reg A
1E90 07       01110          RLCA                     ;Shift into bits 0-2
1E91 07       01120          RLCA
1E92 07       01130          RLCA
1E93 E607     01140          AND     7                ;Remove all else
1E95 3C       01150          INC     A                ;Adjust for zero offset
              01160 ;
              01170 ;        Ck for 2-sided operation
              01180 ;
1E96 6F       01190          LD      L,A              ;Save current grans/cyl
1E97 3E04     01200          LD      A,4
1E99 CD2B1A   01210          CALL    @DCTBYT          ;Get 2-sided flag
1E9C CB6F     01220          BIT     5,A              ;Test 2-sided
1E9E 7D       01230          LD      A,L              ;Xfer value back
1E9F 2801     01240          JR      Z,$+3            ;Bypass if 1-sided
1EA1 87       01250          ADD     A,A              ;  else multiply by 2
1EA2 32BA1E   01260          LD      (GRNSCYL+1),A    ;Modify later instruction
1EA5 6B       01270          LD      L,E              ;Relative cylinder -> L
1EA6 2623     01280          LD      H,DIRBUF$<-8     ;Point to GAT byte
```

```
1EA8 7A         01290          LD      A,D             ;Rel gran & # of grans
1EA9 E61F       01300          AND     1FH             ;Get # of grans
1EAB 4F         01310          LD      C,A             ;  into reg C & adjust
1EAC 0C         01320          INC     C               ;  for zero offset
1EAD AA         01330          XOR     D               ;Get rel gran & shift
1EAE 07         01340          RLCA                    ;  into bits 0-2
1EAF 07         01350          RLCA
1EB0 07         01360          RLCA
1EB1 F5         01370 RMVEX1   PUSH    AF              ;Save rel starting gran
1EB2 46         01380          LD      B,(HL)          ;P/u allocation byte
1EB3 CDC51E     01390          CALL    RMVGRN          ;Turn off bit for a gran
1EB6 70         01400          LD      (HL),B          ;Update GAT byte
1EB7 F1         01410          POP     AF              ;Recover starting gran
1EB8 3C         01420          INC     A               ;Bump up
1EB9 FE00       01430 GRNSCYL  CP      0               ;Ck with grans per cyl
1EBB 2002       01440          JR      NZ,DECGRNS      ;Go if still on this cyl
1EBD AF         01450          XOR     A               ;  else zero gran counter
1EBE 2C         01460          INC     L               ;Bump to next cyl in GAT
1EBF 0D         01470 DECGRNS  DEC     C               ;Decrement # of grans
1EC0 20EF       01480          JR      NZ,RMVEX1       ;Go if more to deallocate
1EC2 C1         01490          POP     BC              ;  else recover regs
1EC3 E1         01500          POP     HL              ;  & go home
1EC4 C9         01510          RET
                01520 ;
                01530 ;        Remove a bit to deallocate & free up a gran
                01540 ;
1EC5 E607       01550 RMVGRN   AND     7               ;Max 8-grans per cyl
1EC7 07         01560          RLCA                    ;Shift to create RES
1EC8 07         01570          RLCA
1EC9 07         01580          RLCA
1ECA F680       01590          OR      80H             ;Merge rest of RES code
1ECC 32D01E     01600          LD      (RMVGRN1+1),A   ;Stuff into the inst
1ECF CB80       01610 RMVGRN1  RES     0,B             ;Reset proper bit
1ED1 C9         01620          RET
                01630 ;
1ED2            01640 LAST     EQU     $
                01650          IFGT    $,DIRBUF$
                01660          ERR     'Module too big'
                01670          ENDIF
23FE            01680          ORG     MAXCOR$-2
23FE D200       01690          DW      LAST-SYS10      ;Overlay size
                01700 ;
1E00            01710          END     SYS10
```

```
$A1             03B7 $A2             03B8 $A3             03B9
$CKEOF          1470 @$SYS           08F0 @@1             0000
@@2             0000 @@3             0000 @@4             0000
@ABORT          1B08 @ADTSK          1CDA @BANK           0877
@BKSP           1486 @BREAK          196F @BYTEIO         1300
@CHNIO          0689 @CKBRKC         0553 @CKDRV          1993
@CKEOF          158F @CKTSK          1CF5 @CLOSE          1999
@CLS            0545 @CMNDI          197E @CMNDR          197B
@CTL            0623 @DATE           07A8 @DBGHK          199F
@DCINIT         19C0 @DCRES          19C4 @DCSTAT         19B5
@DCTBYT         1A2B @DEBUG          19A0 @DECHEX         03E1
@DIRCYL         18F7 @DIRRD          18BB @DIRWR          1803
@DIV16          06E3 @DIV8           1927 @DODIR          19AF
@DOKEY          19A9 @DSP            0642 @DSPLY          052D
@ERROR          1B0F @EXIT           1B0B @FEXT           1984
@FLAGS          196A @FNAME          199C @FRENCH         0000
@FSPEC          1981 @GATRD          1874 @GATWR          1875
@GERMAN         0000 @GET            0638 @GTDCB          1990
@GTDCT          1A1E @GTMOD          19B2 @HDFMT          19E4
@HEX16          07BD @HEX8           07C2 @HEXDEC         06F6
@HIGH$          1948 @HITRD          1897 @HITWR          1898
@HZ50           0000 @ICNFG          0086 @INIT           198D
@INTL           0000 @IPL            1BF2 @JCL            0630
@KBD            0635 @KEY            0628 @KEYIN          0585
@KITSK          0089 @KLTSK          1CD0 @LOAD           1B38
@LOC            14B3 @LOF            14DE @LOGER          0503
@LOGOT          0500 @MOD2           0000 @MOD4           FFFF
@MSG            0530 @MUL16          06C9 @MUL8           190A
@NMI            0066 @OPEN           198A @OPREG          0084
@PARAM          1987 @PAUSE          0382 @PEOF           14A2
@POSN           1434 @PRINT          0528 @PRT            063D
@PUT            0645 @RAMDIR         19AC @RDHDR          19D8
@RDSEC          19F4 @RDSSC          18D8 @RDTRK          19E0
@READ           1513 @REMOVE         19A6 @RENAME         1996
@REW            149B @RMTSK          1CD7 @RPTSK          1CEB
@RREAD          1473 @RSLCT          19D4 @RST00          0000
@RST08          0008 @RST10          0010 @RST18          0018
@RST20          0020 @RST28          0028 @RST30          0030
@RST38          0038 @RSTNMI         0FE9 @RSTOR          19C8
@RSTREG         0680 @RUN            1B1D @RWRIT          13AD
@SEEK           19D0 @SEEKSC         1421 @SKIP           1430
@SLCT           19BC @SOUND          0392 @STEPI          19CC
@TIME           078D @USA            FFFF @VDCTL          0B99
@VDCTL3         0D38 @VER            1560 @VRSEC          19DC
@WEOF           14EC @WHERE          1979 @WRITE          1531
@WRSEC          19E8 @WRSSC          19EC @WRTRK          19F0
@_VDCTL         0D42 ADDR_2_ROWCOL   0DF1 AFLAG$          006A
AUTO?           1FF1 BAR$            0201 BOOTST$         439D
BREAK?          1C60 BRKVEC$         1C88 BUR$            0200
CASHK$          0A7B CFCB$           00E0 CFGFCB$         00E0
CFLAG$          006C CKMOD@          1A7F CKOPEN@         1568
CLOSDCB         1E6B CONFIG$         203F CORE$           0300
CR              000D CRTBGN$         F800 CYL_GRN         16AE
D@FBYT8         1A26 DATE$           0033 DAYTBL$         04C7
DBGSV$          00A0 DCBKL$          0031 DCT$            0470
DCTBYT8@        1A29 DCTFLD@         1A34 DECGRNS         1EBF
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 DTPMT$          04C2 DVREND$         0FF4
DVRHI$          0206 EFLAG$          006E ENADIS_DO_RAM   0817
EXTDBG$         19A4 EXTINFO         1E53 FDDINT$         000E
FEMSK$          006F FIXDIR          1E3E FLGTAB$         006A
GET_@_ROWCOL    0DAE GRNSCYL         1EB9 HERTZ$          0750
HIGH$           040E HKRES$          1A6C IFLAG$          0072
INBUF$          0420 INTIM$          003C INTMSK$         003D
INTVC$          003E JCLCB$          0203 JDCB$           0024
JFCB$           00C0 JLDCB$          0230 JRET$           0026
KCK@            07D6 KFLAG$          0074 KIDATA$         08FC
KIDCB$          0208 LAST            1ED2 LBANK$          0202
LDRV$           0023 LFLAG$          0075 LNKFCB@         1566
LOW$            001E LSVC$           000D MAXCOR$         2400
MAXDAY$         0401 MINCOR$         3000 MODOUT$         0076
MONTBL$         04DC NFLAG$          0077 OPREG$          0078
OPREG_SV_AREA   086E OPREG_SV_PTR    0835 ORARET@         14DC
OSRLS$          003B OSVER$          0085 OVRLY$          0069
PAKNAM$         0410 PAUSE@          0382 PCSAVE$         07AF
PDRV$           001B PHIGH$          001C PRDCB$          0218
PUTA@DE         0DCD PUT_@           0DCA PUT_@_ROWCOL    0DC6
REMOV1          1E1B REMOV2          1E24 REMOV3          1E2C
RFLAG$          007B RMVEX1          1EB1 RMVEXT          1E89
RMVGRN          1EC5 RMVGRN1         1ECF ROWCOL_2_ADDR   0DD0
RST38@          1BFF RSTOR$          04C4 RWRIT@          13A2
S1DCB$          0238 SBUFF$          1D00 SET@EXEC        1A79
SET_SCROLL      0CF3 SFCB$           008C SFLAG$          007C
SIDCB$          0220 SODCB$          0228 SPACE4$         2142
STACK$          0380 START$          0000 SVCRET$         000B
SVCTAB$         0100 SYS10           1E00 SYSERR$         1B13
TCB$            004E TFLAG$          007D TIME$           002D
TIMER$          002C TIMSL$          002B TIMTSK$         0713
TMPMT$          04C3 TRACE_INT       07B1 TYPHK$          0A8F
TYPTSK$         0B26 USTOR$          0013 VFLAG$          007F
WRINT$          0080 ZERLP1          1E66 ZERO$           0401
ZEROA@          13A0
1E00 is the transfer address
00000 Total errors
```

NOTES:

NOTES:

SYS11/SYS

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         01290          LD      H,A             ;Set = 0 for @EXIT
1EC7 6F         01300          LD      L,A
1EC8 110802     01310          LD      DE,KIDCB$       ;Clear any type ahead
1ECB 3E03       01320          LD      A,3
1ECD CD2306     01330          CALL    @CTL            ;  buffer (no streaming)
1ED0 3E93       01340          LD      A,93H           ;Restore @EXIT SVC
1ED2 320C1B     01350          LD      (@EXIT+1),A     ;  back to SYS1
1ED5 C9         01360          RET
                01370 ;
                01380 ;        Keyboard request processor
                01390 ;
1ED6 210A00     01400 KEYREQ   LD      HL,10           ;Back stack up 5 words
1ED9 39         01410          ADD     HL,SP           ;SYS0,RET,DE,HL,IX,BC
1EDA 4E         01420          LD      C,(HL)          ;Get contents of BC
1EDB 23         01430          INC     HL              ;  prior to keyboard
1EDC 46         01440          LD      B,(HL)          ;  request & DRIVER save
                01450 ;
                01460 ;        @KEYIN is requesting an entire line
                01470 ;
1EDD 11C000     01480 KEYLINE  LD      DE,JFCB$        ;Ck on end of JCL file
1EE0 C5         01490          PUSH    BC
1EE1 CD8F15     01500          CALL    @CKEOF
1EE4 C1         01510          POP     BC
1EE5 20CB       01520          JR      NZ,EXIT
1EE7 78         01530          LD      A,B             ;Do we need to re-read
1EE8 B9         01540          CP      C               ;  the JCL sector?
1EE9 C23806     01550          JP      NZ,@GET
1EEC CD7314     01560          CALL    @RREAD          ;Get the sector back
1EEF C20F1B     01570          JP      NZ,@ERROR
1EF2 CD3806     01580          CALL    @GET
1EF5 B7         01590          OR      A
1EF6 28BA       01600          JR      Z,EXIT
1EF8 FE2F       01610          CP      '/'             ;Is this line execution
1EFA 2802       01620          JR      Z,GOTSLSH       ;  JCL code to parse?
1EFC BF         01630          CP      A               ;Set Z-flag
1EFD C9         01640          RET
                01650 ;
                01660 ;        Found an execution code line
                01670 ;
1EFE C5         01680 GOTSLSH  PUSH    BC
1EFF D5         01690          PUSH    DE
1F00 064F       01700          LD      B,79            ;Only 79 char line
1F02 212004     01710          LD      HL,INBUF$       ;Get rest of line
1F05 E5         01720          PUSH    HL              ;  into JCL buffer
1F06 77         01730 GOTSL1   LD      (HL),A          ;Compare for CR as end
1F07 23         01740          INC     HL              ;  of line
1F08 FE0D       01750          CP      CR
1F0A 2807       01760          JR      Z,GOTSL2
1F0C CD3806     01770          CALL    @GET            ;Get a character
1F0F 10F5       01780          DJNZ    GOTSL1          ;  up to 79 max
1F11 183F       01790          JR      BADJCL          ;Line too long
1F13 E1         01800 GOTSL2   POP     HL              ;Rcvr pointer to buf
1F14 E5         01810          PUSH    HL
1F15 23         01820          INC     HL              ;Pt to 2nd char
1F16 7E         01830          LD      A,(HL)
1F17 FE2F       01840          CP      '/'             ;Found a //?
1F19 2032       01850          JR      NZ,REKEY2
1F1B 23         01860          INC     HL              ;Ck on ///
1F1C 96         01870          SUB     (HL)
1F1D CAC11F     01880          JP      Z,KEYIN6        ;Jump if ///
1F20 D6F6       01890          SUB     0F6H
```

```
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         02510          PUSH    HL
1F80 7E         02520 KEYIN1   LD      A,(HL)          ;Display JCL command line
1F81 23         02530          INC     HL
1F82 FE0D       02540          CP      CR
1F84 2805       02550          JR      Z,KEYIN2
1F86 CD4206     02560          CALL    @DSP
1F89 18F5       02570          JR      KEYIN1
1F8B CD2806     02580 KEYIN2   CALL    @KEY            ;Get & display the char
1F8E CD4206     02590          CALL    @DSP
1F91 32BE1F     02600          LD      (KEYIN5+1),A    ;Stuff for compare
1F94 3E0D       02610          LD      A,CR
1F96 CD4206     02620          CALL    @DSP            ;Write new line
1F99 E1         02630 KEYIN3   POP     HL
1F9A E5         02640          PUSH    HL
1F9B 11C000     02650          LD      DE,JFCB$        ;Ck for end of JCL
1F9E CD8F15     02660          CALL    @CKEOF
1FA1 C2B21E     02670          JP      NZ,EXIT
1FA4 CD3806     02680 KEYIN4   CALL    @GET            ;Xfer a line of JCL
1FA7 77         02690          LD      (HL),A          ;  to buffer
1FA8 23         02700          INC     HL
1FA9 FE0D       02710          CP      CR
1FAB 20F7       02720          JR      NZ,KEYIN4
1FAD E1         02730          POP     HL
1FAE E5         02740          PUSH    HL
1FAF 7E         02750          LD      A,(HL)          ;Look for // to find
1FB0 FE2F       02760          CP      '/'             ;Start of procedure block
1FB2 20E5       02770          JR      NZ,KEYIN3
1FB4 23         02780          INC     HL
1FB5 BE         02790          CP      (HL)            ;//?
1FB6 20E1       02800          JR      NZ,KEYIN3
1FB8 23         02810          INC     HL              ;Point to proc label
1FB9 96         02820          SUB     (HL)            ;Is label a '/' noting
1FBA 2805       02830          JR      Z,KEYIN6        ;  exec phase cond's end?
1FBC 7E         02840          LD      A,(HL)          ;Nope, get proc label
1FBD FE00       02850 KEYIN5   CP      0               ;Same as key entry?
1FBF 20D8       02860          JR      NZ,KEYIN3       ;No match? check next one
1FC1 32BE1F     02870 KEYIN6   LD      (KEYIN5+1),A    ;Stuff 0 if ///
1FC4 E1         02880          POP     HL
1FC5 E5         02890          PUSH    HL
1FC6 CD0305     02900          CALL    @LOGER          ;Log the command
1FC9 1882       02910          JR      REKEY2
                02920 ;
                02930 ;        Process //ALERT
                02940 ;
1FCB AF         02950 ALERT    XOR     A
1FCC 32FB1F     02960          LD      (ALERT4+1),A    ;Start with clean flag
1FCF 7E         02970 ALERT1   LD      A,(HL)          ;Ignore spaces
1FD0 23         02980          INC     HL
1FD1 FE20       02990          CP      ' '
1FD3 28FA       03000          JR      Z,ALERT1
1FD5 FE2C       03010          CP      ','             ;Comma separator?
1FD7 28F6       03020          JR      Z,ALERT1
1FD9 FE0D       03030          CP      CR              ;End of line?
1FDB CA4D1F     03040          JP      Z,REKEY2
1FDE FE29       03050          CP      ')'             ;Closing paren?
1FE0 2809       03060          JR      Z,ALERT2
1FE2 FE28       03070          CP      '('             ;Start of parms?
1FE4 200F       03080          JR      NZ,ALERT3       ;If none of the above...
1FE6 22EC1F     03090          LD      (ALERT2+1),HL   ;Save ptr to parm start
1FE9 18E4       03100          JR      ALERT1
                03110 ;
                03120 ;         Check here when closing parm received
                03130 ;
1FEB 210000     03140 ALERT2    LD      HL,0            ;P/u ptr to '(' if there
1FEE 7C         03150           LD      A,H             ;If the //ALERT1 started
1FEF B5         03160           OR      L               ;  with a '(', then
1FF0 20DD       03170           JR      NZ,ALERT1       ;  repeat the parm
1FF2 C3521F     03180           JP      BADJCL          ;  parsing else exit
                03190 ;
                03200 ;         Assumed integer parm found
                03210 ;
1FF5 2B         03220 ALERT3    DEC     HL              ;Backup pointer
1FF6 CDE103     03230           CALL    @DECHEX         ;Cvrt value to binary
1FF9 41         03240           LD      B,C             ;Keep value as counter
1FFA 3E00       03250 ALERT4    LD      A,0             ;Flip flag: entries 1, 3,
1FFC EEFF       03260           XOR     0FFH            ;  5, ... are noise, 2,
1FFE 32FB1F     03270           LD      (ALERT4+1),A    ;  4, 6, ... are silence
2001 4F         03280           LD      C,A
2002 CB41       03290           BIT     0,C             ;Test noise or silence
2004 C49203     03300           CALL    NZ,@SOUND       ;Call for sound out
2007 CB41       03310           BIT     0,C             ;  then test again
2009 CC1420     03320           CALL    Z,SILENCE       ;Silence is golden
200C CD881E     03330           CALL    KSCN            ;Ck BREAK or ENTER
200F C24D1F     03340           JP      NZ,REKEY2       ;Go on enter
2012 18BB       03350           JR      ALERT1          ;Loop if not
                03360 ;
                03370 ;         Silence routine
                03380 ;
2014 B0         03390 SILENCE   OR      B               ;A was zero
2015 C8         03400           RET     Z
2016 CD1C20     03410           CALL    SILEN1          ;Delay a bit
2019 10F9       03420           DJNZ    SILENCE         ;  for duration
201B C9         03430           RET
201C C5         03440 SILEN1    PUSH    BC              ;Delay for 0.1 sec
201D 019B19     03450           LD      BC,6555
2020 CD8203     03460           CALL    @PAUSE
2023 C1         03470           POP     BC
2024 C9         03480           RET
                03490 ;
                03500 ;         Process //FLASH
                03510 ;
2025 CDE103     03520 FLASH     CALL    @DECHEX
2028 41         03530           LD      B,C             ;P/u the flash count
2029 E1         03540           POP     HL
202A E5         03550           PUSH    HL
202B C5         03560 FLASH1    PUSH    BC
202C CD2D05     03570           CALL    @DSPLY          ;Display the prompt
202F 010040     03580           LD      BC,4000H        ;Countdown to flash msg
2032 CD881E     03590 FLASH2    CALL    KSCN            ;Keep testing <ENTER>
2035 C24C1F     03600           JP      NZ,REKEY1       ;  key during countdown
2038 0B         03610           DEC     BC              ;BREAK would abort
2039 78         03620           LD      A,B
203A B1         03630           OR      C
203B 20F5       03640           JR      NZ,FLASH2       ;Loop until count=0
203D 3E1B       03650           LD      A,27            ;Erase the message line
203F CD4206     03660           CALL    @DSP
2042 3E1E       03670           LD      A,30
2044 CD4206     03680           CALL    @DSP
2047 CD1C20     03690           CALL    SILEN1          ;Delay while blanked
204A C1         03700           POP     BC
204B 10DE       03710           DJNZ    FLASH1
204D C34D1F     03720 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 327120    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         05560           LD      E,(HL)          ;Xfer vector to DE
2140 23         05570           INC     HL
2141 56         05580           LD      D,(HL)
2142 E1         05590           POP     HL
2143 AF         05600           XOR     A               ;  & show it found
2144 C9         05610           RET
                05620 ;
                05630 ;         No match if alphanumeric unless a space
                05640 ;
2145 CD0321     05650 FND6      CALL    TST09AZ         ;Ck for 0-9, A-Z
2148 3005       05660           JR      NC,FND8         ;Go if one of the above
214A 7E         05670 FND7      LD      A,(HL)          ;Loop if table has
214B FE20       05680           CP      ' '             ;  trailing spaces
214D 28EA       05690           JR      Z,FND5
214F D1         05700 FND8      POP     DE
2150 E1         05710           POP     HL
2151 18C5       05720           JR      FND2
                05730 ;
0006            05740 LILBUF    DS      6
2153            05750 TIMFLD    EQU     LILBUF
2159 42         05760 BADJCL$ DB        'Bad JCL, '
     61 64 20 4A 43 4C 2C 20
2162 4A         05770 ABORT$  DB        'Job aborted',CR
     6F 62 20 61 62 6F 72 74
     65 64 0D
216E 4A         05780 JOBDUN$ DB        'Job done',CR
     6F 62 20 64 6F 6E 65 0D
2177 41         05790 PARMTBL DB        'ABORT'
     42 4F 52 54
217C 801E       05800           DW      ABORT
217E 41         05810           DB      'ALERT'
     4C 45 52 54
2183 CB1F       05820           DW      ALERT
2185 44         05830           DB      'DELAY'
     45 4C 41 59
218A 621F       05840           DW      DELAY
218C 45         05850           DB      'EXIT '
     58 49 54 20
2191 B21E       05860           DW      EXIT
2193 46         05870           DB      'FLASH'
     4C 41 53 48
2198 2520       05880           DW      FLASH
219A 4B         05890           DB      'KEYIN'
     45 59 49 4E
219F 7E1F       05900           DW      KEYIN
21A1 50         05910           DB      'PAUSE'
     41 55 53 45
21A6 721F       05920           DW      PAUSE
21A8 53         05930           DB      'SLEEP'
     4C 45 45 50
21AD 5020       05940           DW      SLEEP
21AF 53         05950           DB      'STOP '
     54 4F 50 20
21B4 581F       05960           DW      STOP
21B6 57         05970           DB      'WAIT '
     41 49 54 20
21BB 5120       05980           DW      WAIT
21BD 49         05990           DB      'INPUT'
     4E 50 55 54
21C2 A920       06000           DW      INPUT
21C4 00         06010           NOP
```

```
21C5           06020 LAST     EQU     $
               06030          IFGT    $,DIRBUF$
               06040          ERR     'Module too big'
               06050          ENDIF
23FE           06060          ORG     MAXCOR$-2
23FE C503      06070          DW      LAST-SYS11      ;Overlay size
               06080 ;
1E00           06090          END     SYS11
```

```
$?1             1E74 $A1            03B7 $A2            03B8
$A3             03B9 $CKEOF         1470 @$SYS          08F0
@@1             0000 @@2            0000 @@3            0000
@@4             0000 @ABORT         1B08 @ADTSK         1CDA
@BANK           0877 @BKSP          1486 @BREAK         196F
@BYTEIO         1300 @CHNIO         0689 @CKBRKC        0553
@CKDRV          1993 @CKEOF         158F @CKTSK         1CF5
@CLOSE          1999 @CLS           0545 @CMNDI         197E
@CMNDR          197B @CTL           0623 @DATE          07A8
@DBGHK          199F @DCINIT        19C0 @DCRES         19C4
@DCSTAT         19B5 @DCTBYT        1A2B @DEBUG         19A0
@DECHEX         03E1 @DIRCYL        18F7 @DIRRD         18BB
@DIRWR          1803 @DIV16         06E3 @DIV8          1927
@DODIR          19AF @DOKEY         19A9 @DSP           0642
@DSPLY          052D @ERROR         1B0F @EXIT          1B0B
@FEXT           1984 @FLAGS         196A @FNAME         199C
@FRENCH         0000 @FSPEC         1981 @GATRD         1874
@GATWR          1875 @GERMAN        0000 @GET           0638
@GTDCB          1990 @GTDCT         1A1E @GTMOD         19B2
@HDFMT          19E4 @HEX16         07BD @HEX8          07C2
@HEXDEC         06F6 @HIGH$         1948 @HITRD         1897
@HITWR          1898 @HZ50          0000 @ICNFG         0086
@INIT           198D @INTL          0000 @IPL           1BF2
@JCL            0630 @KBD           0635 @KEY           0628
@KEYIN          0585 @KITSK         0089 @KLTSK         1CD0
@LOAD           1B38 @LOC           14B3 @LOF           14DE
@LOGER          0503 @LOGOT         0500 @MOD2          0000
@MOD4           FFFF @MSG           0530 @MUL16         06C9
@MUL8           190A @NMI           0066 @OPEN          198A
@OPREG          0084 @PARAM         1987 @PAUSE         0382
@PEOF           14A2 @POSN          1434 @PRINT         0528
@PRT            063D @PUT           0645 @RAMDIR        19AC
@RDHDR          19D8 @RDSEC         19F4 @RDSSC         18D8
@RDTRK          19E0 @READ          1513 @REMOVE        19A6
@RENAME         1996 @REW           149B @RMTSK         1CD7
@RPTSK          1CEB @RREAD         1473 @RSLCT         19D4
@RST00          0000 @RST08         0008 @RST10         0010
@RST18          0018 @RST20         0020 @RST28         0028
@RST30          0030 @RST38         0038 @RSTNMI        0FE9
@RSTOR          19C8 @RSTREG        0680 @RUN           1B1D
@RWRIT          13AD @SEEK          19D0 @SEEKSC        1421
@SKIP           1430 @SLCT          19BC @SOUND         0392
@STEPI          19CC @TIME          078D @USA           FFFF
@VDCTL          0B99 @VDCTL3        0D38 @VER           1560
@VRSEC          19DC @WEOF          14EC @WHERE         1979
@WRITE          1531 @WRSEC         19E8 @WRSSC         19EC
@WRTRK          19F0 @_VDCTL        0D42 ABORT          1E80
ABORT$          2162 ADDR_2_ROWCOL  0DF1 AFLAG$         006A
ALERT           1FCB ALERT1         1FCF ALERT2         1FEB
ALERT3          1FF5 ALERT4         1FFA AUTOA          1FF1
BADJCL          1F52 BADJCL$        2159 BAR$           0201
BOOTST$         439D BREAKA         1C60 BRKVEC$        1C88
BUR$            0200 CASHK$         0A7B CFCB$          00E0
CFGFCB$         00E0 CFLAG$         006C CKMOD@         1A7F
CKOPEN@         1568 CKTIME         209F CONFIG$        203F
CORE$           0300 CR             000D CRTBGN$        F800
CYL_GRN         16AE D@FBYT8        1A26 DATE$          0033
DAYTBL$         04C7 DBGSV$         00A0 DCBKL$         0031
DCT$            0470 DCTBYT8@       1A29 DCTFLD@        1A34
DELAY           1F62 DELAY1         1F6B DFLAG$         006D
```

```
DIRBUF$           2300 DIS_DO_RAM        0846 DODATA$           0B94
DODCB$            0210 DOOFF             1EBC 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 DTPMT$            04C2 DVREND$           0FF4
DVRHI$            0206 EFLAG$            006E ENADIS_DO_RAM     0817
EXIT              1EB2 EXIT1             1EB8 EXITC             210F
EXTDBG$           19A4 FDDINT$           000E FEMSK$            006F
FLASH             2025 FLASH1            202B FLASH2            2032
FLASH3            204D FLGTAB$           006A FND1              2114
FND2              2118 FND3              2125 FND4              2129
FND5              2139 FND6              2145 FND7              214A
FND8              214F FNDPARM           2111 GETKEY            20B4
GETLINE           1E77 GET_@_ROWCOL      0DAE GOTSL1            1F06
GOTSL2            1F13 GOTSL3            1F2A GOTSLSH           1EFE
HERTZ$            0750 HIGH$             040E HKRES$            1A6C
IFLAG$            0072 INBUF$            0420 INPUT             20A9
INTIM$            003C INTMSK$           003D INTVC$            003E
IPLDO1            1E2F IPLDO2            1E4F JCLCB$            0203
JDCB$             0024 JFCB$             00C0 JLDCB$            0230
JOBDUN$           216E JRET$             0026 KCK@              07D6
KEYIN             1F7E KEYIN1            1F80 KEYIN2            1F8B
KEYIN3            1F99 KEYIN4            1FA4 KEYIN5            1FBD
KEYIN6            1FC1 KEYLINE           1EDD KEYREQ            1ED6
KFLAG$            0074 KIDATA$           08FC KIDCB$            0208
KSCN              1E88 KSCN1             1E96 KSCN2             1E99
LAST              21C5 LBANK$            0202 LDRV$             0023
LF                000A LFLAG$            0075 LILBUF            2153
LNKFCB@           1566 LOW$              001E LSVC$             000D
MAXCOR$           2400 MAXDAY$           0401 MINCOR$           3000
MODOUT$           0076 MONTBL$           04DC NEWEX1            1E69
NEWEXIT           1E55 NFLAG$            0077 OPREG$            0078
OPREG_SV_AREA     086E OPREG_SV_PTR      0835 ORARET@           14DC
OSRLS$            003B OSVER$            0085 OVRLY$            0069
PAKNAM$           0410 PAKTIM            2061 PAKTIM1           2066
PAR1              20CA PAR2              20CF PAR3              20EC
PAR4              20F7 PAR5              20FA PAR6              20FE
PARMTBL           2177 PARSER            20C8 PAUSE             1F72
PAUSE1            1F77 PAUSE@            0382 PCSAVE$           07AF
PDRV$             001B PHIGH$            001C PRDCB$            0218
PUTA@DE           0DCD PUT_@             0DCA PUT_@_ROWCOL      0DC6
REKEY1            1F4C REKEY2            1F4D RFLAG$            007B
ROWCOL_2_ADDR     0DD0 RST38@            1BFF RSTOR$            04C4
RWRIT@            13A2 S1DCB$            0238 SBUFF$            1D00
SET@EXEC          1A79 SET_SCROLL        0CF3 SFCB$             008C
SFLAG$            007C SIDCB$            0220 SILEN1            201C
SILENCE           2014 SLEEP             2050 SLP1              207D
SLP2              2088 SLPWT             2070 SODCB$            0228
SPACE4$           2142 STACK$            0380 START$            0000
STOP              1F58 SVCRET$           000B SVCTAB$           0100
SYS11             1E00 SYSERR$           1B13 TCB$              004E
TFLAG$            007D TIME$             002D TIMER$            002C
TIMFLD            2153 TIMSL$            002B TIMTSK$           0713
TMPMT$            04C3 TRACE_INT         07B1 TST09AZ           2103
TSTIME            2094 TYPHK$            0A8F TYPTSK$           0B26
UNHOOK            20C1 USTOR$            0013 VFLAG$            007F
WAIT              2051 WRINT$            0080 ZERO$             0401
ZEROA@            13A0
1E00 is the transfer address
00000 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 321620     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 CD3620     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         00680         INC     HL
1E3A 72         00690         LD      (HL),D
1E3B 23         00700         INC     HL
1E3C 71         00710         LD      (HL),C          ;Stuff "free to use"
1E3D 23         00720         INC     HL
1E3E 70         00730         LD      (HL),B
1E3F AF         00740         XOR     A               ;Show no error
1E40 C9         00750         RET
                00760 ;
                00770 ;       Do RAMDIR directory info
                00780 ;
1E41 05         00790 DIRINFO DEC     B               ;If DEC=0, do it all
1E42 286C       00800         JR      Z,DOALL         ;Go if all of it
1E44 04         00810         INC     B               ;1=>2, 2=>3, ..., FE=>FF
                00820 ;
                00830 ;       Calculate the number of directory sectors
                00840 ;       = (#sectors x #heads) - 2 for GAT & HIT
                00850 ;
1E45 3E07       00860         LD      A,7             ;Get highest # sector
1E47 CD2B1A     00870         CALL    @DCTBYT
1E4A 57         00880         LD      D,A             ;Store heads & sectors
1E4B E61F       00890         AND     1FH             ;Rake off # sectors
1E4D 5F         00900         LD      E,A             ;  & stuff into E
1E4E 1C         00910         INC     E               ;Bump for 0 offset
1E4F AA         00920         XOR     D               ;Recover # heads
1E50 07         00930         RLCA                    ;  into bits 0-2
1E51 07         00940         RLCA
1E52 07         00950         RLCA
1E53 3C         00960         INC     A               ;Bump for 0 offset
1E54 CD0A19     00970         CALL    @MUL8           ;Multiply sectors x heads
1E57 5F         00980         LD      E,A             ;Now check double bit
1E58 3E04       00990         LD      A,4
1E5A CD2B1A     01000         CALL    @DCTBYT
1E5D CB6F       01010         BIT     5,A             ;Set if 2-sided
1E5F 7B         01020         LD      A,E
1E60 2801       01030         JR      Z,ONESID        ;Go if not set else
1E62 87         01040         ADD     A,A             ;  double value
1E63 D602       01050 ONESID  SUB     2               ;Reduce for GAT & HIT
1E65 57         01060         LD      D,A             ;D => # dir sectors
1E66 78         01070         LD      A,B             ;Get requested DEC
1E67 E61F       01080         AND     1FH
1E69 BA         01090         CP      D               ;See if in range
1E6A 3804       01100         JR      C,DIRINF1       ;Go if so
1E6C 3E10       01110         LD      A,16            ;"Illegal logical file #
1E6E B7         01120         OR      A               ;Return out of range error
1E6F C9         01130         RET
                01140 ;
1E70 E5         01150 DIRINF1 PUSH    HL              ;Save buffer ptr
1E71 CDBB18     01160         CALL    @DIRRD          ;Get its directory record
1E74 D1         01170         POP     DE              ;Rcvr buf ptr
1E75 C0         01180         RET     NZ              ;Back on an error
1E76 7E         01190         LD      A,(HL)          ;Get attributes
1E77 E6D8       01200         AND     0D8H            ;Only if in use & VIS
1E79 EE10       01210         XOR     10H             ;Flip state so NZ=no
1E7B 3E19       01220         LD      A,25            ;Init file access denied
1E7D C0         01230         RET     NZ              ;Back on no file, SYS, INV
1E7E E5         01240 GETSTUF PUSH    HL              ;Save DIR ptr
1E7F CDE11F     01250         CALL    STUFBUF         ;Stuff the filespec
1E82 E1         01260         POP     HL
1E83 7E         01270         LD      A,(HL)
1E84 E607       01280         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 321620   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
```

```
1F30 C620       02510          ADD    A,32
1F32 6F         02520          LD     L,A
1F33 3018       02530          JR     NC,MDIR3        ;Bypass if still same
1F35 2C         02540          INC    L               ;  else point to next one
1F36 CB6D       02550          BIT    5,L             ;Finished with
1F38 2813       02560          JR     Z,MDIR3         ;  this drive?
1F3A 3A6B1F     02570          LD     A,(TSTOPT+1)    ;If option 1 or 3,
1F3D E601       02580          AND    1               ;  must stuff buffer end
1F3F 2007       02590          JR     NZ,CLSBUF
1F41 3E0D       02600          LD     A,CR            ;  else do a blank line
1F43 CD4206     02610          CALL   @DSP
1F46 AF         02620          XOR    A
1F47 C9         02630          RET
                02640 ;
1F48 3EFF       02650 CLSBUF   LD     A,0FFH          ;Put in buffer terminator
1F4A 12         02660          LD     (DE),A
1F4B AF         02670          XOR    A
1F4C C9         02680          RET
                02690 ;
1F4D 7E         02700 MDIR3    LD     A,(HL)          ;P/u HIT entry
1F4E B7         02710          OR     A
1F4F 28DE       02720          JR     Z,MDIR2         ;Jump if spare
1F51 45         02730          LD     B,L             ;Save DEC in reg B
1F52 C5         02740          PUSH   BC              ;  & to stack
1F53 7D         02750          LD     A,L             ;Pt to dir record for
1F54 E6E0       02760          AND    0E0H            ;  this DEC
1F56 6F         02770          LD     L,A             ;Get the dir sector for
1F57 A8         02780          XOR    B               ;  this DEC
1F58 FEFF       02790 MDIR4    CP     0FFH            ;Same as one in core?
1F5A 2808       02800          JR     Z,MDIR5         ;Jump if so
1F5C 32591F     02810          LD     (MDIR4+1),A     ;Else update one we have
1F5F CDBB18     02820          CALL   @DIRRD          ;  and read it into buf
1F62 205D       02830          JR     NZ,MDIR12       ;Jump on read error
1F64 261D       02840 MDIR5    LD     H,SBUFF$<-8     ;Sysbuf hi order
1F66 012B1F     02850          LD     BC,MDIR1        ;Set up the return addr
1F69 C5         02860          PUSH   BC
1F6A 3E00       02870 TSTOPT   LD     A,0             ;P/u option number
1F6C E5         02880          PUSH   HL
1F6D D5         02890          PUSH   DE
1F6E CDC31F     02900          CALL   TSTSAM          ;Check for extension match
1F71 D1         02910          POP    DE
1F72 E1         02920          POP    HL
1F73 C0         02930          RET    NZ              ;Back to MDIR1
1F74 3A6B1F     02940          LD     A,(TSTOPT+1)
1F77 0F         02950          RRCA                   ;Test option 1 or 3
1F78 7E         02960          LD     A,(HL)
1F79 3013       02970          JR     NC,DSPLYIT      ;Go if 0 or 2
1F7B E690       02980          AND    90H             ;Test FXDE & in-use
1F7D EE10       02990          XOR    10H             ;If not used, FXDE
1F7F C0         03000          RET    NZ              ;Back to MDIR1
1F80 011000     03010          LD     BC,16
1F83 EDB0       03020          LDIR                   ;User's buffer
1F85 2C         03030          INC    L               ;Bypass stored passwords
1F86 2C         03040          INC    L
1F87 2C         03050          INC    L
1F88 2C         03060          INC    L
1F89 0E02       03070          LD     C,2             ;  and xfer ERN
1F8B EDB0       03080          LDIR
1F8D C9         03090          RET                    ;Back to MDIR1
                03100 ;
1F8E E6D8       03110 DSPLYIT  AND    0D8H            ;Test if we want this
```

```
1F90 EE10     03120          XOR    10H             ;Only if in-use & VIS
1F92 C0       03130          RET    NZ              ;Back to MDIR1
1F93 11B421   03140          LD     DE,LILBUF+3
1F96 D5       03150          PUSH   DE
1F97 CDE11F   03160          CALL   STUFBUF         ;Move filespec to buffer
1F9A E1       03170          POP    HL              ;Rcvr LILBUF ptr
1F9B CD2D05   03180          CALL   @DSPLY          ;Display the file
1F9E 3E00     03190 MDIR11   LD     A,0             ;Count down 5-across
1FA0 3D       03200          DEC    A
1FA1 329F1F   03210          LD     (MDIR11+1),A    ;Update count
1FA4 C0       03220          RET    NZ              ;Loop if more to go
1FA5 3E05     03230          LD     A,5             ;  else re-init
1FA7 329F1F   03240          LD     (MDIR11+1),A
1FAA 3E0D     03250          LD     A,CR
1FAC CD4206   03260          CALL   @DSP            ;New line
1FAF 3E00     03270 CKPAGE   LD     A,0             ;P/u display count
1FB1 3D       03280          DEC    A
1FB2 32B01F   03290          LD     (CKPAGE+1),A
1FB5 C0       03300          RET    NZ
1FB6 3E17     03310          LD     A,23
1FB8 32B01F   03320          LD     (CKPAGE+1),A    ;Reset for max
1FBB CD2806   03330          CALL   @KEY            ;Wait for keyboard input
1FBE C34505   03340          JP     @CLS            ;Clear screen and ret
              03350 ;
1FC1 C1       03360 MDIR12   POP    BC
1FC2 C9       03370          RET
              03380 ;
1FC3 CB4F     03390 TSTSAM   BIT    1,A             ;Ck if /EXT option
1FC5 C8       03400          RET    Z               ;Ret with Z if
1FC6 010D00   03410          LD     BC,13           ;  option <> /EXT
1FC9 09       03420          ADD    HL,BC           ;Else point to /EXT
1FCA 0603     03430          LD     B,3             ;  field of dir record
1FCC 11B121   03440          LD     DE,LILBUF       ;  & check for match
1FCF 1A       03450 TSTS1    LD     A,(DE)
1FD0 FE24     03460          CP     '$'             ;'$' matches with all
1FD2 2808     03470          JR     Z,TSTS2
1FD4 FE41     03480          CP     'A'             ;If numeric, don't cvrt
1FD6 3802     03490          JR     C,$+4           ;  to upper case
1FD8 CBAF     03500          RES    5,A             ;Cvrt to UC if lc
1FDA BE       03510          CP     (HL)
1FDB C0       03520          RET    NZ              ;Ret on no match
1FDC 23       03530 TSTS2    INC    HL
1FDD 13       03540          INC    DE
1FDE 10EF     03550          DJNZ   TSTS1           ;Loop for 3 chars
1FE0 C9       03560          RET
              03570 ;
              03580 ;        Routine to construct the filespec field
              03590 ;
1FE1 7D       03600 STUFBUF  LD     A,L
1FE2 C605     03610          ADD    A,5             ;Pt to start of filename
1FE4 6F       03620          LD     L,A
1FE5 0E0D     03630          LD     C,13            ;Init for 15 (-2) chars
1FE7 0608     03640          LD     B,8             ;Filename
1FE9 7E       03650 STUFB1   LD     A,(HL)
1FEA 23       03660          INC    HL
1FEB FE20     03670          CP     ' '             ;Exit on 1st space
1FED 2807     03680          JR     Z,STUFB2
1FEF 12       03690          LD     (DE),A          ;Stuff the char
1FF0 13       03700          INC    DE
1FF1 0D       03710          DEC    C               ;String count down
1FF2 10F5     03720          DJNZ   STUFB1          ;Field loop
```

```
1FF4 1804       03730          JR      STUFB3           ;Bypass ext calculation
1FF6 7D         03740 STUFB2   LD      A,L              ;Calculate start of
1FF7 80         03750          ADD     A,B              ;EXT field in dir record
1FF8 3D         03760          DEC     A
1FF9 6F         03770          LD      L,A
1FFA 7E         03780 STUFB3   LD      A,(HL)           ;Display EXT if present
1FFB FE20       03790          CP      ' '
1FFD 2812       03800          JR      Z,STUFB5         ;Exit if no EXT
1FFF 3E2F       03810          LD      A,'/'            ;Display slash
2001 12         03820          LD      (DE),A           ;Stuff the char
2002 13         03830          INC     DE
2003 0D         03840          DEC     C                ;Dsplay char countdown
2004 0603       03850          LD      B,3              ;3 chars max for EXT
2006 7E         03860 STUFB4   LD      A,(HL)
2007 23         03870          INC     HL
2008 FE20       03880          CP      ' '
200A 2805       03890          JR      Z,STUFB5         ;Exit on 1st blank
200C 12         03900          LD      (DE),A           ;  else stuff the char
200D 13         03910          INC     DE
200E 0D         03920          DEC     C
200F 10F5       03930          DJNZ    STUFB4           ;Loop 3 chars
2011 3E3A       03940 STUFB5   LD      A,':'            ;Stuff a drive sep
2013 12         03950          LD      (DE),A           ;Reg C already accounted
2014 13         03960          INC     DE               ;  for in the init
2015 3E00       03970 DSTDRV   LD      A,0              ;P/u drive #
2017 12         03980          LD      (DE),A
2018 13         03990          INC     DE
2019 3E20       04000 STUFB6   LD      A,' '            ;Stuff a space
201B 12         04010          LD      (DE),A
201C 13         04020          INC     DE
201D 0D         04030          DEC     C                ;Count down
201E 20F9       04040          JR      NZ,STUFB6        ;Display trailing spaces
2020 3E03       04050          LD      A,3              ;Stuff the ETX
2022 12         04060          LD      (DE),A
2023 C9         04070          RET
                04080 ;
                04090 ;        Routine to get the free space info
                04100 ;
2024 E5         04110 SPACE0   PUSH    HL               ;Save buf start
2025 111000     04120          LD      DE,16            ;Index for space
2028 D5         04130          PUSH    DE
2029 19         04140          ADD     HL,DE
202A CD3620     04150          CALL    SPACE            ;Get the space data
202D C1         04160          POP     BC               ;  name & date
202E D1         04170          POP     DE               ;Now shift in the
202F 21D023     04180          LD      HL,DIRBUF$+0D0H  ;  disk name and date
2032 EDB0       04190          LDIR
2034 AF         04200          XOR     A
2035 C9         04210          RET
                04220 ;
2036 CD7418     04230 SPACE    CALL    @GATRD           ;Read GAT
2039 C0         04240          RET     NZ               ;Ret on GAT read error
203A FDE5       04250          PUSH    IY
203C CD1E1A     04260          CALL    @GTDCT           ;Get DCT vector
203F EB         04270          EX      DE,HL            ;User buf ptr to DE
2040 2600       04280          LD      H,0              ;P/u highest # cylinder
2042 FD6E06     04290          LD      L,(IY+6)         ;  & adjust for 0 offset
2045 23         04300          INC     HL
2046 FD7E08     04310          LD      A,(IY+8)         ;P/u # of sectors/granule
2049 E61F       04320          AND     1FH
204B 3C         04330          INC     A                ;Adjust for zero offset
```

```
204C F5         04340          PUSH    AF                 ;Save # of sectors/gran
204D D5         04350          PUSH    DE                 ;Save user buf ptr
204E 5F         04360          LD      E,A
204F FD7E08     04370          LD      A,(IY+8)           ;P/u # of granules/cyl
2052 E6E0       04380          AND     0E0H
2054 07         04390          RLCA                       ;  & shift to bits 0-2
2055 07         04400          RLCA
2056 07         04410          RLCA
2057 3C         04420          INC     A                  ;Adjust for zero offset
2058 CD0A19     04430          CALL    @MUL8              ;Calc # of sectors/cyl
205B FDCB046E   04440          BIT     5,(IY+4)           ;Double sided?
205F 2801       04450          JR      Z,$+3              ;Bypass  if one-sided
2061 87         04460          ADD     A,A                ;  else double the count
2062 C1         04470          POP     BC                 ;Rcvr user buf ptr
2063 CD8620     04480          CALL    DOMUL16            ;Calculate total sectors
2066 23         04490          INC     HL                 ;Bump to next buf pos
2067 E5         04500          PUSH    HL                 ;  & save pointer
2068 210023     04510          LD      HL,DIRBUF$         ;Pt to start of GAT
206B 110000     04520          LD      DE,0               ;Init gran counter
206E 3ACC23     04530          LD      A,(DIRBUF$+0CCH)         ;P/u cyl excess
2071 C623       04540          ADD     A,35               ;Add base
2073 47         04550          LD      B,A                ;Set loop counter
2074 7E         04560 PUGAT    LD      A,(HL)             ;P/u GAT byte
2075 37         04570 KEEP7    SCF                        ;Keep bit 7 set
2076 1F         04580          RRA                        ;Slide gran bit to carry
2077 3801       04590          JR      C,BYTEND?          ;Ignore if in use
2079 13         04600          INC     DE                 ;Free, bump gran counter
207A FEFF       04610 BYTEND?  CP      0FFH               ;End of byte?
207C 20F7       04620          JR      NZ,KEEP7           ;Loop if not
207E 2C         04630          INC     L                  ;Bump GAT byte pointer
207F 10F3       04640          DJNZ    PUGAT              ;Loop for # cyls
2081 EB         04650          EX      DE,HL              ;# free grans -> HL
2082 C1         04660          POP     BC                 ;Pop user buf ptr
2083 F1         04670          POP     AF                 ;Rcvr # of sectors/gran
2084 FDE1       04680          POP     IY
2086 CDC906     04690 DOMUL16  CALL    @MUL16             ;Calc # of free sectors
2089 60         04700          LD      H,B                ;Cvrt # of free sectors
208A 55         04710          LD      D,L
208B 69         04720          LD      L,C                ;To free space in K by
208C 5F         04730          LD      E,A
208D 13         04740          INC     DE                 ;  dividing the # by 4
208E 13         04750          INC     DE                 ;Round up adjustment
208F CB3A       04760          SRL     D                  ;Divide 16-bit reg by 2
2091 CB1B       04770          RR      E
2093 CB3A       04780          SRL     D                  ;  & divide again
2095 CB1B       04790          RR      E
2097 73         04800          LD      (HL),E             ;Stuff the value
2098 23         04810          INC     HL
2099 72         04820          LD      (HL),D
209A C9         04830          RET
                04840 ;
                04850 ;        Read the hash index table
                04860 ;
209B 210023     04870 HITRD1   LD      HL,DIRBUF$         ;Pt to buffer
209E C5         04880          PUSH    BC
209F D5         04890          PUSH    DE
20A0 CDF718     04900          CALL    @DIRCYL            ;Dir cyl to reg D
20A3 1E01       04910          LD      E,1                ;Sector one
20A5 CDD818     04920          CALL    @RDSSC
20A8 D1         04930          POP     DE
20A9 C1         04940          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          05560        INC    DE
20E7 10F8        05570        DJNZ   GTM3
20E9 EB          05580        EX     DE,HL           ;Name+1 to DE
                 05590 ;
                 05600 ;      Found a match - exit with info
                 05610 ;
20EA E1          05620        POP    HL              ;Keep DE to name end+1
20EB E1          05630        POP    HL              ;Module start address
20EC C1          05640        POP    BC              ;Reg restoral
20ED AF          05650        XOR    A               ;Set Z-flag for return
20EE C9          05660        RET
                 05670 ;
                 05680 ;      No match - loop to next module
                 05690 ;
20EF D1          05700 GTM4   POP    DE
20F0 E1          05710 GTM5   POP    HL
20F1 23          05720        INC    HL              ;Point to last byte used
20F2 23          05730        INC    HL
20F3 7E          05740        LD     A,(HL)          ;P/u low-order
20F4 23          05750        INC    HL
20F5 66          05760        LD     H,(HL)          ;P/u high-order
20F6 6F          05770        LD     L,A
20F7 23          05780 GTM5A  INC    HL              ;Bump to next address
20F8 7C          05790        LD     A,H             ;Ck for wrap to zero
20F9 B5          05800        OR     L
20FA 20C0        05810        JR     NZ,GTM2         ;Loop if not through
20FC C1          05820 GTM6   POP    BC              ;Restore reg
20FD 3E08        05830        LD     A,8             ;Set "device not avail..
20FF B7          05840        OR     A
2100 C9          05850        RET
                 05860 ;
                 05870 ;      Found non-JR - Advance to high memory?
                 05880 ;
2101 7C          05890 GTM7   LD     A,H             ;Past driver core?
2102 FE13        05900        CP     @BYTEIO<-8
2104 30F6        05910        JR     NC,GTM6         ;Exit with "not found"
2106 2A0E04      05920 GTM8   LD     HL,(HIGH$)      ;  else p/u himem pointer
2109 18EC        05930        JR     GTM5A           ;  & hop to it if in use
                 05940 ;
                 05950 ;      Check a drive for availability
                 05960 ;
210B FDE5        05970 CKDRV  PUSH   IY              ;We use IY in disk I/O
210D CD1E1A      05980        CALL   @GTDCT          ;Get driver routine addr
2110 FD7E00      05990        LD     A,(IY+0)        ;P/u drive vector
2113 FEC3        06000        CP     0C3H            ;Ck for enabled
2115 C29D21      06010        JP     NZ,CKDR5        ;Bypass if disabled
2118 E5          06020        PUSH   HL
2119 D5          06030        PUSH   DE
211A FD7E06      06040        LD     A,(IY+6)        ;Make sure that the current
211D FDBE05      06050        CP     (IY+5)          ;  cylinder count is in range
2120 D22921      06060        JP     NC,CKDRV1       ;Go if in range
2123 CDC819      06070        CALL   @RSTOR          ;Restore drive
2126 C2AC21      06080        JP     NZ,CKDR7A       ;Go if error
                 06090 ;
2129 FD5605      06100 CKDRV1 LD     D,(IY+5)        ;P/u current track
212C 1E00        06110        LD     E,0             ;Set for sector 0
212E CDD019      06120        CALL   @SEEK           ;Set track info to FDC
2131 2079        06130        JR     NZ,CKDR7A       ;Go if error
2133 CDD419      06140        CALL   @RSLCT          ;Wait until not busy
2136 2074        06150        JR     NZ,CKDR7A       ;Not there - ret NZ
2138 FDCB035E    06160        BIT    3,(IY+3)        ;If hard drive, bypass
```

```
213C 2055      06170          JR      NZ,CKDR3A         ;  GAT data update
213E FDCB0466  06180          BIT     4,(IY+4)          ;If "ALIEN" by pass
2142 202C      06190          JR      NZ,CKDR2B         ;  test of index pulses
               06200          IF      @MOD4
2144 3A0E00    06210          LD      A,(FDDINT$)       ;Check 'SMOOTH' state
2147 B7        06220          OR      A
2148 3E09      06230          LD      A,09              ;Set MSB of count down
214A 2803      06240          JR      Z,INTRON          ;Go if not SMOOTH
214C CB3F      06250          SRL     A                 ;Divide the count by two
214E F3        06260          DI
               06270          ENDIF
               06280          IF      @MOD2
               06290          LD      A,20
               06300          ENDIF
214F 326121    06310 INTRON   LD      (CDCNT+1),A       ;Store in 'LD H' instruction
2152 212000    06320          LD      HL,0020H          ;Set up count (short)
               06330 ;
               06340 ;        Test for diskette in drive & rotating
               06350 ;
2155 CDA021    06360 CKDR1    CALL    INDEX             ;Test index pulse
2158 20FB      06370          JR      NZ,CKDR1          ;Jump on index
215A FDCB047E  06380          BIT     7,(IY+4)          ;Check CKDRV inhibit bit
215E 2010      06390          JR      NZ,CKDR2B         ;  if on skip index test
2160 2600      06400 CDCNT    LD      H,00H             ;CKDRV counter (long)
               06410          ;Count set from above
2162 CDA021    06420 CKDR2    CALL    INDEX             ;Test index pulse
2165 28FB      06430          JR      Z,CKDR2           ;Jump on no index
               06440          IF      @MOD4
2167 FB        06450          EI                        ;OK for INTs now
               06460          ENDIF
2168 212000    06470          LD      HL,0020H          ;Index off wait (short)
216B CDA021    06480 CKDR2A   CALL    INDEX
216E 20FB      06490          JR      NZ,CKDR2A         ;Jump on index
               06500 ;
               06510 ;        Diskette is rotating
               06520 ;
2170 F5        06530 CKDR2B   PUSH    AF                ;Save FDC status
2171 CDF718    06540          CALL    @DIRCYL           ;Get directory track in D
2174 21001D    06550          LD      HL,SBUFF$         ;Point to HIT buffer
2177 5D        06560          LD      E,L               ;Sector 0 for GAT
2178 CDD818    06570          CALL    @RDSSC            ;Read the GAT
217B 202E      06580          JR      NZ,CKDR7          ;Jump on error
217D 2ACC1D    06590          LD      HL,(SBUFF$+0CCH)        ;P/u excess tracks
2180 3E22      06600          LD      A,22H             ;Add offset
2182 85        06610          ADD     A,L
2183 FD7706    06620          LD      (IY+6),A          ;Max track # to DCT
2186 FDCB04AE  06630          RES     5,(IY+4)          ;Set to side 0
218A CB6C      06640          BIT     5,H               ;Test double sided
218C 2804      06650          JR      Z,CKDR3           ;Jump if only single
218E FDCB04EE  06660          SET     5,(IY+4)          ;Set for side 2
2192 F1        06670 CKDR3    POP     AF                ;Recover FDC status
2193 07        06680 CKDR3A   RLCA                      ;Shift write prot to 7
2194 FDB603    06690          OR      (IY+3)            ;Merge Soft WP bit
2197 E680      06700          AND     80H               ;Strip all but 7
2199 87        06710          ADD     A,A               ;Write prot to carry flg
               06720 ;
219A           06730 CKDR4    EQU     $
219A FB        06740          EI
219B D1        06750          POP     DE
219C E1        06760          POP     HL
219D FDE1      06770 CKDR5    POP     IY
```

```
219F C9         06780           RET
                06790 ;
21A0 7C         06800 INDEX     LD      A,H             ;Check countdown timer
21A1 B5         06810           OR      L
21A2 2807       06820           JR      Z,CKDR7         ;Err exit if 0
21A4 2B         06830           DEC     HL
21A5 CDD419     06840           CALL    @RSLCT          ;Check for index pulse
21A8 CB4F       06850           BIT     1,A             ;Test index
21AA C9         06860           RET
                06870 ;
21AB F1         06880 CKDR7     POP     AF
21AC 3E08       06890 CKDR7A    LD      A,8             ;Set Device not avail
21AE B7         06900           OR      A               ;Set NZ ret
21AF 18E9       06910           JR      CKDR4           ;Exit
                06920 ;
0012            06930 LILBUF    DS      18
21C3            06940 LAST      EQU     $
                06950           IFGT    $,DIRBUF$
                06960           ERR     'Module too big'
                06970           ENDIF
23FE            06980           ORG     MAXCOR$-2
23FE C303       06990           DW      LAST-SYS12      ;Overlay size
                07000 ;
1E00            07010           END     SYS12
```

```
$A1            03B7 $A2            03B8 $A3            03B9
$CKEOF         1470 @$SYS          08F0 @@1            0000
@@2            0000 @@3            0000 @@4            0000
@ABORT         1B08 @ADTSK         1CDA @BANK          0877
@BKSP          1486 @BREAK         196F @BYTEIO        1300
@CHNIO         0689 @CKBRKC        0553 @CKDRV         1993
@CKEOF         158F @CKTSK         1CF5 @CLOSE         1999
@CLS           0545 @CMNDI         197E @CMNDR         197B
@CTL           0623 @DATE          07A8 @DBGHK         199F
@DCINIT        19C0 @DCRES         19C4 @DCSTAT        19B5
@DCTBYT        1A2B @DEBUG         19A0 @DECHEX        03E1
@DIRCYL        18F7 @DIRRD         18BB @DIRWR         1803
@DIV16         06E3 @DIV8          1927 @DODIR         19AF
@DOKEY         19A9 @DSP           0642 @DSPLY         052D
@ERROR         1B0F @EXIT          1B0B @FEXT          1984
@FLAGS         196A @FNAME         199C @FRENCH        0000
@FSPEC         1981 @GATRD         1874 @GATWR         1875
@GERMAN        0000 @GET           0638 @GTDCB         1990
@GTDCT         1A1E @GTMOD         19B2 @HDFMT         19E4
@HEX16         07BD @HEX8          07C2 @HEXDEC        06F6
@HIGH$         1948 @HITRD         1897 @HITWR         1898
@HZ50          0000 @ICNFG         0086 @INIT          198D
@INTL          0000 @IPL           1BF2 @JCL           0630
@KBD           0635 @KEY           0628 @KEYIN         0585
@KITSK         0089 @KLTSK         1CD0 @LOAD          1B38
@LOC           14B3 @LOF           14DE @LOGER         0503
@LOGOT         0500 @MOD2          0000 @MOD4          FFFF
@MSG           0530 @MUL16         06C9 @MUL8          190A
@NMI           0066 @OPEN          198A @OPREG         0084
@PARAM         1987 @PAUSE         0382 @PEOF          14A2
@POSN          1434 @PRINT         0528 @PRT           063D
@PUT           0645 @RAMDIR        19AC @RDHDR         19D8
@RDSEC         19F4 @RDSSC         18D8 @RDTRK         19E0
@READ          1513 @REMOVE        19A6 @RENAME        1996
@REW           149B @RMTSK         1CD7 @RPTSK         1CEB
@RREAD         1473 @RSLCT         19D4 @RST00         0000
@RST08         0008 @RST10         0010 @RST18         0018
@RST20         0020 @RST28         0028 @RST30         0030
@RST38         0038 @RSTNMI        0FE9 @RSTOR         19C8
@RSTREG        0680 @RUN           1B1D @RWRIT         13AD
@SEEK          19D0 @SEEKSC        1421 @SKIP          1430
@SLCT          19BC @SOUND         0392 @STEPI         19CC
@TIME          078D @USA           FFFF @VDCTL         0B99
@VDCTL3        0D38 @VER           1560 @VRSEC         19DC
@WEOF          14EC @WHERE         1979 @WRITE         1531
@WRSEC         19E8 @WRSSC         19EC @WRTRK         19F0
@_VDCTL        0D42 ADDR_2_ROWCOL  0DF1 AFLAG$         006A
AUTO?          1FF1 BAR$           0201 BOOTST$        439D
BREAK?         1C60 BRKVEC$        1C88 BUR$           0200
BYTEND?        207A CASHK$         0A7B CDCNT          2160
CFCB$          00E0 CFGFCB$        00E0 CFLAG$         006C
CKDR1          2155 CKDR2          2162 CKDR2A         216B
CKDR2B         2170 CKDR3          2192 CKDR3A         2193
CKDR4          219A CKDR5          219D CKDR7          21AB
CKDR7A         21AC CKDRV          210B CKDRV1         2129
CKMOD@         1A7F CKOPEN@        1568 CKPAGE         1FAF
CLSBUF         1F48 CONFIG$        203F CORE$          0300
CR             000D CRTBGN$        F800 CYL_GRN        16AE
D@FBYT8        1A26 DATE$          0033 DAYTBL$        04C7
DBGSV$         00A0 DCBKL$         0031 DCT$           0470
```

```
DCTBYT8@        1A29 DCTFLD@         1A34 DFLAG$          006D
DIRBUF$         2300 DIRINF1         1E70 DIRINFO         1E41
DIS_DO_RAM      0846 DOALL           1EB0 DOALL1          1EB7
DOALL2          1EBB DOALL3          1EC8 DOALL4          1ED3
DOALL5          1EE0 DODATA$         0B94 DODCB$          0210
DOMUL16         2086 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
DSPLYIT         1F8E DSTDRV          2015 DTPMT$          04C2
DVREND$         0FF4 DVRHI$          0206 EFLAG$          006E
ENADIS_DO_RAM   0817 EXTDBG$         19A4 FDDINT$         000E
FEMSK$          006F FLGTAB$         006A GETSTUF         1E7E
GET_@_ROWCOL    0DAE GTM1            20B1 GTM2            20BC
GTM2A           20CE GTM3            20E1 GTM4            20EF
GTM5            20F0 GTM5A           20F7 GTM6            20FC
GTM7            2101 GTM8            2106 GTMOD           20AD
HERTZ$          0750 HIGH$           040E HITRD1          209B
HKRES$          1A6C IFLAG$          0072 INBUF$          0420
INDEX           21A0 INTIM$          003C INTMSK$         003D
INTRON          214F INTVC$          003E JCLCB$          0203
JDCB$           0024 JFCB$           00C0 JLDCB$          0230
JRET$           0026 KCK@            07D6 KEEP7           2075
KFLAG$          0074 KIDATA$         08FC KIDCB$          0208
LAST            21C3 LBANK$          0202 LDRV$           0023
LFLAG$          0075 LILBUF          21B1 LNKFCB@         1566
LOW$            001E LSVC$           000D MAXCOR$         2400
MAXDAY$         0401 MDIR            1EF0 MDIR1           1F2B
MDIR11          1F9E MDIR12          1FC1 MDIR2           1F2F
MDIR3           1F4D MDIR4           1F58 MDIR5           1F64
MINCOR$         3000 MODOUT$         0076 MONTBL$         04DC
NFLAG$          0077 ONESID          1E63 OPREG$          0078
OPREG_SV_AREA   086E OPREG_SV_PTR    0835 ORARET@         14DC
OSRLS$          003B OSVER$          0085 OVRLY$          0069
PAKNAM$         0410 PAUSE@          0382 PCSAVE$         07AF
PDRV$           001B PHIGH$          001C PRDCB$          0218
PUGAT           2074 PUTA@DE         0DCD PUT_@           0DCA
PUT_@_ROWCOL    0DC6 RAMDIR          1E10 RFLAG$          007B
ROWCOL_2_ADDR   0DD0 RST38@          1BFF RSTOR$          04C4
RWRIT@          13A2 S1DCB$          0238 SBUFF$          1D00
SET@EXEC        1A79 SET_SCROLL      0CF3 SFCB$           008C
SFLAG$          007C SIDCB$          0220 SODCB$          0228
SPACE           2036 SPACE0          2024 SPACE4$         2142
STACK$          0380 START$          0000 STUFB1          1FE9
STUFB2          1FF6 STUFB3          1FFA STUFB4          2006
STUFB5          2011 STUFB6          2019 STUFBUF         1FE1
SVCRET$         000B SVCTAB$         0100 SYS12           1E00
SYSERR$         1B13 TCB$            004E TFLAG$          007D
TIME$           002D TIMER$          002C TIMSL$          002B
TIMTSK$         0713 TMPMT$          04C3 TRACE_INT       07B1
TSTOPT          1F6A TSTS1           1FCF TSTS2           1FDC
TSTSAM          1FC3 TYPHK$          0A8F TYPTSK$         0B26
USTOR$          0013 VFLAG$          007F WRINT$          0080
ZERO$           0401 ZEROA@          13A0
00000 Total errors
```

NOTES:

SYS13/SYS

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
                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             0000 @@2             0000 @@3             0000
@@4             0000 CR              000D LF              000A
NOCMD           1E38 NOCMD$          1E74 NOSYS13         1E29
NXCI$           1E40 START           1E22 SYS13           1E00

1E00 is the transfer address
00000 Total errors
```

NOTES:

NOTES:

NOTES:

NOTES:

NOTES:

NOTES:

```
                00100 ;LDOS60/EQU - Equates from cross reference of Lowcore
0000            00110          TITLE   <LDOS60/EQU>
                00120 ;
08F0            00130 @$SYS    EQU     08F0H
0000            00140 @@1      DEFL    0000H
0000            00150 @@2      DEFL    0000H
0000            00160 @@3      DEFL    0000H
0000            00170 @@4      DEFL    0000H
0877            00180 @BANK    EQU     0877H
1300            00190 @BYTEIO  EQU     1300H
0689            00200 @CHNIO   EQU     0689H
0553            00210 @CKBRKC  EQU     0553H
0545            00220 @CLS     EQU     0545H
0623            00230 @CTL     EQU     0623H
07A8            00240 @DATE    EQU     07A8H
06E3            00250 @DIV16   EQU     06E3H
0642            00260 @DSP     EQU     0642H
052D            00270 @DSPLY   EQU     052DH
0000            00280 @FRENCH  EQU     0000H
0000            00290 @GERMAN  EQU     0000H
0638            00300 @GET     EQU     0638H
07BD            00310 @HEX16   EQU     07BDH
07C2            00320 @HEX8    EQU     07C2H
06F6            00330 @HEXDEC  EQU     06F6H
0000            00340 @HZ50    EQU     0000H
0000            00350 @INTL    EQU     0000H
0630            00360 @JCL     EQU     0630H
0635            00370 @KBD     EQU     0635H
0628            00380 @KEY     EQU     0628H
0585            00390 @KEYIN   EQU     0585H
0089            00400 @KITSK   EQU     0089H
0503            00410 @LOGER   EQU     0503H
0500            00420 @LOGOT   EQU     0500H
0000            00430 @MOD2    EQU     0000H
FFFF            00440 @MOD4    EQU     0FFFFH
0530            00450 @MSG     EQU     0530H
06C9            00460 @MUL16   EQU     06C9H
0084            00470 @OPREG   EQU     0084H
0528            00480 @PRINT   EQU     0528H
063D            00490 @PRT     EQU     063DH
0645            00500 @PUT     EQU     0645H
0FE9            00510 @RSTNMI  EQU     0FE9H
0680            00520 @RSTREG  EQU     0680H
078D            00530 @TIME    EQU     078DH
FFFF            00540 @USA     EQU     0FFFFH
0B99            00550 @VDCTL   EQU     0B99H
0D38            00560 @VDCTL3  EQU     0D38H
0D42            00570 @_VDCTL  EQU     0D42H
0DF1            00580 ADDR_2_ROWCOL    EQU     0DF1H
0201            00590 BAR$     EQU     0201H
439D            00600 BOOTST$  EQU     439DH
0200            00610 BUR$     EQU     0200H
0A7B            00620 CASHK$   EQU     0A7BH
006C            00630 CFLAG$   EQU     006CH
0300            00640 CORE$    DEFL    0300H
F800            00650 CRTBGN$  EQU     0F800H
0033            00660 DATE$    EQU     0033H
04C7            00670 DAYTBL$  EQU     04C7H
0031            00680 DCBKL$   EQU     0031H
0470            00690 DCT$     EQU     0470H
006D            00700 DFLAG$   EQU     006DH
```

```
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
```

```
0713         01320 TIMTSK$ EQU     0713H
04C3         01330 TMPMT$  EQU     04C3H
07B1         01340 TRACE_INT       EQU     07B1H
0A8F         01350 TYPHK$  EQU     0A8FH
0B26         01360 TYPTSK$ EQU     0B26H
007F         01370 VFLAG$  EQU     007FH
0401         01380 ZERO$   EQU     0401H
             01390 ;
No end statement
00000 Total errors
```

```
                00100 ;SYS0/EQU - Equates from cross reference of Sysres
0000            00110         TITLE   <SYS0/EQU>
                00120 ;
03B7            00130 $A1     EQU     03B7H
03B8            00140 $A2     EQU     03B8H
03B9            00150 $A3     EQU     03B9H
1470            00160 $CKEOF  EQU     1470H
08F0            00170 @$SYS   EQU     08F0H
0000            00180 @@1     DEFL    0000H
0000            00190 @@1     DEFL    0000H
0000            00200 @@2     DEFL    0000H
0000            00210 @@2     DEFL    0000H
0000            00220 @@3     DEFL    0000H
0000            00230 @@3     DEFL    0000H
0000            00240 @@4     DEFL    0000H
0000            00250 @@4     DEFL    0000H
1B08            00260 @ABORT  EQU     1B08H
1CDA            00270 @ADTSK  EQU     1CDAH
0877            00280 @BANK   EQU     0877H
1486            00290 @BKSP   EQU     1486H
196F            00300 @BREAK  EQU     196FH
1300            00310 @BYTEIO EQU     1300H
0689            00320 @CHNIO  EQU     0689H
0553            00330 @CKBRKC EQU     0553H
1993            00340 @CKDRV  EQU     1993H
158F            00350 @CKEOF  EQU     158FH
1CF5            00360 @CKTSK  EQU     1CF5H
1999            00370 @CLOSE  EQU     1999H
0545            00380 @CLS    EQU     0545H
197E            00390 @CMNDI  EQU     197EH
197B            00400 @CMNDR  EQU     197BH
0623            00410 @CTL    EQU     0623H
07A8            00420 @DATE   EQU     07A8H
199F            00430 @DBGHK  EQU     199FH
19C0            00440 @DCINIT EQU     19C0H
19C4            00450 @DCRES  EQU     19C4H
19B5            00460 @DCSTAT EQU     19B5H
1A2B            00470 @DCTBYT EQU     1A2BH
19A0            00480 @DEBUG  EQU     19A0H
03E1            00490 @DECHEX EQU     03E1H
18F7            00500 @DIRCYL EQU     18F7H
18BB            00510 @DIRRD  EQU     18BBH
1803            00520 @DIRWR  EQU     1803H
06E3            00530 @DIV16  EQU     06E3H
1927            00540 @DIV8   EQU     1927H
19AF            00550 @DODIR  EQU     19AFH
19A9            00560 @DOKEY  EQU     19A9H
0642            00570 @DSP    EQU     0642H
052D            00580 @DSPLY  EQU     052DH
1B0F            00590 @ERROR  EQU     1B0FH
1B0B            00600 @EXIT   EQU     1B0BH
1984            00610 @FEXT   EQU     1984H
196A            00620 @FLAGS  EQU     196AH
199C            00630 @FNAME  EQU     199CH
0000            00640 @FRENCH EQU     0000H
1981            00650 @FSPEC  EQU     1981H
1874            00660 @GATRD  EQU     1874H
1875            00670 @GATWR  EQU     1875H
0000            00680 @GERMAN EQU     0000H
0638            00690 @GET    EQU     0638H
1990            00700 @GTDCB  EQU     1990H
```

```
1A1E          00710 @GTDCT  EQU     1A1EH
19B2          00720 @GTMOD  EQU     19B2H
19E4          00730 @HDFMT  EQU     19E4H
07BD          00740 @HEX16  EQU     07BDH
07C2          00750 @HEX8   EQU     07C2H
06F6          00760 @HEXDEC EQU     06F6H
1948          00770 @HIGH$  EQU     1948H
1897          00780 @HITRD  EQU     1897H
1898          00790 @HITWR  EQU     1898H
0000          00800 @HZ50   EQU     0000H
0086          00810 @ICNFG  EQU     0086H
198D          00820 @INIT   EQU     198DH
0000          00830 @INTL   EQU     0000H
1BF2          00840 @IPL    EQU     1BF2H
0630          00850 @JCL    EQU     0630H
0635          00860 @KBD    EQU     0635H
0628          00870 @KEY    EQU     0628H
0585          00880 @KEYIN  EQU     0585H
0089          00890 @KITSK  EQU     0089H
0089          00900 @KITSK  EQU     0089H
1CD0          00910 @KLTSK  EQU     1CD0H
1B38          00920 @LOAD   EQU     1B38H
14B3          00930 @LOC    EQU     14B3H
14DE          00940 @LOF    EQU     14DEH
0503          00950 @LOGER  EQU     0503H
0500          00960 @LOGOT  EQU     0500H
0000          00970 @MOD2   EQU     0000H
FFFF          00980 @MOD4   EQU     0FFFFH
0530          00990 @MSG    EQU     0530H
06C9          01000 @MUL16  EQU     06C9H
190A          01010 @MUL8   EQU     190AH
0066          01020 @NMI    EQU     0066H
198A          01030 @OPEN   EQU     198AH
0084          01040 @OPREG  EQU     0084H
1987          01050 @PARAM  EQU     1987H
0382          01060 @PAUSE  EQU     0382H
14A2          01070 @PEOF   EQU     14A2H
1434          01080 @POSN   EQU     1434H
0528          01090 @PRINT  EQU     0528H
063D          01100 @PRT    EQU     063DH
0645          01110 @PUT    EQU     0645H
19AC          01120 @RAMDIR EQU     19ACH
19D8          01130 @RDHDR  EQU     19D8H
19F4          01140 @RDSEC  EQU     19F4H
18D8          01150 @RDSSC  EQU     18D8H
19E0          01160 @RDTRK  EQU     19E0H
1513          01170 @READ   EQU     1513H
19A6          01180 @REMOVE EQU     19A6H
1996          01190 @RENAME EQU     1996H
149B          01200 @REW    EQU     149BH
1CD7          01210 @RMTSK  EQU     1CD7H
1CEB          01220 @RPTSK  EQU     1CEBH
1473          01230 @RREAD  EQU     1473H
19D4          01240 @RSLCT  EQU     19D4H
0000          01250 @RST00  EQU     0000H
0008          01260 @RST08  EQU     0008H
0010          01270 @RST10  EQU     0010H
0018          01280 @RST18  EQU     0018H
0020          01290 @RST20  EQU     0020H
0028          01300 @RST28  EQU     0028H
0030          01310 @RST30  EQU     0030H
```

```
0038          01320 @RST38  EQU     0038H
0FE9          01330 @RSTNMI EQU     0FE9H
19C8          01340 @RSTOR  EQU     19C8H
0680          01350 @RSTREG EQU     0680H
1B1D          01360 @RUN    EQU     1B1DH
13AD          01370 @RWRIT  EQU     13ADH
19D0          01380 @SEEK   EQU     19D0H
1421          01390 @SEEKSC EQU     1421H
1430          01400 @SKIP   EQU     1430H
19BC          01410 @SLCT   EQU     19BCH
0392          01420 @SOUND  EQU     0392H
19CC          01430 @STEPI  EQU     19CCH
078D          01440 @TIME   EQU     078DH
FFFF          01450 @USA    EQU     0FFFFH
0B99          01460 @VDCTL  EQU     0B99H
0D38          01470 @VDCTL3 EQU     0D38H
1560          01480 @VER    EQU     1560H
19DC          01490 @VRSEC  EQU     19DCH
14EC          01500 @WEOF   EQU     14ECH
1979          01510 @WHERE  EQU     1979H
1531          01520 @WRITE  EQU     1531H
19E8          01530 @WRSEC  EQU     19E8H
19EC          01540 @WRSSC  EQU     19ECH
19F0          01550 @WRTRK  EQU     19F0H
0D42          01560 @_VDCTL EQU     0D42H
0DF1          01570 ADDR_2_ROWCOL   EQU     0DF1H
006A          01580 AFLAG$  EQU     006AH
1FF1          01590 AUTO?   EQU     1FF1H
0201          01600 BAR$    EQU     0201H
439D          01610 BOOTST$ EQU     439DH
1C60          01620 BREAK?  EQU     1C60H
1C88          01630 BRKVEC$ EQU     1C88H
0200          01640 BUR$    EQU     0200H
0A7B          01650 CASHK$  EQU     0A7BH
00E0          01660 CFCB$   EQU     00E0H
00E0          01670 CFGFCB$ EQU     00E0H
006C          01680 CFLAG$  EQU     006CH
006C          01690 CFLAG$  EQU     006CH
1A7F          01700 CKMOD@  EQU     1A7FH
1568          01710 CKOPEN@ EQU     1568H
203F          01720 CONFIG$ EQU     203FH
1CFF          01730 CORE$   DEFL    1CFFH
1BFF          01740 CORE$   DEFL    1BFFH
1948          01750 CORE$   DEFL    1948H
1948          01760 CORE$   DEFL    1948H
0300          01770 CORE$   DEFL    0300H
F800          01780 CRTBGN$ EQU     0F800H
16AE          01790 CYL_GRN EQU     16AEH
1A26          01800 D@FBYT8 EQU     1A26H
0033          01810 DATE$   EQU     0033H
0033          01820 DATE$   EQU     0033H
04C7          01830 DAYTBL$ EQU     04C7H
00A0          01840 DBGSV$  EQU     00A0H
0031          01850 DCBKL$  EQU     0031H
0470          01860 DCT$    EQU     0470H
1A29          01870 DCTBYT8@        EQU     1A29H
1A34          01880 DCTFLD@ EQU     1A34H
006D          01890 DFLAG$  EQU     006DH
006D          01900 DFLAG$  EQU     006DH
2300          01910 DIRBUF$ EQU     2300H
0846          01920 DIS_DO_RAM      EQU     0846H
```

```
0B94       01930 DODATA$ EQU     0B94H
0210       01940 DODCB$  EQU     0210H
0C44       01950 DO_CONTROL      EQU     0C44H
0CB8       01960 DO_DSPCHAR      EQU     0CB8H
0C8C       01970 DO_INVERT_DIS   EQU     0C8CH
0C89       01980 DO_INVERT_ENA   EQU     0C89H
0C9B       01990 DO_INVERT_OFF   EQU     0C9BH
0000       02000 DO_MASK EQU     0000H
0BCB       02010 DO_RET  EQU     0BCBH
0BCC       02020 DO_RET1 EQU     0BCCH
0CCE       02030 DO_SCROLL       EQU     0CCEH
0BEA       02040 DO_TABS EQU     0BEAH
04C0       02050 DSKTYP$ EQU     04C0H
04C2       02060 DTPMT$  EQU     04C2H
0FF4       02070 DVREND$ EQU     0FF4H
0206       02080 DVRHI$  EQU     0206H
006E       02090 EFLAG$  EQU     006EH
0817       02100 ENADIS_DO_RAM   EQU     0817H
19A4       02110 EXTDBG$ EQU     19A4H
000E       02120 FDDINT$ EQU     000EH
000E       02130 FDDINT$ EQU     000EH
006F       02140 FEMSK$  EQU     006FH
006A       02150 FLGTAB$ EQU     006AH
006A       02160 FLGTAB$ EQU     006AH
0DAE       02170 GET_@_ROWCOL    EQU     0DAEH
0750       02180 HERTZ$  EQU     0750H
040E       02190 HIGH$   EQU     040EH
1A6C       02200 HKRES$  EQU     1A6CH
0072       02210 IFLAG$  EQU     0072H
0072       02220 IFLAG$  EQU     0072H
0420       02230 INBUF$  EQU     0420H
003C       02240 INTIM$  EQU     003CH
003D       02250 INTMSK$ EQU     003DH
003E       02260 INTVC$  EQU     003EH
003E       02270 INTVC$  EQU     003EH
0203       02280 JCLCB$  EQU     0203H
0024       02290 JDCB$   EQU     0024H
00C0       02300 JFCB$   EQU     00C0H
0230       02310 JLDCB$  EQU     0230H
0026       02320 JRET$   EQU     0026H
07D6       02330 KCK@    EQU     07D6H
0074       02340 KFLAG$  EQU     0074H
0074       02350 KFLAG$  EQU     0074H
08FC       02360 KIDATA$ EQU     08FCH
0208       02370 KIDCB$  EQU     0208H
0202       02380 LBANK$  EQU     0202H
0023       02390 LDRV$   EQU     0023H
0075       02400 LFLAG$  EQU     0075H
1566       02410 LNKFCB@ EQU     1566H
001E       02420 LOW$    EQU     001EH
000D       02430 LSVC$   EQU     000DH
2400       02440 MAXCOR$ EQU     2400H
0401       02450 MAXDAY$ EQU     0401H
3000       02460 MINCOR$ EQU     3000H
0076       02470 MODOUT$ EQU     0076H
0076       02480 MODOUT$ EQU     0076H
04DC       02490 MONTBL$ EQU     04DCH
0077       02500 NFLAG$  EQU     0077H
0078       02510 OPREG$  EQU     0078H
0078       02520 OPREG$  EQU     0078H
086E       02530 OPREG_SV_AREA   EQU     086EH
```

```
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
0410          02590 PAKNAM$ EQU      0410H
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
0DD0          02710 ROWCOL_2_ADDR    EQU     0DD0H
1BFF          02720 RST38@  EQU      1BFFH
04C4          02730 RSTOR$  EQU      04C4H
13A2          02740 RWRIT@  EQU      13A2H
0238          02750 S1DCB$  EQU      0238H
1D00          02760 SBUFF$  EQU      1D00H
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
0380          02850 STACK$  EQU      0380H
0000          02860 START$  EQU      0000H
0000          02870 START$  EQU      0000H
000B          02880 SVCRET$ EQU      000BH
0100          02890 SVCTAB$ EQU      0100H
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
13A0          03090 ZEROA@  EQU      13A0H
No end statement
00000 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          01930 @SLCT    EQU      19BCH
0392          01940 @SOUND   EQU      0392H
19CC          01950 @STEPI   EQU      19CCH
078D          01960 @TIME    EQU      078DH
FFFF          01970 @USA     EQU      0FFFFH
0B99          01980 @VDCTL   EQU      0B99H
0D38          01990 @VDCTL3  EQU      0D38H
1560          02000 @VER     EQU      1560H
19DC          02010 @VRSEC   EQU      19DCH
14EC          02020 @WEOF    EQU      14ECH
1979          02030 @WHERE   EQU      1979H
1531          02040 @WRITE   EQU      1531H
19E8          02050 @WRSEC   EQU      19E8H
19EC          02060 @WRSSC   EQU      19ECH
19F0          02070 @WRTRK   EQU      19F0H
0D42          02080 @_VDCTL  EQU      0D42H
0DF1          02090 ADDR_2_ROWCOL     EQU      0DF1H
006A          02100 AFLAG$   EQU      006AH
1FF1          02110 AUTO?    EQU      1FF1H
0201          02120 BAR$     EQU      0201H
439D          02130 BOOTST$  EQU      439DH
1C60          02140 BREAK?   EQU      1C60H
1C88          02150 BRKVEC$  EQU      1C88H
0200          02160 BUR$     EQU      0200H
0A7B          02170 CASHK$   EQU      0A7BH
00E0          02180 CFCB$    EQU      00E0H
00E0          02190 CFGFCB$  EQU      00E0H
006C          02200 CFLAG$   EQU      006CH
006C          02210 CFLAG$   EQU      006CH
1A7F          02220 CKMOD@   EQU      1A7FH
1568          02230 CKOPEN@  EQU      1568H
1FD6          02240 CMD_AH   EQU      1FD6H
1E81          02250 CMD_C    EQU      1E81H
208B          02260 CMD_CI   EQU      208BH
1EAB          02270 CMD_D    EQU      1EABH
1EC9          02280 CMD_DEC  EQU      1EC9H
1F82          02290 CMD_G    EQU      1F82H
1EB1          02300 CMD_INC  EQU      1EB1H
1ECE          02310 CMD_O    EQU      1ECEH
203F          02320 CMD_R    EQU      203FH
1E9D          02330 CMD_S    EQU      1E9DH
1EA1          02340 CMD_U    EQU      1EA1H
1E9C          02350 CMD_X    EQU      1E9CH
203F          02360 CONFIG$  EQU      203FH
0300          02370 CORE$    DEFL     0300H
1CFF          02380 CORE$    DEFL     1CFFH
1BFF          02390 CORE$    DEFL     1BFFH
1948          02400 CORE$    DEFL     1948H
1948          02410 CORE$    DEFL     1948H
F800          02420 CRTBGN$  EQU      0F800H
221A          02430 CV2HEX@  EQU      221AH
16AE          02440 CYL_GRN  EQU      16AEH
1A26          02450 D@FBYT8  EQU      1A26H
0033          02460 DATE$    EQU      0033H
0033          02470 DATE$    EQU      0033H
04C7          02480 DAYTBL$  EQU      04C7H
00A0          02490 DBGSV$   EQU      00A0H
0031          02500 DCBKL$   EQU      0031H
0470          02510 DCT$     EQU      0470H
1A29          02520 DCTBYT8@          EQU      1A29H
1A34          02530 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          03760 USTOR$  EQU     0013H
007F          03770 VFLAG$  EQU     007FH
007F          03780 VFLAG$  EQU     007FH
2211          03790 WR1HEX@ EQU     2211H
2215          03800 WR2HEX@ EQU     2215H
0080          03810 WRINT$  EQU     0080H
2231          03820 WRSPA@  EQU     2231H
2157          03830 XY_TAB  EQU     2157H
0401          03840 ZERO$   EQU     0401H
13A0          03850 ZEROA@  EQU     13A0H
No end statement
00000 Total errors
```