

TRSDOS/LS - DOS 6.x

"THE SOURCE"

```

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

```

The HIT entries were transferred prior

```

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

```

"THE SOURCE"

CONTENTS

Introduction

APPEND - See COPY

ATTRIB	Page 1
AUTO	Page 19
BUILD	Page 27
CAT - See DIR	
CLS - See LIB	
COPY	Page 39
CREATE	Page 73
DATE	Page 83
DEBUG	Page 99
DEVICE	Page 109
DIR	Page 125
DO	Page 175
DUMP	Page 197
FILTER - See SET	
FORMS	Page 205
FREE	Page 227
LIB	Page 249
LINK	Page 255
LIST	Page 261
LOAD	Page 279
MEMORY	Page 287
PURGE	Page 301
REMOVE	Page 319
RENAME	Page 327
RESET	Page 337
ROUTE	Page 345
RUN - See LOAD	
SET	Page 355
SETCOM	Page 363
SETKI	Page 387
SPOOL	Page 397
SYSGEN	Page 427
SYSTEM	Page 443
TIME - See DATE	
TOF - See LIB	
VERIFY - See DEBUG	

Appendix



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

TRSDOS and TRS-80 are trademarks of Tandy Corporation
LDOS and LS-DOS are trademarks of Logical Systems, Inc.

Copyright (C) 1982, 1983, 1984 by Logical Systems, Inc.
All Rights Reserved

Introduction to Volume Two

This is volume two 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 Library commands found in SYS6/SYS, SYS7/SYS, and SYS8/SYS.

The individual commands will be presented in alphabetical order without regard to their particular /SYS module. The lead-in page for each command will give the /SYS file containing the command, the ISAM number assigned to the command, and the names of the different commands (if more than one) that the code contains. A symbol table listing will follow each assembly listing.

The SVC macro file used during assembly of the utilities will be listed in Appendix A, along with the equate files generated during assembly of the resident part of the operating system.

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 system Libraries. It can be used for reference purposes and to view examples of different program structures. 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.

TRSDOS and TRS-80 are trademarks of Tandy Corporation
LDOS and LS-DOS are trademarks of Logical Systems, Inc.

Copyright (C) 1982, 1983, 1984 by Logical Systems, Inc.
All Rights Reserved

Command: ATTRIB

Library: SYS7/SYS

ISAM # : 51H


```

00100 ;LBATTRIB/ASM - ATTRIB Command
00000 00110 TITLE <ATTRIB - LS-DOS 6.2>
00120 ;
0000D 00130 CR EQU 13
4296 00140 BLNKMPW EQU 4296H
42E0 00150 PASSWORD EQU 42E0H
00160 ;
00000 00170 *GET SVCMAC:3 ;SVC Macro equivalents
00010 ;SVCMAC/ASM - LS-DOS Version VI
00020 *LIST OFF
00300 *LIST ON
00180 ;
2400 00190 ORG 2400H
00200 ;
00210 ATTRIB
2400 ED733124 00220 LD (SAVE$P+1),SP ;Save stack pointer
2404 CD3724 00230 CALL ATTRIB1 ;Call attrib code
2407 210000 00240 LD HL,0 ;Set no error
240A 1824 00250 JR SAVE$P ;P/u stack & return
00260 ;
00270 ; I/O Error Handling
00280 ;
240C FE3F 00290 IOERR CP 63 ;Extended error?
240E 281A 00300 JR Z,EXTERR
2410 6F 00310 LD L,A ;Error # to HL
2411 2600 00320 LD H,0
2413 F6C0 00330 OR 0C0H ;Abbrev & return
2415 4F 00340 LD C,A
2416 00350 @@ERROR
2416 3E1A 00001 LD A,26
2418 EF 00002 RST 40
2419 1815 00360 JR SAVE$P ;P/u Stack & RETurn
00370 ;
00380 ; Internal Error Message Handling
00390 ;
241B 21FA28 00400 ABORT LD HL,ABORT$ ;"Command aborted
241E DD 00410 DB 0DDH
241F 210A28 00420 SPCREQ LD HL,SPCREQ$ ;"File spec required
2422 DD 00430 DB 0DDH
2423 211D28 00440 NOTDUN LD HL,NOTDUN$ ;"Specifications req.
2426 DD 00450 DB 0DDH
2427 213528 00460 ATBERR LD HL,ATBERR$ ;"Attr. specification error
242A 00470 EXTERR @@LOGOT
00003 IFEQ 00H,1
00004 LD HL,
00005 ENDIF
242A 3E0C 00006 LD A,12
242C EF 00007 RST 40
242D 21FFFF 00480 LD HL,-1 ;Set abort error
2430 310000 00490 SAVE$P LD SP,$-$ ;Reload stack pointer
2433 00500 @@CKBRKC ;Clean up <BREAK>
2433 3E6A 00008 LD A,106
2435 EF 00009 RST 40
2436 C9 00510 RET
00520 ;
00530 ; ATTRIB1 - Set Attributes of a file/disk
00540 ;
00550 ATTRIB1
2437 113F29 00560 LD DE,FCB ;Check filespec or drive #
243A 00570 @@FSPEC
243A 3E4E 00010 LD A,78

```

```

243C EF      00011      RST      40
243D C2DB25  00580      JP        NZ,PROT      ;Assume drive # if error
2440 1A      00590      LD        A,(DE)      ;Cannot be a device
2441 FE2A    00600      CP        '*'
2443 CA1F24  00610      JP        Z,SPCREQ
2446        00620      @@FLAGS      ;Get flag table pointer
2446 3E65    00012      LD        A,101
2448 EF      00013      RST      40
2449 E5      00630      PUSH     HL      ;Save cmdline ptr
244A 21002A  00640      LD        HL,BUFFER ;Use local buffer
244D 45      00650      LD        B,L      ;Open the file
244E FDCB12C6 00660      SET      0,(IY+'S'-'A') ;Don't set file open bit
2452        00670      @@OPEN
2452 3E3B    00014      LD        A,59
2454 EF      00015      RST      40
2455 E1      00680      POP      HL      ;Rcvr cmdline ptr
2456 C20C24  00690      JP        NZ,IOERR ;Jump if non-existent
2459 3A4029  00700      LD        A,(FCB+1) ;P/u protection
245C E607    00710      AND      7      ;Mask other bits
245E 3E25    00720      LD        A,25H   ;Init for access denied
2460 C20C24  00730      JP        NZ,IOERR ;Jump if no can do
2463 AF      00740      XOR      A
2464 32A125  00750      LD        (PRMCO+1),A ;Init prot to 0
                00760 ;
                00770 ;      Convert command line to upper case
                00780 ;
2467 E5      00790      PUSH     HL      ;Save cmdline ptr
2468 7E      00800 ATT0   LD        A,(HL)      ; & cvrt lc to UC
2469 FE0D    00810      CP        CR
246B 2811    00820      JR        Z,ATT02   ;Done if CR
246D FE03    00830      CP        3      ; ETX
246F 280D    00840      JR        Z,ATT02   ;
2471 FE61    00850      CP        'a'      ;Not lc?
2473 3806    00860      JR        C,ATT01
2475 FE7B    00870      CP        'z'+1
2477 3002    00880      JR        NC,ATT01
2479 CBAE    00890      RES      5,(HL)      ;<a-z> to <A-Z>
247B 23      00900 ATT01  INC      HL      ;Bump to next char
247C 18EA    00910      JR        ATT0      ;Loop
                00920 ;
247E E1      00930 ATT02  POP      HL      ;Rcvr orig cmdline ptr
                00940 ;
                00950 ;      Scan command line for parameters
                00960 ;
247F 7E      00970 ATT1   LD        A,(HL)      ;Scan for start of parm
2480 FE28    00980      CP        '('      ;There yet?
2482 2807    00990      JR        Z,ATT2     ;Jump if so
2484 FE20    01000      CP        ' '      ;Ignore spaces
2486 2005    01010      JR        NZ,ATT3    ;Assume parm on dif char
2488 23      01020      INC      HL
2489 18F4    01030      JR        ATT1
248B 23      01040 ATT2   INC      HL      ;Bump past '('
248C 7E      01050      LD        A,(HL)
248D FE49    01060 ATT3   CP        'I'      ;Ck for INV
248F 2870    01070      JR        Z,DOINV
2491 FE56    01080      CP        'V'      ;Ck for VIS
2493 287E    01090      JR        Z,DOVIS
2495 FE55    01100      CP        'U'      ;Ck for USER
2497 CA2525  01110      JP        Z,DOUSE
249A FE4F    01120      CP        'O'      ;Ck for OWNER
249C CA3A25  01130      JP        Z,DOOWN

```

```

249F FE50      01140      CP      'P'      ;Ck for PROT
24A1 C22724    01150      JP      NZ,ATBERR ;Err if none of the above
                01160 ;
                01170 ;      Process PROT=parm
                01180 ;
24A4 CD6925    01190      CALL     PRSFLD      ;Parse field
24A7 CA2724    01200      JP      Z,ATBERR      ;Error if end of line
24AA E5        01210      PUSH     HL          ;Save ptr to next char
24AB 0608      01220      LD      B,8          ;Init for 8 prots
24AD ED5B3329  01230      LD      DE,(PSWDBUF) ;P/u 1st 2 chars
24B1 215328    01240      LD      HL,PROTS$    ;Pt to various prots
24B4 7E        01250 DOPR01 LD      A,(HL)    ;P/u 1st prot char
24B5 23        01260      INC      HL          ;Bump pointer
24B6 BB        01270      CP      E            ;Does 1st match?
24B7 CCC124    01280      CALL     Z,DOPR02      ;Check 2nd if 1st OK
24BA 23        01290      INC      HL          ;Bump to next
24BB 10F7      01300      DJNZ     DOPR01      ;Loop for all 8
24BD E1        01310      POP      HL          ;Stack integrity
24BE C32724    01320      JP      ATBERR      ;Abort if no match
                01330 ;
                01340 ;      Check 2nd prot= char for match
                01350 ;
24C1 7E        01360 DOPR02 LD      A,(HL)    ;P/u 2nd table char
24C2 BA        01370      CP      D            ;Match user's entry?
24C3 C0        01380      RET      NZ          ;Go back if not
24C4 F1        01390      POP      AF          ;Pop the ret addr
24C5 78        01400      LD      A,B          ;Calculate which prot was
24C6 3D        01410      DEC      A            ; entered by the user
24C7 2814      01420      JR      Z,DOPR03      ;Jump on PROT=FU
24C9 FE05      01430      CP      5            ;REname, REad, REMove?
24CB 2010      01440      JR      NZ,DOPR03      ;Go if none of the above
24CD 3A3529    01450      LD      A,(PSWDBUF+2) ;P/u user's 3rd char
24D0 FE4E      01460      CP      'N'          ;Was it 'N'?
24D2 3E02      01470      LD      A,2          ;Init for REname
24D4 2807      01480      JR      Z,DOPR03      ;Go if REName
24D6 3D        01490      DEC      A            ;Else init to REMove
24D7 FE4D      01500      CP      'M'          ; & test entry
24D9 2802      01510      JR      Z,DOPR03      ;
24DB 3E05      01520      LD      A,5            ; else assume READ
24DD 32A925    01530 DOPR03 LD      (PROTLVL+1),A ;Stuff protection level
24E0 E1        01540      POP      HL          ;Rcvr INBUF$ pointer
24E1 0601      01550      LD      B,1          ;Init to show PROT given
24E3 7E        01560 DOPR04 LD      A,(HL)    ;P/u next parm
24E4 FE22      01570      CP      '"'          ;Closing quote on last?
24E6 2001      01580      JR      NZ,DOPR05      ;Go if something else
24E8 23        01590      INC      HL          ;Ignore closing quote
24E9 3AA125    01600 DOPR05 LD      A,(PRMCO+1) ;P/u parm test bits
24EC B0        01610      OR      B            ;Merge PROT entered
24ED 32A125    01620      LD      (PRMCO+1),A ;Restuff parm test bits
24F0 7E        01630      LD      A,(HL)    ;P/u next char
24F1 FE0D      01640      CP      CR          ;End of line?
24F3 2802      01650      JR      Z,$+4        ;Go on end-of-line
24F5 FE29      01660      CP      ')'          ;End of parms?
24F7 CA9525    01670      JP      Z,UPDDIR      ;Go on end of parms
24FA FE2C      01680      CP      ','          ;More parms?
24FC 288D      01690      JR      Z,ATT2        ;Loop on more parms
24FE C32724    01700      JP      ATBERR      ;Exit on wrong char
                01710 ;
                01720 ;      Process INV parm
                01730 ;
2501 CD6925    01740 DOINV CALL     PRSFLD      ;Parse parm

```

```

2504 C22724 01750 JP NZ,ATBERR ;Go on parm error
2507 3AB125 01760 LD A,(IVCOD+1) ;Set bit 3 to indicate
250A F608 01770 OR 8 ; that INV given
250C 32B125 01780 LD (IVCOD+1),A
250F 0608 01790 LD B,8 ;Show vis/inv done
2511 18D6 01800 JR DOPR05 ;Merge w/prev. parms
01810 ;
01820 ; Process VIS parm
01830 ;
2513 CD6925 01840 DOVIS CALL PRSFLD ;Parse parm
2516 C22724 01850 JP NZ,ATBERR ;Quit on parm error
2519 3AB125 01860 LD A,(IVCOD+1) ;Strip bit 3
251C E6F7 01870 AND 0F7H
251E 32B125 01880 LD (IVCOD+1),A
2521 0608 01890 LD B,8 ;Show vis/inv done
2523 18C4 01900 JR DOPR05 ;Merge w/prev. parms
01910 ;
01920 ; Process USER parm
01930 ;
2525 CD6925 01940 DOUSE CALL PRSFLD ;Parse parm
2528 CA2724 01950 JP Z,ATBERR ;Quit on parm error
252B E5 01960 PUSH HL ;Save cmdline ptr
252C 113329 01970 LD DE,PSWDBUF
252F CD4727 01980 CALL DOHASH ;Hash the password
2532 223D29 01990 LD (HASHBUF+2),HL ;Set into position
2535 E1 02000 POP HL
2536 0602 02010 LD B,2 ;Show user done
2538 18A9 02020 JR DOPR04 ;Merge w/prev. parms
02030 ;
02040 ; Process OWNER parm
02050 ;
253A CD6925 02060 DOOWN CALL PRSFLD ;Parse parm
253D CA2724 02070 JP Z,ATBERR ;Quit on parm error
2540 E5 02080 PUSH HL ;Save cmdline ptr
2541 113329 02090 LD DE,PSWDBUF
2544 CD4727 02100 CALL DOHASH ;Hash the password
2547 223B29 02110 LD (HASHBUF),HL
254A E1 02120 POP HL
254B 0604 02130 LD B,4 ;Show OWNER done
254D 1894 02140 JR DOPR04 ;Merge w/prev. parms
02150 ;
02160 ; Transfer the field, 1st char alpha
02170 ;
254F 7E 02180 XSPEC8A LD A,(HL) ;P/u a filespec character
2550 23 02190 INC HL ; & 1st test for A-Z
2551 1809 02200 JR XSPEC10
2553 7E 02210 XSPEC9 LD A,(HL) ;P/u a filespec character
2554 23 02220 INC HL ;Advance to next one
2555 FE30 02230 CP '0' ;Check for 0-9
2557 D8 02240 RET C ;Quit if < 0
2558 FE3A 02250 CP '9'+1
255A 3806 02260 JR C,XSPEC11 ;Go if numeric
255C FE41 02270 XSPEC10 CP 'A' ;Check for A-Z
255E D8 02280 RET C
255F FE5B 02290 CP 'Z'+1
2561 D0 02300 RET NC
2562 12 02310 XSPEC11 LD (DE),A ;Character is valid
2563 13 02320 INC DE ;Advance to next one
2564 10ED 02330 DJNZ XSPEC9 ; & loop
2566 7E 02340 LD A,(HL) ;P/u following character
2567 23 02350 INC HL

```



```

2568 C9      02360      RET                      ;Go home
              02370 ;
              02380 ;      Parse rest of parm (ignore until separator)
              02390 ;

2569 23      02400 PRSFLD INC      HL
256A 7E      02410      LD      A,(HL)          ;Get next char
256B FE0D    02420      CP      CR              ;Ret on end of line
256D C8      02430      RET      Z
256E FE29    02440      CP      ')'            ;Ret on closing paren
2570 C8      02450      RET      Z
2571 FE2C    02460      CP      ','            ;Ret on separator
2573 C8      02470      RET      Z
2574 FE3D    02480      CP      '='            ;Assignment operator?
2576 20F1    02490      JR      NZ,PRSFLD       ;Loop if not
2578 23      02500      INC      HL
2579 7E      02510      LD      A,(HL)
257A FE22    02520      CP      '"'            ;Is quote there?
257C 2001    02530      JR      NZ,$+3
257E 23      02540      INC      HL              ;Bypass the quote
257F 11329   02550      LD      DE,PSWDBUF
2582 0608    02560      LD      B,8
2584 D5      02570      PUSH     DE
2585 C5      02580      PUSH     BC
2586 3E20    02590      LD      A,' '          ;Space out the buffer
2588 12      02600 PRSF01 LD      (DE),A
2589 13      02610      INC      DE
258A 10FC    02620      DJNZ    PRSF01
258C C1      02630      POP      BC
258D D1      02640      POP      DE
258E CD4F25  02650      CALL    XSPEC8A        ;Transfer the spec
2591 2B      02660      DEC      HL
2592 F601    02670      OR      1              ;Show got a parm
2594 C9      02680      RET

              02690 ;
              02700 ;      Routine updates file's directory data
              02710 ;

2595 ED4B4529 02720 UPDDIR LD      BC,(FCB+6)    ;P/u drive & DEC
2599          02730      @DIRRD                ;Read its directory
2599 3E57     00016      LD      A,87
259B EF      00017      RST      40
259C C20C24  02740      JP      NZ,IOERR        ;Quit on read error
259F 7E      02750      LD      A,(HL)          ;P/u attributes byte
25A0 1600    02760 PRMCOD LD      D,$-$        ;P/u parm test bits
25A2 CB42    02770      BIT      0,D            ;Was Prot entered?
25A4 2804    02780      JR      Z,UPDIR1        ;Jump if not
25A6 E6F8    02790      AND      0F8H          ;Rmv prot level
25A8 F600    02800 PROTLVL OR      0            ;Merge new prot level
25AA CB5A    02810 UPDIR1 BIT      3,D          ;Was Inv or Vis entered?
25AC 2804    02820      JR      Z,UPDIR2        ;Bypass if not
25AE E6F7    02830      AND      0F7H          ;Remove any vis/inv
25B0 F600    02840 IVCOD  OR      0            ;Merge new inv/vis
25B2 77      02850 UPDIR2 LD      (HL),A        ; & update dir rec
25B3 7D      02860      LD      A,L            ;Pt to owner pswd
25B4 C610    02870      ADD     A,16
25B6 6F      02880      LD      L,A
25B7 CB52    02890      BIT      2,D            ;Was OWN parm entered?
25B9 280A    02900      JR      Z,UPDIR3        ;Bypass if not
25BB 3A3B29  02910      LD      A,(HASHBUF)     ;Xfer new hashed pswd
25BE 77      02920      LD      (HL),A        ; into the directory
25BF 3A3C29  02930      LD      A,(HASHBUF+1)   ; OWNER psw position
25C2 23      02940      INC      HL

```

```

25C3 77      02950      LD      (HL),A
25C4 2B      02960      DEC      HL
25C5 23      02970  UPDIR3  INC      HL      ;Bypass own pswd field
25C6 23      02980      INC      HL
25C7 CB4A    02990      BIT      1,D      ;Was USER parm entered?
25C9 2809    03000      JR        Z,UPDIR4 ;Bypass if not
25CB 3A3D29  03010      LD      A,(HASHBUF+2) ;Xfer the new hashed pswd
25CE 77      03020      LD      (HL),A      ; into the directory
25CF 3A3E29  03030      LD      A,(HASHBUF+3) ; USER psw position
25D2 23      03040      INC      HL
25D3 77      03050      LD      (HL),A
25D4         03060  UPDIR4  @DIRWR      ;Write dir back to disk
25D4 3E58    00018      LD      A,88
25D6 EF      00019      RST      40
25D7 C20C24  03070      JP        NZ,IOERR      ;Abort on write error
25DA C9      03080      RET              ;Done - return
          03090 ;
          03100 ;      Change attributes of entire disk
          03110 ;
25DB 0E00    03120  PROT   LD      C,0      ;Init for drive 0
25DD 2B      03130      DEC      HL      ;Backup to separator
25DE 7E      03140      LD      A,(HL)      ;Ck for drive entered
25DF FE3A    03150      CP        ':'      ;Colon indicator?
25E1 2018    03160      JR        NZ,PROT01 ;Bypass if not
25E3 23      03170      INC      HL      ;Point to drive #
          03180 ;
          03190 ;      Is the drivespec legal (0-7) ?
          03200 ;
25E4 7E      03210      LD      A,(HL)      ;P/u drive
25E5 D630    03220      SUB      '0'      ;Cvrt to binary
25E7 3804    03230      JR        C,ILLDRVN ;Less than 0, illegal
25E9 FE08    03240      CP        7+1      ;Greater than 7 ?
25EB 3F      03250      CCF
25EC 4F      03260      LD      C,A
25ED 3E20    03270  ILLDRVN LD      A,32      ;Init "Illegal Drive #"
25EF DA0C24  03280      JP        C,IOERR
          03290 ;
          03300 ;      Drive # is legal - Check it out
          03310 ;
25F2         03320      @CKDRV      ;Do a check drive
25F2 3E21    00020      LD      A,33
25F4 EF      00021      RST      40
25F5 3E20    03330      LD      A,32      ;Init "Illegal drive
25F7 C20C24  03340      JP        NZ,IOERR      ;Go if bad
          03350 ;
25FA 23      03360      INC      HL      ;Bump line pointer
25FB 79      03370  PROT01 LD      A,C
25FC 321427  03380      LD      (TSTMPW+1),A ;Stuff drive for later
25FF         03390      @GTDCT      ;Get DCT -> IY
25FF 3E51    00022      LD      A,81
2601 EF      00023      RST      40
2602 110A29  03400      LD      DE,PRMTBL$ ;Get parms
2605         03410      @PARAM
2605 3E11    00024      LD      A,17
2607 EF      00025      RST      40
2608 C20C24  03420      JP        NZ,IOERR      ;Quit on parm error
260B 3A6626  03430      LD      A,(PPARM+1) ;Make sure a parm
260E 218926  03440      LD      HL,LPARM+1 ; was entered
2611 B6      03450      OR       (HL)
2612 219326  03460      LD      HL,UPARM+1
2615 B6      03470      OR       (HL)

```

The Source	LIBRARY Files	ATTRIB - LS-DOS 6.2	Page 00007
2616 212426	03480	LD HL,NPARM+1	
2619 B6	03490	OR (HL)	
261A CA2324	03500	JP Z,NOTDUN	;Quit if none entered
261D CD1327	03510	CALL TSTMPW	;Test master password
2620 C20C24	03520	JP NZ,IOERR	;Quit on error
2623 110000	03530 NPARM	LD DE,0	;P/U Name parm
2626 7A	03540	LD A,D	
2627 B3	03550	OR E	
2628 283B	03560	JR Z,PPARM	;Jump if name not entered
262A 218828	03570	LD HL,PACKNM\$;Get name into buf1
262D CD4A27	03580	CALL GMPW1	
2630 C20C24	03590	JP NZ,IOERR	;Quit on error
2633 21002A	03600	LD HL,BUFFER	;HL = new name
2636 11D02B	03610	LD DE,GATBUF+0D0H	;Where new name goes
2639 D5	03620	PUSH DE	
263A E5	03630	PUSH HL	
263B 010800	03640	LD BC,8	;Name len
263E EDB0	03650	LDIR	;Xfer new name
2640 E1	03660	POP HL	
2641 110200	03670	LD DE,2	;Trk 0, sect 2
2644 3A1427	03680	LD A,(TSTMPW+1)	;P/u drive
2647 4F	03690	LD C,A	
2648	03700	@@RDSEC	;Read SIS
2648 3E31	00026	LD A,49	
264A EF	00027	RST 40	
264B E1	03710	POP HL	;Get Name pointer
264C C20C24	03720	JP NZ,IOERR	;Quit if read error
264F C5	03730	PUSH BC	;Save drive
2650 11102A	03740	LD DE,BUFFER+10H	;Point to where name goes
2653 010800	03750	LD BC,8	
2656 EDB0	03760	LDIR	;Xfer new name to SIS
2658 C1	03770	POP BC	;Recover drive
2659 110200	03780	LD DE,2	
265C 21002A	03790	LD HL,BUFFER	
265F	03800	@@WRSEC	;Write new SIS
265F 3E35	00028	LD A,53	
2661 EF	00029	RST 40	
2662 C20C24	03810	JP NZ,IOERR	;Quit on write error
2665 110000	03820 PPARM	LD DE,0	;Was PW parm used?
2668 7A	03830	LD A,D	
2669 B3	03840	OR E	
266A 280C	03850	JR Z,PROT02	;Jump if PW not entered
266C 21A028	03860	LD HL,NEWMPW\$;Buffer for new MPW
266F CD3A27	03870	CALL GETMPW	;Input the new one
2672 C20C24	03880	JP NZ,IOERR	;Quit if bad PW
2675 22CE2B	03890	LD (GATBUF+0CEH),HL	;Stuff PW
2678 FD5609	03900 PROT02	LD D,(IY+9)	;Dir cl => reg D
267B 3A1427	03910	LD A,(TSTMPW+1)	;P/u drive
267E 4F	03920	LD C,A	
267F CDCD27	03930	CALL GATWR	;Write sector 0 from buf
2682 C20C24	03940	JP NZ,IOERR	;Jump on write error
2685 2ACE2B	03950	LD HL,(GATBUF+0CEH)	;P/u pack MPW
	03960 ;		
	03970 ;	Check on Lock or Unlock	
	03980 ;		
2688 010000	03990 LPARM	LD BC,0	;Lock parm used?
268B 78	04000	LD A,B	
268C B1	04010	OR C	
268D 119642	04020	LD DE,BLNKMPW	;P/u blank MPW for test
2690 2007	04030	JR NZ,PROT03	;Jump if LOCK entered
2692 010000	04040 UPARM	LD BC,0	;Unlock parm used?

The Source	LIBRARY Files	ATTRIB - LS-DOS 6.2	Page 00008
2695 78	04050	LD A,B	
2696 B1	04060	OR C	
2697 C8	04070	RET Z	;Neither LOCK or UNLOCK
2698 EB	04080	EX DE,HL	;Switch New & Test MPW
	04090 ;		
	04100 ;	Lock to pack MPW or unlock pswds to blanks	
	04110 ;		
2699 22EA26	04120 PROT03	LD (REVMPW+1),HL	;Stuff New MPW
269C ED53E126	04130	LD (THISPW+1),DE	;Stuff test MPW
26A0 3A1427	04140	LD A,(TSTMPW+1)	;P/u drive #
26A3 4F	04150	LD C,A	
26A4 FD5609	04160	LD D,(IY+9)	;Get dir cyl => reg D
26A7 1E01	04170	LD E,1	;Point to HIT
26A9 21002C	04180	LD HL,HITBUF	;Point to HIT buffer
26AC	04190	@@RDSSC	;Read it into buffer
26AC 3E55	00030	LD A,85	
26AE EF	00031	RST 40	
26AF C20C24	04200	JP NZ,IOERR	;Quit on read error
26B2 7E	04210 PROT04	LD A,(HL)	;P/u a DEC
26B3 B7	04220	OR A	
26B4 2842	04230	JR Z,PROT09	;Loop on spare
26B6 45	04240	LD B,L	;Put DEC in reg B
26B7 C5	04250	PUSH BC	; & save it in stack
26B8 7D	04260	LD A,L	;Ck if this DEC points
26B9 E6E0	04270	AND 0E0H	; to same dir sector as
26BB 6F	04280	LD L,A	; the previous DEC
26BC A8	04290	XOR B	
26BD FEFF	04300 PROT05	CP 0FFH	;1st time, no DEC
26BF 280D	04310	JR Z,PROT06	;Jump if the same
26C1 32BE26	04320	LD (PROT05+1),A	;Save it for testing
26C4	04330	@@DIRRD	;Read this dir sector
26C4 3E57	00032	LD A,87	
26C6 EF	00033	RST 40	
26C7 C20C24	04340	JP NZ,IOERR	;Quit on read error
26CA 7C	04350	LD A,H	;Set hi-order SBUFF\$
26CB 32CF26	04360	LD (PROT06+1),A	
26CE 2600	04370 PROT06	LD H,\$-\$;Point to buf hi-order
26D0 7E	04380	LD A,(HL)	;P/u type code
26D1 E6F8	04390	AND 0F8H	;Remove protection
26D3 FE10	04400	CP 10H	;Jump if INV, SYS, FXDE
26D5 201D	04410	JR NZ,PROT08	
26D7 7D	04420	LD A,L	;Point to password fields
26D8 C610	04430	ADD A,16	
26DA 6F	04440	LD L,A	
26DB D5	04450	PUSH DE	;Save reg DE
26DC E5	04460 PROT07	PUSH HL	;Save pointer to OWNER
26DD 5E	04470	LD E,(HL)	;P/u owner MPW
26DE 23	04480	INC HL	
26DF 56	04490	LD D,(HL)	
26E0 210000	04500 THISPW	LD HL,\$-\$;P/u test MPW & see
26E3 AF	04510	XOR A	; if this one matches
26E4 ED52	04520	SBC HL,DE	
26E6 E1	04530	POP HL	;Restore ptr to OWN
26E7 200A	04540	JR NZ,PROT07B	;Don't change if diff
26E9 110000	04550 REVMPW	LD DE,\$-\$; else p/u new MPW
26EC 73	04560	LD (HL),E	; & insert it
26ED 2C	04570	INC L	
26EE 72	04580	LD (HL),D	
26EF 2C	04590	INC L	
26F0 73	04600	LD (HL),E	;Change USER pw
26F1 2C	04610	INC L	


```

26F2 72      04620      LD      (HL),D
26F3 D1      04630 PROT07B POP      DE      ;Restore reg DE
26F4 C1      04640 PROT08 POP      BC      ;Recover DEC
26F5 262C    04650      LD      H,HITBUF<-8 ;Point to HIT hi-order
26F7 68      04660      LD      L,B      ;Stuff HIT lo-order
26F8 7D      04670 PROT09 LD      A,L      ;Point to next entry
26F9 C620    04680      ADD     A,32      ; for this dir sector
26FB 6F      04690      LD      L,A
26FC 30B4    04700      JR      NC,PROT04 ;Jump if still in same
26FE 3ABE26  04710      LD      A,(PROT05+1) ;P/u current DEC
2701 AD      04720      XOR      L
2702 2008    04730      JR      NZ,PROT10 ;Jump if different
2704 E5      04740      PUSH    HL
2705         04750      @DIRWR ;Write out this sector
2705 3E58    04760      LD      A,88
2707 EF      04770      RST      40
2708 E1      04780      POP      HL
2709 C20C24  04790      JP      NZ,IOERR ;Quit on write error
270C 7D      04780 PROT10 LD      A,L      ;Advance to the next
270D 2C      04790      INC     L      ; directory sector
270E FE1F    04800      CP      1FH      ;At end of disk?
2710 20A0    04810      JR      NZ,PROT04 ;Loop if not
2712 C9      04820      RET      ; else go home
          04830 ;
          04840 ;
          04850 ;
          04860 TSTMPW LD      C,$-$      ;Init to drive requested
2713 0E00    04870      CALL    GATRD      ;Read GAT into GATBUF
2715 CDCC27  04880      RET      NZ      ;Back on error
2718 C0      04890      LD      HL,(GATBUF+0CEH)
2719 2ACE2B  04900      LD      DE,PASSWORD ;Password=PASSWORD?
271C 11E042  04910      XOR      A
271F AF      04920      SBC     HL,DE
2720 ED52    04930      RET      Z      ;Back if PASSWORD
2722 C8      04940 ;
          04950 ;
          04960 ;
          04970 MPARM LD      DE,0      ;P/u MPW string addr
2723 110000  04980      LD      HL,CURMPW$ ;Init prompt
2726 21B828  04990      CALL    GETMPW ;Hash parm or entry
2729 CD3A27  05000      RET      NZ      ;Back on bad PW
272C C0      05010      EX      DE,HL ;Xfer hashed MPW to DE
272D EB      05020      LD      HL,(GATBUF+0CEH) ;Grab pack MPW &
272E 2ACE2B  05030      XOR      A ; check if user entered
2731 AF      05040      SBC     HL,DE ; the pack MPW
2732 ED52    05050      LD      HL,BADMPW$ ;Init error pointer
2734 21D028  05060      LD      A,63 ;Set extended error
2737 3E3F    05070      RET      ;Z or NZ
2739 C9      05080 ;
          05090 ;
          05100 ;
          05110 GETMPW CALL    GMPW1 ;Get MPW into buffer
273A CD4A27  05120      JR      Z,DOHASH
273D 2808    05130      CP      63 ;Extended error?
273F FE3F    05140      RET      NZ
2741 C0      05150      LD      HL,BADMPW$ ;Switch error message
2742 21D028  05160      OR      A ; to password error
2745 B7      05170      RET
2746 C9      05180 DOHASH LD      A,0E4H ;Hash password (DE) to HL
2747 3EE4    05190      RST      28H ;Ret to what called
2749 EF      05200 ;

```

```

05210 ;      Routine places a password field into buffer
05220 ;
274A 7A      05230 GMPW1 LD      A,D          ;Test if user entered MPW
274B B3      05240      OR      E
274C 281D    05250      JR      Z,GMPW3      ;Prompt if not
274E 3C      05260      INC     A          ; or if no operand
274F 281A    05270      JR      Z,GMPW3
05280 ;
05290 ;      Place entered password into buffer
05300 ;
2751 21002A  05310      LD      HL,BUFFER    ;Point to buffer
2754 E5      05320      PUSH    HL
2755 0608     05330      LD      B,8          ;Init for 8 chars
2757 1A      05340 GMPW2 LD      A,(DE)      ;P/u a char
2758 FE0D     05350      CP      CR          ;End of line?
275A 282F     05360      JR      Z,GMPW4
275C FE2C     05370      CP      ','         ;Comma separator?
275E 282B     05380      JR      Z,GMPW4
2760 FE22     05390      CP      '"'         ;Closing quote?
2762 2827     05400      JR      Z,GMPW4
2764 13      05410      INC     DE          ;Bump input pointer
2765 77      05420      LD      (HL),A      ;Transfer character
2766 23      05430      INC     HL          ;Bump output pointer
2767 10EE     05440      DJNZ    GMPW2      ;Loop until done
2769 1825     05450      JR      CKMPW
05460 ;
05470 ;      MPW not entered - Prompt & fetch
05480 ;
276B CDF827  05490 GMPW3 CALL    CKINDO      ;Can't prompt in <DO>
276E C0      05500      RET      NZ
276F         05510      @@DSPLY      ;Display prompt
00036       00036      IFEQ     00H,1
00037       00037      LD      HL,
00038       00038      ENDIF
276F 3E0A     00039      LD      A,10
2771 EF      00040      RST      40
2772 C0      05520      RET      NZ
2773 010008   05530      LD      BC,8<8      ;Init for 8 chars
2776 21002A   05540      LD      HL,BUFFER    ;Point to buffer
2779 E5      05550      PUSH    HL
277A         05560      @@KEYIN      ;Get parm input
277A 3E09     00041      LD      A,9
277C EF      00042      RST      40
277D DA1B24  05570      JP      C,ABORT      ;Quit on Break
2780 EB      05580      EX      DE,HL        ;Start pointer to reg DE
2781 2600     05590      LD      H,0         ;Calculate trailing
2783 68      05600      LD      L,B          ; spaces needed for MPW
2784 19      05610      ADD     HL,DE
2785 3E08     05620      LD      A,8
2787 90      05630      SUB     B
2788 2806     05640      JR      Z,CKMPW      ;Go if 8 chars entered
278A 47      05650      LD      B,A          ;Set loop count
278B 3620     05660 GMPW4 LD      (HL),' '    ; and fill to end
278D 23      05670      INC     HL          ; with spaces
278E 10FB     05680      DJNZ    GMPW4
05690 ;
05700 ;      Convert (SP) through (SP)+7 to upper case
05710 ;
2790 E1      05720 CKMPW POP     HL          ;Get buffer start
2791 E5      05730      PUSH    HL
2792 0608     05740      LD      B,8          ;Init loop 8 chars

```

```

2794 7E      05750      LD      A,(HL)      ;P/u 1st char
2795 180E    05760      JR      CKMPW2      ; & check <A-Z>
2797 23      05770 CKMPW1 INC      HL
2798 7E      05780      LD      A,(HL)
2799 FE20    05790      CP      ' '      ;Got to a space?
279B 2823    05800      JR      Z,CKMPW7
279D FE30    05810      CP      '0'      ;Less than '0' is error
279F 3823    05820      JR      C,INVNAM
27A1 FE3A    05830      CP      '9'+1      ;<0-9> is okay for 2-n
27A3 3812    05840      JR      C,CKMPW3
27A5 FE41    05850 CKMPW2 CP      'A'      ;Less than "A" is error
27A7 381B    05860      JR      C,INVNAM
27A9 FE5B    05870      CP      'Z'+1      ;<A-Z> is okay
27AB 380A    05880      JR      C,CKMPW3
27AD FE61    05890      CP      'a'      ;Ck convert to upper
27AF 3813    05900      JR      C,INVNAM
27B1 FE7B    05910      CP      'z'+1
27B3 300F    05920      JR      NC,INVNAM
27B5 CBAE    05930      RES      5,(HL)
27B7 10DE    05940 CKMPW3 DJNZ    CKMPW1      ;Loop if more
27B9 D1      05950 CKMPW4 POP      DE      ;Point to buffer start
27BA AF      05960      XOR      A      ;Set Z = good
27BB C9      05970      RET
                05980 ;
27BC 23      05990 CKMPW5 INC      HL
27BD BE      06000      CP      (HL)      ;No imbedded spaces
27BE 2004    06010      JR      NZ,INVNAM
27C0 10FA    06020 CKMPW7 DJNZ    CKMPW5      ;A space found, now
27C2 18F5    06030      JR      CKMPW4      ; must be all spaces
27C4 21E828  06040 INVNAM LD      HL,BADNAM$      ;Pt to error string
27C7 3E3F    06050      LD      A,63      ;Init extended error
27C9 B7      06060      OR      A      ;Set NZ
27CA D1      06070      POP      DE
27CB C9      06080      RET
                06090 ;
                06100 ;      Read or write the granule allocation table
                06110 ;
27CC F6      06120 GATRD  DB      0F6H      ;Set NZ for test
27CD AF      06130 GATWR  XOR      A      ;Set Z for test
27CE D5      06140      PUSH    DE
27CF E5      06150      PUSH    HL
27D0 F5      06160      PUSH    AF
27D1 FDE5    06170      PUSH    IY
27D3         06180      @@GTDCT      ;DCT to reg IY
27D3 3E51    00043      LD      A,81
27D5 EF      00044      RST      40
27D6 FD5609  06190      LD      D,(IY+9)      ;Get dir track
27D9 FDE1    06200      POP      IY
27DB 21002B  06210      LD      HL,GATBUF
27DE 5D      06220      LD      E,L      ;Set to sector 0
27DF F1      06230      POP      AF
27E0 2807    06240      JR      Z,GATRW1      ;Go if GAT write
27E2         06250      @@RDSSC
27E2 3E55    00045      LD      A,85
27E4 EF      00046      RST      40
27E5 3E14    06260      LD      A,14H      ;Init "GAT read error
27E7 180C    06270      JR      GATRW3
27E9         06280 GATRW1 @@WRSSC
27E9 3E36    00047      LD      A,54
27EB EF      00048      RST      40
27EC 2003    06290      JR      NZ,GATRW2      ;Skip verify if error

```

```

27EE      06300      @VRSEC      ;Verify the write
27EE 3E32      00049      LD      A,50
27F0 EF      00050      RST      40
27F1 FE06      06310 GATRW2 CP      6      ;Error 6 expected
27F3 3E15      06320      LD      A,15H      ;Init "GAT write error"
27F5 E1      06330 GATRW3 POP      HL
27F6 D1      06340      POP      DE
27F7 C9      06350      RET
      06360 ;
27F8 FDE5      06370 CKIND0 PUSH    IY
27FA      06380      @FLAGS
27FA 3E65      00051      LD      A,101
27FC EF      00052      RST      40
27FD FDCB126E 06390      BIT      5,(IY+'S'-'A') ;Ck on DO in effect
2801 FDE1      06400      POP      IY
2803 C8      06410      RET      Z      ;Go back if not in DO
2804 216328      06420      LD      HL,NOINDO$      ; else set error code
2807 3E3F      06430      LD      A,63      ;Set extended error
2809 C9      06440      RET      ; & back with NZ
      06450 ;
      06460 ;      Messages
      06470 ;
280A 46      06480 SPCREQ$ DB      'File spec required',CR
      69 6C 65      20 73 70 65 63
      20 72 65      71 75 69 72 65
      64 0D
281D 53      06490 NOTDUN$ DB      'Specifications Required',CR
      70 65 63      69 66 69 63 61
      74 69 6F      6E 73 20 52 65
      71 75 69      72 65 64 0D
2835 41      06500 ATBERR$ DB      'Attribute specification error',CR
      74 74 72      69 62 75 74 65
      20 73 70      65 63 69 66 69
      63 61 74      69 6F 6E 20 65
      72 72 6F      72 0D
2853 4E      06510 PROTS$ DB      'NOEXREUPWRRNRMFU'
      4F 45 58      52 45 55 50 57
      52 52 4E      52 4D 46 55
2863 49      06520 NOINDO$ DB      'Invalid command during DO processing',CR
      6E 76 61      6C 69 64 20 63
      6F 6D 6D      61 6E 64 20 64
      75 72 69      6E 67 20 44 4F
      20 70 72      6F 63 65 73 73
      69 6E 67      0D
2888 4E      06530 PACKNM$ DB      'New disk pack name ? ',3
      65 77 20      64 69 73 6B 20
      70 61 63      6B 20 6E 61 6D
      65 20 3F      20 20 20 03
28A0 4E      06540 NEWMPW$ DB      'New master password ? ',3
      65 77 20      6D 61 73 74 65
      72 20 70      61 73 73 77 6F
      72 64 20      3F 20 20 03
28B8 4D      06550 CURMPW$ DB      'Master password ? ',3
      61 73 74      65 72 20 70 61
      73 73 77      6F 72 64 20 3F
      20 20 20      20 20 20 03
28D0 49      06560 BADMPW$ DB      'Invalid master password',CR
      6E 76 61      6C 69 64 20 6D
      61 73 74      65 72 20 70 61
      73 73 77      6F 72 64 0D
28E8 49      06570 BADNAM$ DB      'Invalid disk name',CR

```



```

        6E 76 61 6C 69 64 20 64
        69 73 6B 20 6E 61 6D 65
        0D
28FA 43      06580 ABORT$ DB      'Command aborted',CR
        6F 6D 6D 61 6E 64 20 61
        62 6F 72 74 65 64 0D
                06590 ;
290A 80      06600 PRMTBL$ DB      80H
                06610 ;
0080      06620 VAL      EQU      80H
0040      06630 SW      EQU      40H
0020      06640 STR      EQU      20H
0010      06650 SGL      EQU      10H
                06660 ;
290B 62      06670      DB      SW!STR!2,'PW',0
        50 57 00
290F 6626    06680      DW      PPARAM+1
2911 54      06690      DB      SW!SGL!4,'LOCK',0
        4C 4F 43 4B 00
2917 8926    06700      DW      LPARAM+1
2919 56      06710      DB      SW!SGL!6,'UNLOCK',0
        55 4E 4C 4F 43 4B 00
2921 9326    06720      DW      UPARAM+1
2923 74      06730      DB      SW!STR!SGL!4,'NAME',0
        4E 41 4D 45 00
2929 2426    06740      DW      NPARAM+1
292B 73      06750      DB      SW!STR!SGL!3,'MPW',0
        4D 50 57 00
2930 2427    06760      DW      MPARAM+1
2932 00      06770      NOP
                06780 ;
0008      06790 PSWDBUF DS      8      ;Password buffer
0004      06800 HASHBUF DS      4      ;Owner & user hashes
0020      06810 FCB      DS      32
                06820 ;
2A00      06830      ORG      $<-8+1<+8
0100      06840 BUFFER DS      256
0100      06850 GATBUF  DS      256
0100      06860 HITBUF  DS      256
                06870 ;
2400      06880      END      ATTRIB

```

@@1	0000 @@2	0000 @@3	0000
@@4	0000 @MOD2	0000 @MOD4	FFFF
ABORT	241B ABORT\$	28FA ATBERR	2427
ATBERR\$	2835 ATT0	2468 ATT01	247B
ATT02	247E ATT1	247F ATT2	248B
ATT3	248D ATTRIB	2400 ATTRIB1	2437
BADMPW\$	28D0 BADNAM\$	28E8 BLNKMPW	4296
BUFFER	2A00 CKINDO	27F8 CKMPW	2790
CKMPW1	2797 CKMPW2	27A5 CKMPW3	27B7
CKMPW4	27B9 CKMPW5	27BC CKMPW7	27C0
CR	000D CURMPW\$	28B8 DOHASH	2747
DOINV	2501 DOOWN	253A DOPR01	24B4
DOPR02	24C1 DOPR03	24DD DOPR04	24E3
DOPR05	24E9 DOUSE	2525 DOVIS	2513
EXTERR	242A FCB	293F GATBUF	2B00
GATRD	27CC GATRW1	27E9 GATRW2	27F1
GATRW3	27F5 GATWR	27CD GETMPW	273A
GMPW1	274A GMPW2	2757 GMPW3	276B
GMPW4	278B HASHBUF	293B HITBUF	2C00
ILLDRVN	25ED INVNAM	27C4 IOERR	240C
IVCOD	25B0 LPARM	2688 MPARM	2723
NEWMPW\$	28A0 NOINDO\$	2863 NOTDUN	2423
NOTDUN\$	281D NPARM	2623 PACKNM\$	2888
PASSWORD	42E0 PPARM	2665 PRMCOB	25A0
PRMTBL\$	290A PROT	25DB PROT01	25FB
PROT02	2678 PROT03	2699 PROT04	26B2
PROT05	26BD PROT06	26CE PROT07	26DC
PROT07B	26F3 PROT08	26F4 PROT09	26F8
PROT10	270C PROTLVL	25A8 PROTS\$	2853
PRSF01	2588 PRSFLD	2569 PSWDBUF	2933
REVMPW	26E9 SAVESP	2430 SGL	0010
SPCREQ	241F SPCREQ\$	280A STR	0020
SW	0040 THISPW	26E0 TSTMPW	2713
UPARM	2692 UPDIR	2595 UPDIR1	25AA
UPDIR2	25B2 UPDIR3	25C5 UPDIR4	25D4
VAL	0080 XSPEC10	255C XSPEC11	2562
XSPEC8A	254F XSPEC9	2553 @@ABORT	ACED
@@ADTSK	AD80 @@BANK	B298 @@BKSP	AF78
@@BREAK	B2AE @@CHNIO	ACD8 @@CKBRKC	B2FC
@@CKDRV	ADD4 @@CKEOF	AF8D @@CKTSK	AD6B
@@CLOSE	AF63 @@CLS	B2E6 @@CMNDI	AD17
@@CMNDR	AD2C @@CTL	AB3C @@DATE	ACAE
@@DCSTAT	AE13 @@DEBUG	AD56 @@DECHEX	B218
@@DIRRD	B185 @@DIRWR	B19A @@DIV16	B203
@@DIV8	B1EE @@DODIR	ADE9 @@DSP	AB00
@@DSPLY	ABA0 @@ERROR	AD41 @@EXIT	AD02
@@FEXT	B0F2 @@FLAGS	B282 @@FNAME	B107
@@FSPEC	B0DD @@GATRD	B170 @@GATWR	B1AF
@@GET	AB14 @@GTDCB	B131 @@GTDCB	B11C
@@GTMOD	B146 @@HDFMT	AE8B @@HEX16	B257
@@HEX8	B242 @@HEXDEC	B22D @@HIGH\$	B26C
@@INIT	AF39 @@KBD	AB78 @@KEY	AAEC
@@KEYIN	AB8C @@KLTSK	ADBF @@LOAD	B0B3
@@LOC	AFA2 @@LOF	AFB7 @@LOGGER	ABD7
@@LOGOT	ABEC @@MSG	AC23 @@MUL16	B1D9
@@MUL8	B1C4 @@OPEN	AF4E @@PARAM	AC99
@@PAUSE	AC84 @@PEOF	AFCC @@POSN	AFE1
@@PRINT	AC38 @@PRT	AB50 @@PUT	AB28
@@RAMDIR	ADFE @@RDSEC	AE91 @@RDSSC	B15B
@@READ	AFF6 @@REMOV	AF24 @@RENAM	AF0F

@@REW	B00B @@RMTSK	AD95 @@RPTSK	ADAA
@@RREAD	B020 @@RSLCT	AE7C @@RSTOR	AE3D
@@RUN	B0C8 @@RWIT	B035 @@SEEK	AE67
@@SEEKSC	B04A @@SKIP	B05F @@SLCT	AE28
@@STEPI	AE52 @@TIME	ACC3 @@VDCTL	AC6F
@@VER	B074 @@VRSEC	AEA6 @@WEOF	B089
@@WHERE	AB64 @@WRITE	B09E @@WRSEC	AED0
@@WRSSC	AEE5 @@WRTRK	AEFA	

2400 is the transfer address
00000 Total errors

NOTES:

NOTES:

Command: AUTO

Library: SYS7/SYS

ISAM # : 11H

```

00100 ;LBAUTO/ASM - AUTO Command
00000 00110 TITLE <AUTO - LS-DOS 6.2>
00120 ;
00130 ;
00000 00140 *GET SVC MAC:3 ;SVC Macro equivalents
00010 ;SVC MAC/ASM - LS-DOS Version VI
00020 *LIST OFF
00390 *LIST ON
00000 00150 *GET VALUES:3 ;Misc. equates
00390 ;VALUES/ASM - Version 6
00390 *LIST OFF
00420 *LIST ON
00160 ;
0002 00170 SECTOR EQU 0<8+2
0020 00180 OFFSET EQU 20H ;Offset to Auto Buffer
0028 00190 RST28 EQU 28H ;SVC restart
0019 00200 @CMNDR EQU 25 ;Execute & return
0031 00210 @RDSEC EQU 49 ;Read a sector SVC #
0035 00220 @WRSEC EQU 53 ;Write a sector SVC #
00230 ;
2400 00240 ORG 2400H
00250 ;
00260 AUTO
2400 ED730B24 00270 LD (SAVE SP+1),SP ;Save SP address
2404 CD0E24 00280 CALL PARSE ;Do AUTO command
2407 2100000 00290 LD HL,0 ;Set no error
00300 ;
00310 ; Reinstall SP & RETurn
00320 ;
240A 3100000 00330 SAVESP LD SP,$-$ ;P/u old SP
240D C9 00340 RET
00350 ;
00360 ; Skip any leading spaces
00370 ;
240E 2B 00380 PARSE DEC HL ;Back one
240F 23 00390 PARSE2 INC HL ;Bump buff ptr
2410 7E 00400 LD A,(HL) ;P/u character
2411 FE20 00410 CP ' ' ;Space ?
2413 28FA 00420 JR Z,PARSE2 ;Go til non-space
00430 ;
00440 ; Display auto Buffer on drive :d ?
00450 ;
2415 FE3F 00460 CP '?' ;Display auto buffer ?
2417 2809 00470 JR Z,DISAUTO
00480 ;
00490 ; Execute Auto Buffer on drive :d ?
00500 ;
2419 FE3D 00510 CP '=' ;Execute auto buffer ?
241B 280F 00520 JR Z,EXAUTO
00530 ;
00540 ; Install the command in the auto buffer
00550 ;
241D CD5124 00560 CALL GETDRV ;P/u drive #
2420 184D 00570 JR WRBUFF ;Write new auto & RETurn
00580 ;
00590 ; Display command in auto buffer
00600 ;
2422 CD4724 00610 DISAUTO CALL GETAUTO ;P/u drive #
2425 00620 @@LOGOT ;Display it
00001 IFEQ 00H,1
00002 LD HL,

```

```

00003      ENDIF
2425 3E0C    00004      LD      A,12
2427 EF      00005      RST     40
2428 C2A024  00630      JP      NZ,IOERR      ;NZ - I/O Error
242B C9      00640      RET                ;Z - done
          00650 ;
          00660 ;      Execute command in auto buffer
          00670 ;
242C CD4724  00680 EXAUTO CALL    GETAUTO      ;P/u drive #
242F          00690      @@FLAGS
242F 3E65    00006      LD      A,101
2431 EF      00007      RST     40
2432 7E      00700      LD      A,(HL)      ;P/u first character
2433 FE2A    00710      CP      '*'          ;Non-<BREAK>able auto ?
2435 2005    00720      JR      NZ,DOAUTO     ;No - do it
          00730 ;
          00740 ;      Non <BREAK>able auto command - skip "*"
          00750 ;
2437 23      00760      INC     HL            ;Bump to next char
2438 FDCB12E6 00770      SET     4,(IY+SFLAG$) ;Disable <BREAK>
          00780 ;
          00790 ;      Exit via @CMNDI or @CMNDR if requested
          00800 ;
243C FDCB024E 00810 DOAUTO BIT     1,(IY+CFLAG$) ;If CMNDR executing, then
2440 3E19    00820      LD      A,@CMNDR      ; exit via CMNDR
2442 C22800  00830      JP      NZ,RST28
2445 3D      00840      DEC     A            ; else readjust to
2446 EF      00850      RST     40            ; CMNDI & exit
          00860 ;
          00870 ;      GETAUTO - Get Auto Command from BOOT sector
          00880 ;
          00890 ;      Advance to next byte in line
          00900 ;
2447 23      00910 GETAUTO INC     HL            ;Bump to next char
2448 7E      00920      LD      A,(HL)      ; & p/u 1st char of drvspec
2449 CD5124  00930      CALL    GETDRV        ;Get drive #
244C CD8424  00940      CALL    RDBOOT       ;Read BOOT sector
244F EB      00950      EX      DE,HL        ;Pt HL to auto buffer
2450 C9      00960      RET                ;RETurn
          00970 ;
          00980 ;      GETDRV - Check if (HL) contains a legal drive #
          00990 ;
2451 FE0E    01000 GETDRV CP      CR+1          ;No drivespec ?
2453 D8      01010      RET     C            ;Use drive 0 as default
          01020 ;
          01030 ;      Drivespec indicator entered ?
          01040 ;
2454 FE3A    01050      CP      ':'          ;Drivespec indicator ?
2456 C0      01060      RET     NZ          ;No - use drive 0
          01070 ;
          01080 ;      Drivespec ":" entered - p/u drive #
          01090 ;
2457 23      01100      INC     HL            ;Position to drive #
2458 46      01110      LD      B,(HL)      ;P/u drive #
          01120 ;
          01130 ;      Position to command following drivespec
          01140 ;
2459 23      01150      INC     HL            ;Position to command
245A 7E      01160      LD      A,(HL)      ;C/R following spec ?
245B FE0E    01170      CP      CR+1
245D 3801    01180      JR      C,SKIPINC    ;Yes - don't INC

```

```

245F 23      01190      INC      HL      ;Position to command
              01200 ;
              01210 ;      Is the drive number in legal range ?
              01220 ;
2460 78      01230 SKIPINC LD      A,B      ;P/u Drive #
2461 D630    01240      SUB      '0'      ;Less than "0" ?
2463 329624  01250      LD      (DRIVE+1),A ;Stuff away drive #
2466 3803    01260      JR      C,ILDRIVE  ;C - Illegal drive #
2468 FE08    01270      CP      7+1      ;Greater than 7 ?
246A D8      01280      RET      C      ;RETurn if legal
              01290 ;
              01300 ;      Illegal Drive Number - Display & abort
              01310 ;
246B 3E20    01320 ILDRIVE LD      A,32      ;Init "Illegal drive #
246D 1831    01330      JR      IOERR      ;Abort
              01340 ;
              01350 ;      WRBUFF - Write Auto Command to Boot
              01360 ;
246F CD8424  01370 WRBUFF CALL    RDBOOT      ;Read in BOOT
              01380 ;
              01390 ;      Skip Leading spaces
              01400 ;
2472 2B      01410      DEC      HL      ;Skip leading spaces
2473 23      01420 SKPSPCS INC      HL
2474 7E      01430      LD      A,(HL)      ;P/u char
2475 FE20    01440      CP      ' '      ;Space ?
2477 28FA    01450      JR      Z,SKPSPCS   ;Yes - skip it
              01460 ;
              01470 ;      Transfer command into buffer
              01480 ;
2479 7E      01490 XFLP   LD      A,(HL)      ;P/u byte
247A EDA0    01500      LDI      ;(HL) => (DE) INC HL & DE
247C FE0E    01510      CP      CR+1      ;Eol ?
247E 30F9    01520      JR      NC,XFLP
              01530 ;
              01540 ;      RDBOOT/WRBOOT - Read/Write BOOT sector
              01550 ;
              01560 ;      Set A = @WRSEC Supervisory CALL #
              01570 ;
2480 3E35    01580 WRBOOT LD      A,@WRSEC   ;Write Sector SVC #
2482 180D    01590      JR      BOOTIO      ;Go to I/O routine
              01600 ;
              01610 ;      RDBOOT - Check if the Drive is there
              01620 ;
2484 3A9624  01630 RDBOOT LD      A,(DRIVE+1) ;P/u drive #
2487 4F      01640      LD      C,A      ;Xfer to C for @CKDRV
2488      01650      @CKDRV ;Drive alive
2488 3E21    00008      LD      A,33
248A EF      00009      RST      40
248B 3E20    01660      LD      A,32      ;Init "Illegal drive #
248D 2011    01670      JR      NZ,IOERR      ;Abort if bad drive
              01680 ;
              01690 ;      Read GAT from directory if Model II
              01700 ;
              01710 ;
              01720 ;      IF @MOD2
              01730 ;      PUSH IY
              01740 ;      @GTDCT
248F      00010      LD      A,81
              00011      RST      40
              01740      LD      D,(IY+9)      ;Dir cyl
              01750      LD      A,(IY+3)      ;Get DCT data

```

```

01760      LD      E,(IY+4)      ;Get dct data
01770      POP     IY             ;Restore
01780      AND     28H            ;8" floppy?
01790      CP      20H           ;Yes?
01800      JR      NZ,SETSYSI    ;Nope, sysinfo on 0
01810      LD      A,E           ;Get +4
01820      AND     50H           ;Bit 6/4
01830      CP      40H           ;DD not alien?
01840      JR      NZ,SETSYSI    ;Go if alien
01850      ;
01860      PUSH    HL             ;Save
01870      LD      HL,BUFF       ;Start buffer
01880      LD      E,0           ;GAT table
248F      01890    @RDSSC      ;Read directory
00012      LD      A,85
00013      RST     40
01900      LD      DE,(BUFF+0CDH) ;Get GAT info byte
01910      POP     HL            ;Restore buffer
01920      JR      NZ,IOERR      ;Go on disk error
01930      BIT     7,E           ;System disk?
01940 SETSYSI LD      DE,SECTOR ;Sysinfo sector
01950      JR      NZ,$+3        ;Go if data disk
01960      INC     D             ;Sysinfo on cyl 1
01970      LD      (SYSINFO),DE ;Save sysinfo sector
01980      ENDIF
248F 3E31 01990      LD      A,@RDSEC      ;A = @RDSEC SVC #
02000      ;
02010      ;      Pt HL => Buffer, DE = T/S, C = :d, A = SVC #
02020      ;
2491 E5    02030 BOOTIO PUSH    HL             ;Save command ptr
2492 210025 02040      LD      HL,BUFF       ;HL => I/O buffer
2495 0E00    02050 DRIVE  LD      C,$-$      ;P/u drive # in C
02060      ;
2497 110200 02070      LD      DE,SECTOR     ;DE = Track 0, Sector 2
2498      02080 SYSINFO EQU      $-2
02090      ;
02100      ;      Issue SVC & point DE => Auto comm buffer
02110      ;
249A EF     02120      RST     40             ;Read or Write Sector
249B E1     02130      POP     HL            ;Recover command ptr
249C 112025 02140      LD      DE,BUFF+OFFSET ;DE => Auto command buff
249F C8     02150      RET     Z             ;RETurn if successful
02160      ;
02170      ;      I/O Error Handler - Clean up stack & Abort
02180      ;
24A0 6F     02190 IOERR  LD      L,A         ;Set HL = Error #
24A1 2600    02200      LD      H,0
24A3 F6C0    02210      OR      0C0H        ;Short error message
24A5 4F     02220      LD      C,A         ;Xfer to C for @ERROR
24A6      02230 @ERROR      ;Display error
24A6 3E1A    00014      LD      A,26
24A8 EF     00015      RST     40
24A9 C30A24 02240      JP      SAVESP       ;Exit
02250      ;
2500      02260 BUFF  EQU      $-8+1<+8    ;Next page boundary
02270      ;
2400      02280      END      AUTO

```

@@1	0000 @@2	0000 @@3	0000
@@4	0000 @CMNDR	0019 @MOD2	0000
@MOD4	FFFF @RDSEC	0031 @WRSEC	0035
ABB	0010 AP	0027 AUTO	2400
BOOTIO	2491 BREAK	0080 BS	0008
BUFF	2500 CFLAG\$	0002 CR	000D
DFLAG\$	0003 DISAUTO	2422 DOAUTO	243C
DRIVE	2495 ETX	0003 EXAUTO	242C
FLAG	0040 GETAUTO	2447 GETDRV	2451
ILDRIVE	246B IOERR	24A0 KFLAG\$	000A
LF	000A NUM	0080 OFFSET	0020
PARSE	240E PARSE2	240F PAR ERR	002C
RDBOOT	2484 RST28	0028 SAVESP	240A
SECTOR	0002 SFLAG\$	0012 SKIPINC	2460
SKSPSCS	2473 STR	0020 SYSINFO	2498
TAB	0009 VFLAG\$	0015 WRBOOT	2480
WRBUFF	246F XFLP	2479 @@ABORT	7FD8
@@ADTSK	806B @@BANK	8583 @@BKSP	8263
@@BREAK	8599 @@CHNIO	7FC3 @@CKBRKC	85E7
@@CKDRV	80BF @@CKEOF	8278 @@CKTSK	8056
@@CLOSE	824E @@CLS	85D1 @@CMNDI	8002
@@CMNDR	8017 @@CTL	7E27 @@DATE	7F99
@@DCSTAT	80FE @@DEBUG	8041 @@DECHEX	8503
@@DIRRD	8470 @@DIRWR	8485 @@DIV16	84EE
@@DIV8	84D9 @@DODIR	80D4 @@DSP	7DEB
@@DSPLY	7E8B @@ERROR	802C @@EXIT	7FED
@@FEXT	83DD @@FLAGS	856D @@FNAME	83F2
@@FSPEC	83C8 @@GATRD	845B @@GATWR	849A
@@GET	7DFF @@GTDCB	841C @@GTDCT	8407
@@GTMOD	8431 @@HDFMT	81A6 @@HEX16	8542
@@HEX8	852D @@HEXDEC	8518 @@HIGH\$	8557
@@INIT	8224 @@KBD	7E63 @@KEY	7DD7
@@KEYIN	7E77 @@KLTSK	80AA @@LOAD	839E
@@LOC	828D @@LOF	82A2 @@LOGGER	7EC2
@@LOGOT	7ED7 @@MSG	7F0E @@MUL16	84C4
@@MUL8	84AF @@OPEN	8239 @@PARAM	7F84
@@PAUSE	7F6F @@PEOF	82B7 @@POSN	82CC
@@PRINT	7F23 @@PRT	7E3B @@PUT	7E13
@@RAMDIR	80E9 @@RDSEC	817C @@RDSSC	8446
@@READ	82E1 @@REMOV	820F @@RENAM	81FA
@@REW	82F6 @@RMTSK	8080 @@RPTSK	8095
@@RREAD	830B @@RSLCT	8167 @@RSTOR	8128
@@RUN	83B3 @@RWRT	8320 @@SEEK	8152
@@SEEKSC	8335 @@SKIP	834A @@SLCT	8113
@@STEPI	813D @@TIME	7FAE @@VDCTL	7F5A
@@VER	835F @@VRSEC	8191 @@WEOF	8374
@@WHERE	7E4F @@WRITE	8389 @@WRSEC	81BB
@@WRSSC	81D0 @@WRTRK	81E5	

2400 is the transfer address

00000 Total errors

NOTES:

NOTES:

Command: BUILD

Library: SYS7/SYS

ISAM # : 33H

```

00100 ;LBBUILD/ASM - BUILD Command
0000 00110 TITLE <BUILD - LS-DOS 6.2>
00120 ;
00130 ;
0050 00140 CPL EQU 80 ;Characters per line
00150 ;
0000 00160 *GET SVCMAC:3 ;SVC Macro equivalents
00010 ;SVC MAC/ASM - LS-DOS Version VI
00020 *LIST OFF
03900 *LIST ON
0000 00170 *GET VALUES:3 ;Misc. equates
03920 ;VALUES/ASM - Version 6
03930 *LIST OFF
04200 *LIST ON
00180 ;
2400 00190 ORG 2400H
00200 ;
00210 ; Was the <BREAK> key hit ?
00220 ;
00230 START
2400 00240 @@CKBRKC ;Break key down?
2400 3E6A 00001 LD A,106
2402 EF 00002 RST 40
2403 2804 00250 JR Z,$+6 ;Go if not
2405 21FFFF 00260 LD HL,-1 ; else abort
2408 C9 00270 RET
00280 ;
00290 ; <BREAK> not hit - execute module
00300 ;
2409 ED731424 00310 LD (SAVE$P+1),SP ;Save SP address
240D CD3C24 00320 CALL BUILD ;Build a file
2410 210000 00330 LD HL,0 ;Set no error
2413 310000 00340 SAVE$P LD SP,$-$ ;P/u original SP addr.
2416 00350 @@CKBRKC
2416 3E6A 00003 LD A,106
2418 EF 00004 RST 40
2419 C9 00360 RET ;Exit with retcode
00370 ;
00380 ; I/O Error Handler
00390 ;
241A 2600 00400 IOERR LD H,0 ;Set HL = Error #
241C 6F 00410 LD L,A ;
241D F6C0 00420 OR C00H ;Short error mess & RET
241F 4F 00430 LD C,A ;Stuff in C
2420 00440 @@ERROR ;Display error
2420 3E1A 00005 LD A,26
2422 EF 00006 RST 40
2423 18EE 00450 JR SAVE$P ;ABORT
00460 ;
00470 ; Internal Error Message Handling
00480 ;
2425 219825 00490 BADIG$ LD HL,BADIG$ ;"Bad Hex digit"
2428 DD 00500 DB 0DDH
2429 218425 00510 ODDIG$ LD HL,ODDIG$ ;"Odd # of hex digits"
242C DD 00520 DB 0DDH
242D 21B225 00530 SPCREQ LD HL,SPCREQ$ ;"Filespec Required"
2430 DD 00540 DB 0DDH
2431 21C525 00550 EXISTS LD HL,EXISTS$ ;"File already exists"
2434 00560 @@LOGOT ;Log error message
00007 IFEQ 00H,1
00008 LD HL,

```

```

00009          ENDIF
2434 3E0C      00010      LD      A,12
2436 EF        00011      RST      40
2437 21FFFF    00570 ABORT  LD      HL,-1      ;Set abort code
243A 18D7      00580      JR       SAVESP      ;Exit
                00590 ;
                00600 ;      BUILD - Build A file
                00610 ;
243C 112126    00620 BUILD  LD      DE,FCB1      ;DE => FCB
243F           00630      @FSPEC      ;Legal Filespec ?
243F 3E4E      00012      LD      A,78
2441 EF        00013      RST      40
2442 20E9      00640      JR       NZ,SPCREQ      ;No - "Filespec req"
                00650 ;
                00660 ;      Legal Filespec - Stuff default /ext of JCL
                00670 ;
2444 E5        00680      PUSH     HL      ;Save cmd buf ptr
2445 210626    00690      LD      HL,JCLEXT      ;HL => "JCL"
2448           00700      @FEXT      ;Fetch extension
2448 3E4F      00014      LD      A,79
244A EF        00015      RST      40
244B E1        00710      POP      HL      ;Rcvr command ptr
                00720 ;
                00730 ;      Pick up parameters if any
                00740 ;
244C 110E26    00750      LD      DE,PRMTBL$      ;DE => Parameter table
244F           00760      @PARAM      ;Get any parameters
244F 3E11      00016      LD      A,17
2451 EF        00017      RST      40
2452 20C6      00770      JR       NZ,IOERR      ;Quit on parm error
                00780 ;
                00790 ;      Position to Extension
                00800 ;
2454 212126    00810      LD      HL,FCB1      ;Point to start of FCB
2457 7E        00820 SLASH? LD      A,(HL)      ;P/u a char
2458 23        00830      INC      HL      ; & bump pointer
2459 FE0D      00840      CP       CR      ;End of line?
245B 281B      00850      JR       Z,NOTKSM      ;Yes - not a KSM file
245D FE2F      00860      CP       '/'      ;Start of EXT?
245F 20F6      00870      JR       NZ,SLASH?      ;Loop if not
                00880 ;
                00890 ;      Is the extension KSM ?
                00900 ;
2461 E5        00910      PUSH     HL      ;HL => Extension
2462 7E        00920      LD      A,(HL)      ;P/u character
2463 23        00930      INC      HL      ;Bump ptr
2464 FE4B      00940      CP       'K'      ;Match K?
2466 2010      00950      JR       NZ,NOTKSM
2468 7E        00960      LD      A,(HL)
2469 23        00970      INC      HL
246A FE53      00980      CP       'S'      ;Match S?
246C 200A      00990      JR       NZ,NOTKSM
246E 7E        01000      LD      A,(HL)
246F FE4D      01010      CP       'M'      ;Match M?
2471 2005      01020      JR       NZ,NOTKSM
                01030 ;
                01040 ;      Extension is /KSM - stuff 0FFH in indicator
                01050 ;
2473 3EFF      01060      LD      A,0FFH
2475 32DE24    01070      LD      (KSM?+1),A      ;Stuff KSM indicator
                01080 ;

```

```

01090 ;      Is the extension /JCL ?
01100 ;
2478 E1 01110 NOTKSM POP HL ;HL => Extension field
2479 7E 01120 LD A,(HL) ;Ck if EXT is JCL
247A 23 01130 INC HL
247B FE4A 01140 CP 'J' ;Match J?
247D 2010 01150 JR NZ,INIT
247F 7E 01160 LD A,(HL)
2480 23 01170 INC HL
2481 FE43 01180 CP 'C' ;Match C?
2483 200A 01190 JR NZ,INIT
2485 7E 01200 LD A,(HL)
2486 FE4C 01210 CP 'L' ;Match L?
2488 2005 01220 JR NZ,INIT
248A 3E4F 01230 LD A,CPL-1 ;Max 79 cpl on JCL
248C 32F524 01240 LD (LINLEN+2),A
01250 ;
01260 ;      Init the file with LRL of 256
01270 ;
248F 210027 01280 INIT LD HL,IOBUF ;HL => I/O buffer
2492 112126 01290 LD DE,FCB1 ;DE => FCB
2495 0600 01300 LD B,0 ;B = LRL = 256
2497 01310 @@INIT ;Init the file
2497 3E3A 00018 LD A,58
2499 EF 00019 RST 40
249A 2025 01320 JR NZ,IOERRA ;Jump on error
01330 ;
01340 ;      Stuff Filespec into Buffer
01350 ;
249C F5 01360 PUSH AF ;Save Carry
249D 11D925 01370 LD DE,FILEBUF ;DE => Filespec
24A0 ED4B2726 01380 LD BC,(FCB1+6) ;B = DEC, C = Drive #
24A4 3A2126 01390 LD A,(FCB1) ;P/u to test device/file
24A7 CD6E25 01400 CALL $FNAME
24AA 2015 01410 JR NZ,IOERRA
24AC F1 01420 POP AF ;F = Status from @INIT
01430 ;
24AD 21E825 01440 LD HL,BMESS1 ;Default "Building : "
24B0 3818 01450 JR C,SETBUF ;Jump if New file
01460 ;
01470 ;      File already exists - Was APPEND specified ?
01480 ;
24B2 010000 01490 APPEND LD BC,$-$ ;P/u APPEND parameter
24B5 112126 01500 LD DE,FCB1 ;DE => FCB
24B8 0C 01510 INC C ;Specified ?
24B9 2809 01520 JR Z,APP1 ;Go if so
24BB 01530 @@CLOSE ;Close to reset open bit
24BB 3E3C 00020 LD A,60
24BD EF 00021 RST 40
24BE CA3124 01540 JP Z,EXISTS ;Quit with "file exists..."
24C1 C31A24 01550 IOERRA JP IOERR ; or if error on close
01560 ;
01570 ;      Position to end of file for append
01580 ;
24C4 01590 APP1 @@PEOF ;Position to EOF
24C4 3E41 00022 LD A,65
24C6 EF 00023 RST 40
24C7 21F325 01600 LD HL,BMESS2 ;"Appending *KI to "
01610 ;
01620 ;      Display Building/Appending Message
01630 ;

```

The Source	LIBRARY Files	BUILD - LS-DOS 6.2	Page 00004
24CA CD5425	01640 SETBUF	CALL DSPLY	;Display message
24CD 21D925	01650	LD HL,FILEBUF	;Filename buffer
24D0 CD5425	01660	CALL DSPLY	
24D3 0E0D	01670	LD C,CR	;End line
24D5	01680	@@DSP	
24D5 3E02	00024	LD A,2	
24D7 EF	00025	RST 40	
24D8 20E7	01690	JR NZ,IOERRA	
24DA 210028	01700	LD HL,BUFFER	;HL => Input buffer
	01710 ;		
	01720 ;	Is this a KSM File ?	
	01730 ;		
24DD 3E00	01740 KSM?	LD A,\$-\$;Not zero if KSM
24DF B7	01750	OR A	
24E0 2811	01760	JR Z,LINLEN	;Z - not a KSM
	01770 ;		
	01780 ;	KSM loop - p/u current letter & increment	
	01790 ;		
24E2 E5	01800 KSM1	PUSH HL	;Save text pointer
24E3 210926	01810	LD HL,LETBUF	; "A=> "
24E6 34	01820	INC (HL)	;Increment letter
24E7 7E	01830	LD A,(HL)	;P/u letter
	01840 ;		
	01850 ;	Finished with all the KSM keys ?	
	01860 ;		
24E8 FE5B	01870	CP 'Z'+1	;Go past Z?
24EA 2003	01880	JR NZ,DISSTR	;No - display string
24EC E1	01890	POP HL	;Recover text ptr
24ED 183C	01900	JR GOTEND	;Finished
	01910 ;		
	01920 ;	Display letter & "=> "	
	01930 ;		
24EF CD5425	01940 DISSTR	CALL DSPLY	;Display string
24F2 E1	01950	POP HL	;Recover text ptr
	01960 ;		
	01970 ;	Input line with either 255 or 79 characters	
	01980 ;		
24F3 0100FF	01990 LINLEN	LD BC,255<8	;255 or 79 (JCL)
24F6	02000	@@KEYIN	;Input line
24F6 3E09	00026	LD A,9	
24F8 EF	00027	RST 40	
24F9 382C	02010	JR C,GOTBRK	;Exit on <BREAK>
24FB 2025	02020	JR NZ,TSTEOF	;Ck if EOF key used
	02030 ;		
	02040 ;	Got a line of input, check if HEX parameter	
	02050 ;		
24FD 110000	02060 HPARM	LD DE,\$-\$;P/u HEX parameter
2500 1C	02070	INC E	;Specified ?
2501 2018	02080	JR NZ,NOTHEX	;No - ASCII input
	02090 ;		
	02100 ;	HEX parm was entered - convert input to hex	
	02110 ;		
2503 54	02120	LD D,H	;Point DE => Input
2504 5D	02130	LD E,L	
	02140 ;		
2505 CD5B25	02150 HP1	CALL CVRTHDX	;Convert char @ DE
2508 05	02160	DEC B	;Decrement count
2509 CA2924	02170	JP Z,ODDIGS	;Done ? - odd # of digits
	02180 ;		
	02190 ;	Stuff first digit into high order nibble	
	02200 ;		

```

250C 07      02210      RLCA              ;Shift to hi-order nybble
250D 07      02220      RLCA
250E 07      02230      RLCA
250F 07      02240      RLCA
2510 4F      02250      LD      C,A          ;Save in C
                02260 ;
                02270 ;      P/u low-order digit & OR with high order
                02280 ;
2511 CD5B25  02290      CALL    CVRTHX      ;Convert char @ DE
2514 B1      02300      OR      C          ;OR with hi-order nibble
2515 77      02310      LD      (HL),A      ;Stuff in buffer
                02320 ;
                02330 ;      Increment converted input ptr & count down
                02340 ;
2516 23      02350      INC      HL          ;Bump conv input ptr
2517 10EC    02360      DJNZ    HP1         ;B hex digits
2519 18C2    02370      JR      KSM?        ;Done conv, back to loop
                02380 ;
                02390 ;      ASCII input, point HL to next free location
                02400 ;
251B 48      02410 NOTHEX LD      C,B      ;Advance memory buffer
251C 0600    02420      LD      B,0        ;To end of this line
251E 09      02430      ADD     HL,BC      ;HL => End of line
251F 23      02440      INC     HL          ;Bump to 1st free posn
2520 18BB    02450      JR      KSM?        ;Loop for input
                02460 ;
                02470 ;      EOF or BREAK hit - Test Number of chars entered
                02480 ;      If not at line start, convert to CR and continue
                02490 ;
2522 FE1C    02500 TSTEOF CP      1CH      ;EOF?
2524 C21A24  02510      JP      NZ,IOERR    ;Real error if not
2527 78      02520 GOTBRK LD      A,B      ;Any characters ?
2528 B7      02530      OR      A
2529 20D2    02540      JR      NZ,HPARM    ;Yes - continue Build
                02550 ;
                02560 ;      Save input pointer address
                02570 ;
252B 223225  02580 GOTEND LD      (ENDTXT+1),HL ;Save buffer posn
252E 210028  02590      LD      HL,BUFFER  ;HL => Start of input
2531 110000  02600 ENDTXT LD      DE,$-$   ;DE => Last used address
                02610 ;
                02620 ;      Is there any more text to write out ?
                02630 ;
2534 EB      02640      EX      DE,HL      ;Swap for math
2535 AF      02650      XOR     A          ;Clear carry
2536 ED52    02660      SBC     HL,DE      ;Any more text to write ?
2538 280B    02670      JR      Z,ATEND    ;No - don't write any
                02680 ;
                02690 ;      Write a byte to the file
                02700 ;
253A EB      02710      EX      DE,HL      ;HL => Byte to output
253B 112126  02720      LD      DE,FCB1    ;DE => FCB
253E 4E      02730      LD      C,(HL)     ;P/u a char
253F 23      02740      INC     HL          ;Bump ptr
2540        02750      @@PUT              ;Output char
2540 3E04    00028      LD      A,4
2542 EF      00029      RST     40
2543 28EC    02760      JR      Z,ENDTXT    ;Loop for more if i/o OK
                02770 ;
                02780 ;      Done writing - either end of text or error
                02790 ;

```

```

2545 F5      02800 ATEND  PUSH  AF          ;Save error code if any
            02810 ;
            02820 ;          Close the file
            02830 ;
2546 112126  02840      LD      DE,FCB1      ;DE => FCB
2549         02850      @@CLOSE      ;Close the file
2549 3E3C     00030      LD      A,60
254B EF      00031      RST      40
254C C21A24  02860      JP      NZ,IOERR      ;Can't - I/O error
            02870 ;
            02880 ;          Make sure @PUT didn't RETurn any I/O error
            02890 ;
254F F1      02900      POP      AF          ;Recover error code
2550 C8      02910      RET      Z          ;Ok - Return
2551 C31A24  02920      JP      IOERR      ;NZ - I/O Error
            02930 ;
            02940 ;          DSPLY - Display Line & HL
            02950 ;
            02960 ;
2554         02970 DSPLY  @@DSPLY      ;Display line
            00032      IFEQ  00H,1
            00033      LD      HL,
            00034      ENDIF
2554 3E0A     00035      LD      A,10
2556 EF      00036      RST      40
2557 C8      02980      RET      Z          ;Back if good
2558 C31A24  02990      JP      IOERR      ; else abort
            03000 ;
            03010 ;          CVRTHEx - Convert character at DE to hex
            03020 ;
255B 1A      03030 CVRTHEx LD      A,(DE)      ;Get a char
255C 13      03040      INC      DE          ;Bump ptr
255D D630    03050      SUB      '0'        ;Convert to binary
255F 380A    03060      JR      C,BADIGSA    ;Can't be < '0'
2561 FE0A    03070      CP      10         ;Numeric character ?
2563 D8      03080      RET      C          ;Yes - return
2564 CBAF    03090      RES      5,A        ;No - convert to U/C
2566 D607    03100      SUB      7          ;Adjust A-F to be 10-15
2568 FE10    03110      CP      16         ;Legal hex digit ?
256A D8      03120      RET      C          ;Yes - return
256B C32524  03130 BADIGSA JP      BADIGS    ;No - Bad Hex Digit
            03140 ;
            03150 ;          Routine to pick up device/file name
            03160 ;
256E CB7F    03170 $FNAME BIT      7,A        ;Test device/file
2570 2804    03180      JR      Z,FNAME1    ;Go if device
2572         03190      @@FNAME
2572 3E50     00037      LD      A,80
2574 EF      00038      RST      40
2575 C9      03200      RET
2576 3E2A     03210 FNAME1 LD      A,'*'      ;Stuff device indicator
2578 12      03220      LD      (DE),A
2579 13      03230      INC      DE
257A 79      03240      LD      A,C        ;Stuff 1st character
257B 12      03250      LD      (DE),A
257C 13      03260      INC      DE
257D 78      03270      LD      A,B        ;Stuff 2nd character
257E 12      03280      LD      (DE),A
257F 13      03290      INC      DE
2580 3E03     03300      LD      A,3        ;Stuff ETX
2582 12      03310      LD      (DE),A

```



```

2583 C9      03320      RET
              03330 ;
              03340 ;      ERROR Messages
              03350 ;
2584 4F      03360 ODDIGS$ DB      'Odd # of hex digits',CR
64 64 20 23 20 6F 66 20
68 65 78 20 64 69 67 69
74 73 0D
2598 42      03370 BADIGS$ DB      'Bad hex digit encountered',CR
61 64 20 68 65 78 20 64
69 67 69 74 20 65 6E 63
6F 75 6E 74 65 72 65 64
0D
25B2 46      03380 SPCREQ$ DB      'File spec required',CR
69 6C 65 20 73 70 65 63
20 72 65 71 75 69 72 65
64 0D
25C5 46      03390 EXISTS$ DB      'File already exists',CR
69 6C 65 20 61 6C 72 65
61 64 79 20 65 78 69 73
74 73 0D
              03400 ;
000F      03410 FILEBUF DS      15
25E8 42      03420 BMESS1 DB      'Building: ',ETX
75 69 6C 64 69 6E 67 3A
20 03
25F3 41      03430 BMESS2 DB      'Appending: *KI to ',ETX
70 70 65 6E 64 69 6E 67
3A 20 2A 4B 49 20 74 6F
20 03
              03440 ;
2606 4A      03450 JCLEXT DB      'JCL'
43 4C
              03460 ;
2609 40      03470 LETBUF DB      'A'-1,'=> ',ETX
3D 3E 20 03
              03480 ;
              03490 ;      PARAMETER TABLE
              03500 ;
260E 80      03510 PRMTBL$ DB      80H      ;6.x Parameter table
              03520 ;
              03530 ;      HEX (H) parameter - Flag input only
              03540 ;
260F 53      03550 DB      FLAG!ABB!3
2610 48      03560 DB      'HEX'
45 58
2613 00      03570 DB      0
2614 FE24    03580 DW      HPARM+1
              03590 ;
              03600 ;      APPEND (A) parameter - Flag input only
              03610 ;
2616 56      03620 DB      FLAG!ABB!6
2617 41      03630 DB      'APPEND'
50 50 45 4E 44
261D 00      03640 DB      0
261E B324    03650 DW      APPEND+1
              03660 ;
2620 00      03670 DB      0
              03680 ;
              03690 ;      Buffer Area
              03700 ;

```

The Source

LIBRARY Files

BUILD - LS-DOS 6.2

Page 00008

2621 00	03710 FCB1	DB	0
001F	03720	DS	31
2700	03730	ORG	\$<-8+1<8
0100	03740 IOBUF	DS	256
0100	03750 BUFFER	DS	256
	03760 ;		
2400	03770	END	START

\$FNAME	256E @@1	0000 @@2	0000
@@3	0000 @@4	0000 @MOD2	0000
@MOD4	FFFF ABB	0010 ABORT	2437
AP	0027 APP1	24C4 APPEND	24B2
ATEND	2545 BADIGS	2425 BADIGS\$	2598
BADIGSA	256B BMESS1	25E8 BMESS2	25F3
BREAK	0080 BS	0008 BUFFER	2800
BUILD	243C CFLAG\$	0002 CPL	0050
CR	000D CVRTH\$	255B DFLAG\$	0003
DISSTR	24EF DSPLY	2554 ENDTXT	2531
ETX	0003 EXISTS	2431 EXISTS\$	25C5
FCB1	2621 FILEBUF	25D9 FLAG	0040
FNAME1	2576 GOTBRK	2527 GOTEND	252B
HP1	2505 HPARM	24FD INIT	248F
IOBUF	2700 IOERR	241A IOERRA	24C1
JCLEXT	2606 KFLAG\$	000A KSM1	24E2
KSM?	24DD LETBUF	2609 LF	000A
LINLEN	24F3 NOTHEX	251B NOTKSM	2478
NUM	0080 ODDIGS	2429 ODDIGS\$	2584
PAR_ERR	002C PRMTBL\$	260E SAVESP	2413
SETBUF	24CA SFLAG\$	0012 SLASH?	2457
SPCREQ	242D SPCREQ\$	25B2 START	2400
STR	0020 TAB	0009 TSTEOF	2522
VFLAG\$	0015 @ABORT	8B5C @ADTSK	8BEF
@@BANK	9107 @BKSP	8DE7 @BREAK	911D
@@CHNIO	8B47 @CKBRKC	916B @CKDRV	8C43
@@CKEOF	8DFC @CKTSK	8BDA @CLOSE	8DD2
@@CLS	9155 @CMNDI	8B86 @CMNDR	8B9B
@@CTL	89AB @DATE	8B1D @DCSTAT	8C82
@@DEBUG	8BC5 @DECHEX	9087 @DIRRD	8FF4
@@DIRWR	9009 @DIV16	9072 @DIV8	905D
@@DODIR	8C58 @DSP	896F @DSPLY	8A0F
@@ERROR	8BB0 @EXIT	8B71 @FEXT	8F61
@@FLAGS	90F1 @FNAME	8F76 @FSPEC	8F4C
@@GATRD	8FDF @GATWR	901E @GET	8983
@@GTDCB	8FA0 @GTDCT	8F8B @GTMOD	8FB5
@@HDFMT	8D2A @HEX16	90C6 @HEX8	90B1
@@HEXDEC	909C @HIGH\$	90DB @INIT	8DA8
@@KBD	89E7 @KEY	895B @KEYIN	89FB
@@KLTSK	8C2E @LOAD	8F22 @LOC	8E11
@@LOF	8E26 @LOGGER	8A46 @LOGOT	8A5B
@@MSG	8A92 @MUL16	9048 @MUL8	9033
@@OPEN	8DBD @PARAM	8B08 @PAUSE	8AF3
@@PEOF	8E3B @POSN	8E50 @PRINT	8AA7
@@PRT	89BF @PUT	8997 @RAMDIR	8C6D
@@RDSEC	8D00 @RDSSC	8FCA @READ	8E65
@@REMOV	8D93 @RENAM	8D7E @REW	8E7A
@@RMTSK	8C04 @RPTSK	8C19 @RREAD	8E8F
@@RSLCT	8CEB @RSTOR	8CAC @RUN	8F37
@@RWRIT	8EA4 @SEEK	8CD6 @SEEKSC	8EB9
@@SKIP	8ECE @SLCT	8C97 @STEPI	8CC1
@@TIME	8B32 @VDCTL	8ADE @VER	8EE3
@@VRSEC	8D15 @WEOF	8EF8 @WHERE	89D3
@@WRITE	8F0D @WRSEC	8D3F @WRSSC	8D54
@@WRTRK	8D69		

2400 is the transfer address

00000 Total errors

: NOTES:

NOTES:

Command: COPY, APPEND

Library: SYS6/SYS

ISAM # : 32H, 31H

```
00100 ;LBCOPY/ASM - Copy/Append Commands
0000 00110 TITLE <COPY/APPEND - LS-DOS 6.2>
00120 ;
0000 00130 *GET SVCMAC:3
00010 ;SVC MAC/ASM - LS-DOS Version VI
00020 *LIST OFF
03900 *LIST ON
0000 00140 *GET VALUES:3
03920 ;VALUES/ASM - Version 6
03930 *LIST OFF
04200 *LIST ON
00150 ;
001E 00160 EL EQU 30 ;Erase Line
001D 00170 BL EQU 29 ;Beginning of Line
003A 00180 @INIT EQU 58 ;@INIT SVC #
003B 00190 @OPEN EQU 59 ;@OPEN SVC #
0080 00200 BREAK EQU 80H ;BREAK character
00210 ;
2400 00220 ORG 2400H
00230 ;
2400 00240 *GET LBCOPYA:3
04210 ;LBCOPYA/ASM - Copy/Append commands
2400 04220 SUBTTL '<LBCOPYA - APPEND Mainline>'
```

LBCOPYA - APPEND Mainline

```

2400      04230      PAGE
          04240 ;
          04250 ;      Jump to COPY Entry Point
          04260 ;
2400 C3F524 04270 COPY JP      COPYST      ;Go to COPY
          04280 ;
          04290 ;      APPEND Entry Point - Was the <BREAK> hit ?
          04300 ;
          04310 APPEND
2403      04320 @CKBRKC      ;Check for break
2403 3E6A 04330 LD      A,106
2405 EF 04340 RST      40
2406 2804 04350 JR      Z,APPENDA      ;Continue if not
2408 21FFFF 04360 LD      HL,-1      ; else abort
240B C9 04370 RET
          04380 ;
          04390 ;      <BREAK> not hit - Execute Module
          04400 ;
          04410 APPENDA
240C ED734924 04420 LD      (SAVEBP+1),SP      ;Save SP address
2410 CD4F24 04430 CALL    APCODE      ;Call Append code
2413 210000 04440 LD      HL,0      ;Good exit
2416 1830 04450 JR      SAVESP
          04460 ;
          04470 ;      I/O Error Display & Abort Routine
          04480 ;
2418 F5 04490 IOERR PUSH    AF      ;Save error code
2419 CD1629 04500 CALL    PMTSYS      ;Prompt SYSTEM Disk
241C F1 04510 POP      AF      ;Rcvr error code
241D F6C0 04520 OR      C0H      ;Set for abbrev error msg
241F 4F 04530 LD      C,A      ;Save Error # in C
2420 04540 @ERROR      ;Display & abort
2420 3E1A 04550 LD      A,26
2422 EF 04560 RST      40
2423 1823 04570 JR      SAVESP      ;Good bye
          04580 ;
          04590 ;      Load HL with error message string to display
          04600 ;
2425 21C92A 04610 SAMERR LD      HL,SAMERR$      ;"Source & Dest same"
2428 DD 04620 DB      0DDH
2429 218F2A 04630 SPCREQ LD      HL,SPCREQ$      ;"File spec required"
242C DD 04640 DB      0DDH
242D 21A22A 04650 NOINDO LD      HL,NOINDO$      ;"Invalid during <DO>"
2430 DD 04660 DB      0DDH
2431 21492A 04670 DIFLRL LD      HL,DIFLRL$      ;"Files have diff LRLs"
2434 DD 04680 DB      0DDH
2435 21632A 04690 DSTREQ LD      HL,DSTREQ$      ;"Dest spec Required"
2438 04700 @LOGOT      ;Log error message
          04710 IFEQ      00H,1
          04720 LD      HL,
          04730 ENDF
2438 3E0C 04740 LD      A,12
243A EF 04750 RST      40
          04760 ;
          04770 ;      Attempt to close any OPEN destination file
          04780 ;
          04790 ;
243B 11B32B 04800 LD      DE,FCB2      ;Point to dest FCB
243E 1A 04810 LD      A,(DE)      ;Is the file OPEN?
243F 07 04820 RLCA
2440 3003 04830 JR      NC,ABORT      ;No - abort

```


LBCOPYA - APPEND Mainline

```

2442      04740      @@CLOSE
2442 3E3C      00010      LD      A,60
2444 EF      00011      RST      40
2445 21FFFF      04750 ABORT LD      HL,-1      ;Abort code to HL
                04760 ;
                04770 ;      P/u Stack, clear any <BREAK> & return
                04780 ;
2448 310000      04790 SAVESP LD      SP,$-$      ;P/u stack
244B      04800      @@CKBRKC      ;Clear any <BREAK>
244B 3E6A      00012      LD      A,106
244D EF      00013      RST      40
244E C9      04810      RET      ; and RETURN
                04820 ;
                04830 ;      APCODE - Append spec to spec
                04840 ;
                04850 APCODE
244F AF      04860      XOR      A      ;Turn off CLONE parm
2450 328026      04870      LD      (CPARM+1),A
2453 328126      04880      LD      (CPARM+2),A
2456 327828      04890      LD      (APPFLAG+1),A      ;We're in APPEND not COPY
                04900 ;
2459 CDC228      04910      CALL    DOINIT      ;Set High memory
                04920 ;
                04930 ;      Check if Filespec/Devspec #1 is legal
                04940 ;
245C 11072C      04950      LD      DE,FCB1      ;DE => File #1 FCB
245F      04960      @@FSPEC      ;Check out filespec
245F 3E4E      00014      LD      A,78
2461 EF      00015      RST      40
2462 C22924      04970      JP      NZ,SPCREQ      ;NZ - Filespec Required
                04980 ;
                04990 ;      Check if Filespec/Devspec #2 is legal
                05000 ;
2465 11B32B      05010      LD      DE,FCB2      ;DE => File #2 FCB
2468      05020      @@FSPEC      ;Check if legal
2468 3E4E      00016      LD      A,78
246A EF      00017      RST      40
246B C43027      05030      CALL    NZ,CVRTUC      ;Convert line to U/C
                05040 ;
                05050 ;      Is the second FCB a device ?
                05060 ;
246E 3AB32B      05070 APND1 LD      A,(FCB2)      ;P/u byte 0 of FCB2
2471 FE2A      05080      CP      '*'      ;Is this a devspec ?
2473 CA2924      05090      JP      Z,SPCREQ      ;Z - Filespec required
                05100 ;
                05110 ;      Parse any parameters entered
                05120 ;
2476 11812B      05130      LD      DE,APPTBL      ;DE => Parameter Table
2479      05140      @@PARAM      ;Check out parameters
2479 3E11      00018      LD      A,17
247B EF      00019      RST      40
247C C21824      05150      JP      NZ,IOERR      ;NZ - Parameter Error
                05160 ;
247F CD5A27      05170      CALL    PRSPC      ;P/u FCB ptr in DE
                05180 ;
                05190 ;      Open Filespec #2 with LRL of 256
                05200 ;
2482 CD252A      05210      CALL    OPENSF2      ;Open Filespec #2
2485 CD6128      05220      CALL    PUTDEST      ;Xfer Dest filespec

```

LBCOPYA - APPEND Mainline

```

2488 CD9628 05230 CALL GETLRL ;Get LRL from DIR entry
                05240 ;
248B 32A424 05250 LD (LRL2+1),A ;Set dir LRL into parm
248E 327D26 05260 LD (GEOf1+1),A ;Also stuff for later
                05270 ;
                05280 ; Open Filespec #1 with LRL of 256
                05290 ;
2491 CD1B2A 05300 CALL OPENSRC ;Open Filespec #1
2494 CD5B28 05310 CALL PUTSOUR ;Xfer source filespec
                05320 ;
                05330 ; Is the Source a Device ?
                05340 ;
2497 EB 05350 EX DE,HL
2498 CB7E 05360 BIT 7,(HL) ;P/u FCB+0 of source
249A EB 05370 EX DE,HL ;Device ?
249B CC7228 05380 CALL Z,CPYFILE ;Display "Appending ..."
249E 2819 05390 JR Z,APND2 ;Yes - don't check LRLs
                05400 ;
                05410 ; File Source - Check if LRLs are different
                05420 ;
24A0 CD9628 05430 CALL GETLRL ;P/u LRL of Filespec #1
24A3 0600 05440 LRL2 LD B,$-$ ;P/u LRL of Filespec #2
24A5 A8 05450 XOR B ;Same ?
24A6 C23124 05460 JP NZ,DIFLRL ;No - Different LRLs
24A9 CD7228 05470 CALL CPYFILE ;"Appending : "
                05480 ;
                05490 ; Files have same LRLs, check STRIP parameter
                05500 ;
24AC 110000 05510 SPARM LD DE,$-$ ;P/u strip parameter
24AF 7A 05520 LD A,D ;If STRIP, then must do
24B0 B3 05530 OR E ; byte I/O
24B1 2006 05540 JR NZ,APND2 ;Go if STRIP
                05550 ;
                05560 ; Pick up End of File offset byte from FCB
                05570 ;
24B3 3ABB2B 05580 LD A,(FCB2+8) ;Get eof mark
24B6 B7 05590 OR A ;If full sectors, use
24B7 281F 05600 JR Z,APND3 ;Sector I/O
                05610 ;
                05620 ; EOF not on page boundary - use byte I/O
                05630 ;
24B9 11B32B 05640 APND2 LD DE,FCB2
24BC 05650 @PEOF ;Position to end of file
24BC 3E41 00020 LD A,65
24BE EF 00021 RST 40
                05660 ;
                05670 ; If STRIP, then backspace the dest by 1 byte
                05680 ;
24BF 3AAD24 05690 LD A,(SPARM+1) ;P/u SPARM
24C2 B7 05700 OR A ;Specified ?
24C3 280A 05710 JR Z,APND2A ;No - don't backspace
                05720 ;
                05730 ; SPARM specified - Backspace one byte
                05740 ;
24C5 21BC2B 05750 LD HL,FCB2+9 ;HL => LRL of FCB #2
24C8 46 05760 LD B,(HL) ;P/u current dest LRL
24C9 3601 05770 LD (HL),1 ;Reset LRL=1
24CB 05780 @BKSP ;Backspace 1 byte
24CB 3E3D 00022 LD A,61

```

LBCOPYA - APPEND Mainline

```

24CD EF      00023      RST      40
24CE 70      05790      LD       (HL),B      ;Reset LRL back
                05800 ;
                05810 ;      Replace the I/O buffer in FCB #2
                05820 ;
24CF 21002F  05830 APND2A LD      HL,BUF2      ;HL => New buffer addr
24D2 22B62B  05840      LD      (FCB2+3),HL    ;Stuff in FCB
24D5 C3F426  05850      JP      BYTI00        ;
                05860 ;
                05870 ;      EOF on page boundary, use sector I/O
                05880 ;
24D8 ED4B132C 05890 APND3 LD      BC,(FCB1+12)  ;P/u ERN of source
24DC 78      05900      LD      A,B          ;If source is a null
24DD B1      05910      OR      C            ; file, don't do any
24DE CAA326  05920      JP      Z,GE0F3        ; appending, just close
                05930 ;
                05940 ;      Write Ending Record Number
                05950 ;
24E1 2ABF2B  05960      LD      HL,(FCB2+12)    ;P/u ERN of dest
24E4 E5      05970      PUSH     HL            ;Save it for later
24E5 09      05980      ADD     HL,BC          ;Add the two to find new
24E6 44      05990      LD      B,H          ; ERN & Xfer new ERN to BC
24E7 4D      06000      LD      C,L
24E8 CDB526  06010      CALL     WRERN          ;Write a data sector
24EB E1      06020      POP      HL            ;Recover original ERN
24EC 22BF2B  06030      LD      (FCB2+12),HL    ; & reset FCB to it
24EF        06040      @@PEOF        ;Position to end of file
24EF 3E41    00024      LD      A,65
24F1 EF      00025      RST      40
24F2 C31126  06050      JP      XFER5
24F5        06060      SUBTTL   '<LBCOPYA - COPY Mainline>'

```

LBCOPYA - COPY Mainline

```

24F5      06070      PAGE
          06080 ;
          06090 ;      COPY Entry Point - was <BREAK> hit ?
          06100 ;
          06110 COPYST
24F5      06120      @@CKBRKC      ;Check for break
24F5 3E6A  06026      LD      A,106
24F7 EF    06027      RST      40
24F8 2804  06130      JR      Z,COPYSTA      ;Continue if not
24FA 21FFFF 06140      LD      HL,-1      ; else abort
24FD C9    06150      RET
          06160 ;
          06170 ;      <BREAK> not hit - execute module
          06180 ;
          06190 COPYSTA LD      (SAVE SP+1),SP      ;Save SP address
2502 CD0825 06200      CALL     COPYCD      ;Execute Copy code
2505 C31324 06210      JP      EXIT      ;Go to common exit
          06220 ;
          06230 ;      COPYCD - Copy spec to spec
          06240 ;
          06250 COPYCD
2508 CDC228 06260      CALL     DOINIT      ;Set high mem test byte
          06270 ;
          06280 ;      Check if Source Filespec is legal
          06290 ;
250B 11072C 06300      LD      DE,FCB1      ;DE => Source FCB
250E        06310      @@FSPEC      ;Check out filespec
250E 3E4E    06028      LD      A,78
2510 EF      06029      RST      40
2511 C22924 06320      JP      NZ,SPCREQ      ;NZ - Filespec required
          06330 ;
          06340 ;      Check if Destination Filespec is legal
          06350 ;
2514 11B32B 06360      LD      DE,FCB2      ;DE => Destination FCB
2517        06370      @@FSPEC      ;Check out filespec
2517 3E4E    06030      LD      A,78
2519 EF      06031      RST      40
251A C43027 06380      CALL     NZ,CVRTUC      ;Convert line to U/C
          06390 ;
          06400 ;      Process any parameters entered
          06410 ;
251D 11942B 06420 COPY1 LD      DE,COPYTBL      ;DE => Parameter Table
2520        06430      @@PARAM      ;Check out parameters
2520 3E11    06032      LD      A,17
2522 EF      06033      RST      40
2523 C21824 06440      JP      NZ,IOERR      ;NZ - Parameter Error
          06450 ;
          06460 ;      Test if X parameter was entered
          06470 ;
2526 110000 06480 XPARAM LD      DE,$-$      ;P/u (X) parm - We don't
2529 7A      06490      LD      A,D      ; XFER devices
252A B3      06500      OR      E      ;
252B 200B    06510      JR      NZ,XFER      ;
          06520 ;
          06530 ;      Is the Source or Destination a device ?
          06540 ;
252D CDEA29 06550      CALL     CKDEV      ;Device ?
2530 CAD526 06560      JP      Z,BYTEIO      ;Yes - use byte I/O
          06570 ;
          06580 ;      Pick up Defaults for source and destination

```

LBCOPYA - COPY Mainline

```

06590 ;
2533 CD5A27 06600 CALL PRSPC ;P/u defaults
2536 185B 06610 JR OPNSRC
06620 ;
06630 ; XFER initialization code
06640 ;
2538 06650 XFER @@FLAGS ;Position IY to flags
2538 3E65 00034 LD A,101
253A EF 00035 RST 40
253B FDCB126E 06660 BIT 5,(IY+SFLAG$) ;DO in Effect ?
253F C22D24 06670 JP NZ,NOIND0 ;Yes - abort
06680 ;
06690 ; If the Source or Dest is a Device - abort
06700 ;
2542 CDEA29 06710 CALL CKDEV ;Device ?
2545 CA2924 06720 JP Z,SPCREQ ;Yes - Filespecs required
06730 ;
06740 ; P/u Drivespec of Source Filespec if entered
06750 ;
2548 21072C 06760 LD HL,FCB1 ;HL => FCB #1
254B 0E00 06770 LD C,0 ;Init to drive zero
06780 ;
06790 ; Loop to Pick up Drive # or terminator
06800 ;
254D 7E 06810 XFER1 LD A,(HL) ;Look for drive spec
254E 23 06820 INC HL
254F FE3A 06830 CP ':' ;Colon indicator?
2551 2806 06840 JR Z,XFER2 ;Jump if found
2553 FE20 06850 CP ' ' ;Jump on end
2555 3806 06860 JR C,XFER3
2557 18F4 06870 JR XFER1 ;Loop
06880 ;
06890 ; Colon indicator present - p/u drive #
06900 ;
2559 7E 06910 XFER2 LD A,(HL) ;P/u user drive
255A D630 06920 SUB '0' ;Cvrt to binary
255C 4F 06930 LD C,A ; & stuff in C
06940 ;
06950 ; Save Source drive number
06960 ;
255D 211C29 06970 XFER3 LD HL,XFRDRV+1 ;HL => Drive #
2560 71 06980 LD (HL),C ;Save drive # for later
06990 ;
07000 ; Stuff drive # into Prompt strings
07010 ;
2561 3E30 07020 LD A,'0' ;Cvt drive # to ASCII
2563 81 07030 ADD A,C
2564 3232B 07040 LD (SRC_DR),A ;Source Drive #
2567 326C2B 07050 LD (DEST_DR),A ;Destination Drive #
07060 ;
07070 ; Transfer source FCB to destination FCB
07080 ;
256A 21072C 07090 LD HL,FCB1 ;HL => Source FCB
256D 11B32B 07100 LD DE,FCB2 ;DE => Destination FCB
2570 012000 07110 LD BC,32 ;32 bytes to Xfer
2573 EDB0 07120 LDIR ;Xfer
07130 ;
2575 CDE429 07140 CALL GETSYS2 ;Load SYS2 for OPEN
07150 ;

```

LBCOPYA - COPY Mainline

```

      07160 ;      Flash "Insert Source Disk" Message
      07170 ;
2578 21122B 07180 LD      HL,PMTSRC$      ;Prompt for source
257B CD7729 07190 CALL     FLASH          ; and wait for <ENTER>
      07200 ;
      07210 ;      Read in the GAT of the source disk
      07220 ;
257E 3A1C29 07230 LD      A,(XFRDRV+1)    ;P/u source drive
2581 4F      07240 LD      C,A          ;Stuff in C
2582 CDF629 07250 CALL     RDGAT         ;Read in GAT
2585 C21824 07260 JP      NZ,IOERR       ;Abort on GAT error
      07270 ;
      07280 ;      Xfer Password, Name, & Date to destination
      07290 ;
2588 21CE2D 07300 LD      HL,GAT+0CEH     ;Disk pw, name, date
258B 11272C 07310 LD      DE, SRCSTR      ;DE => Destination
258E 011200 07320 LD      BC,18
2591 EDB0    07330 LDIR                     ;Xfer
      07340 ;
      07350 ;      OPEN the Source File with LRL of 256
      07360 ;
2593 CD1B2A 07370 OPNSRC CALL     OPNSRC      ;Open source file
2596 CD5B28 07380 CALL     PUTSOUR        ;Xfer source filespec
2599 CDA528 07390 CALL     GETCLON        ;Get clone data
      07400 ;
      07410 ;      Pick up Source LRL
      07420 ;
259C 7D      07430 LD      A,L          ;Pt back to LRL of source
259D D610    07440 SUB      16
259F 6F      07450 LD      L,A
25A0 7E      07460 LD      A,(HL)        ;P/u source LRL
      07470 ;
      07480 ;      Save LRL from source FCB or LRL Parameter
      07490 ;
25A1 0100FF 07500 LPARM LD      BC,0FF00H    ;P/u LRL
25A4 04      07510 INC      B
25A5 2001    07520 JR      NZ,USEREGC     ;If parm entered, use it
25A7 4F      07530 LD      C,A
25A8 217D26 07540 USEREGC LD      HL,GEOF1+1 ;HL => stuff LRL here
25AB 71      07550 LD      (HL),C        ;Stuff LRL for close here
      07560 ;
      07570 ;      Ignore this if not COPY (X)
      07580 ;
25AC 3A2725 07590 LD      A,(XPARAM+1)    ;Bypass if not (X)
25AF B7      07600 OR      A
25B0 282D    07610 JR      Z,OPNDST
      07620 ;
      07630 ;      Flash "Insert Destination Disk" message
      07640 ;
25B2 21472B 07650 LD      HL,PMTDST$      ;Prompt destination
25B5 CD7729 07660 CALL     FLASH          ;Flash until loaded
      07670 ;
      07680 ;      Read in GAT of Destination Disk
      07690 ;
25B8 3A1C29 07700 LD      A,(XFRDRV+1)    ;P/u drive
25BB 4F      07710 LD      C,A          ;Read GAT from dest
25BC CDF629 07720 CALL     RDGAT
25BF C21824 07730 JP      NZ,IOERR       ;Jump on GAT read error
      07740 ;

```

LBCOPYA - COPY Mainline

```

07750 ;      Xfer Name, Password & Date to destination
07760 ;
25C2 21CE2D 07770 LD      HL,GAT+0CEH      ;HL => GAT + X'CE'
25C5 11392C 07780 LD      DE,DSTSTR      ;DE => Destination
25C8 011200 07790 LD      BC,18          ;To match up when
25CB D5      07800 PUSH     DE
25CC EDB0    07810 LDIR           ; swapping disks
25CE D1      07820 POP      DE          ;Restore Dest ptr
07830 ;
07840 ;      Check if Source ID = Destination ID
07850 ;
25CF 21272C 07860 LD      HL, SRCSTR      ;Compare source & dest
25D2 0612    07870 LD      B,18          ;CANNOT be same
25D4 CD6C29 07880 CALL     CPRHLE      ;Ck MPW, PackID, Date
25D7 2006    07890 JR      NZ, OPNDST    ;Bypass if different
07900 ;
07910 ;      Display "Source & Dest. Disks Identical"
07920 ;
25D9 CD1629 07930 CALL     PMTSYS      ;Prompt for SYSTEM
25DC C32524 07940 JP      SAMERR      ;Disk packs are identical
07950 ;
07960 ;      OPEN the destination File
07970 ;
25DF 11B32B 07980 OPNDST LD      DE,FCB2      ;DE => FCB #2
25E2 21002E 07990 LD      HL,BUF1      ;HL => I/O buffer #1
25E5 CD172A 08000 CALL     INITDES      ;Init the file
25E8 CD6128 08010 CALL     PUTDEST     ;Xfer Dest filespec
08020 ;
08030 ;      Check if X parm entered
08040 ;
25EB 3A2725 08050 LD      A,(XPARAM+1)    ;If (X), then source &
25EE B7      08060 OR      A          ; dest can be same file
25EF 200D    08070 JR      NZ,XF2      ;Bypass if (X)
08080 ;
08090 ;      Does Source & Dest. have same DEC & drive #
08100 ;
25F1 2A0D2C 08110 LD      HL,(FCB1+6)      ;If SRC & DST have same
25F4 ED5BB92B 08120 LD      DE,(FCB2+6)      ; DEC & drive, they are
25F8 AF      08130 XOR      A          ; identical, abort if so
25F9 ED52    08140 SBC      HL,DE
25FB CA3524 08150 JP      Z,DSTREQ      ;Same - dest spec needed
08160 ;
08170 ;      Write revised ERN for space check
08180 ;
25FE CD7228 08190 XF2    CALL     CPYFILE      ;"Copying : ..."
2601 ED4B132C 08200 LD      BC,(FCB1+12)    ;P/u ESN
2605 CDB526 08210 CALL     WRERN        ;Write a FORMAT sector
08220 ;
08230 ;      Reset Destination ESN to Zero
08240 ;
2608 210000 08250 LD      HL,0          ;Rewind file
260B 22BF2B 08260 LD      (FCB2+12),HL      ;
260E        08270 @@REW          ;Rewind the file
260E 3E44    08036 LD      A,68
2610 EF      08037 RST      40
08280 ;
2611 CD2829 08290 XFER5  CALL     PMTSRC      ;Display "Insert source"
08300 ;
08310 ;      Stuff Correct Buffer Address in Source FCB

```

LBCOPYA - COPY Mainline

```

08320 ;
2614 21002E 08330 LD HL,BUF1 ;Stuff in FCB
2617 220A2C 08340 RDREC1 LD (FCB1+3),HL ;Set buffer addr
08350 ;
08360 ; Read in a Source Sector
08370 ;
261A 11072C 08380 LD DE,FCB1 ;DE => Source FCB
261D 08390 @@READ ;Read a sector
261D 3E43 00038 LD A,67
261F EF 00039 RST 40
2620 280B 08400 JR Z,RDREC2 ;Bypass if no error
08410 ;
08420 ; Some sort of I/O Error - Check it out
08430 ;
2622 FE1C 08440 CP 1CH ;EOF?
2624 2827 08450 JR Z,GOTEOF
2626 FE1D 08460 CP 1DH ;NRN>ERN?
2628 2823 08470 JR Z,GOTEOF
262A C31824 08480 JP IOERR ;Abort
08490 ;
08500 ; Successful READ - is there enough memory ?
08510 ;
262D 24 08520 RDREC2 INC H ;Bump memory pointer
262E 7C 08530 LD A,H ;Go past top?
262F FE00 08540 RDREC3 CP $-$
2631 20E4 08550 JR NZ,RDREC1 ;Loop if not
08560 ;
08570 ; Read in all we could - display "Insert Dest"
08580 ;
2633 CD4A29 08590 CALL PMTDST ;Get destination
08600 ;
08610 ; Stuff Source FCB buffer into Destination FCB
08620 ;
2636 21002E 08630 LD HL,BUF1 ;Set buffer start
2639 22B62B 08640 RDREC4 LD (FCB2+3),HL
08650 ;
08660 ; Loop to WRITE Destination file
08670 ;
263C 11B32B 08680 LD DE,FCB2 ;DE => Destination FCB
263F 08690 @@WRITE ;Write a sector
263F 3E4B 00040 LD A,75
2641 EF 00041 RST 40
2642 C21824 08700 JP NZ,IOERR ;Jump on write error
08710 ;
08720 ; Bump memory ptr & check if finished
08730 ;
2645 24 08740 INC H ;Else bump memory pointer
2646 7C 08750 LD A,H ;At top?
2647 FE00 08760 RDREC5 CP $-$
2649 20EE 08770 JR NZ,RDREC4 ;Loop if not
264B 18C4 08780 JR XFER5 ;Else go back to source
08790 ;
08800 ; Got EOF error from source - Write out EOF
08810 ;
264D CDF228 08820 GOTEOF CALL GEOF5 ;Write any memory left
2650 2A0F2C 08830 LD HL,(FCB1+8) ;P/u EOF & LRL
2653 22BB2B 08840 LD (FCB2+8),HL ;Xfer to FCB2
08850 ;
08860 ; Get @CLOSE module if needed

```


LBCOPYA - COPY Mainline

```

08870 ;
2656 CD1629 08880 CALL PMTSYS ;Prompt SYSTEM if needed
2659 CDE729 08890 CALL GETSYS3 ;Load SYS3 for CLOSE
265C 3A1C29 08900 LD A,(XFRDRV+1) ;P/u drive #
265F B7 08910 OR A ;Is it zero ?
2660 CC4A29 08920 CALL Z,PMTDST ;Get dest if drive 0
08930 ;
08940 ; Close the destination file
08950 ;
2663 ED4BB92B 08960 LD BC,(FCB2+6) ;P/u drive # & DEC
2667 11B32B 08970 LD DE,FCB2 ;DE => Destin file FCB
266A 08980 @@CLOSE ;Close the dest file
266A 3E3C 09042 LD A,60
266C EF 09043 RST 40
266D C21824 08990 JP NZ,IOERR ;Jump on error
09000 ;
09010 ; Get the destination file directory record
09020 ;
2670 09030 @DIRRD ;Get destin dir entry
2670 3E57 09044 LD A,87
2672 EF 09045 RST 40
2673 C21824 09040 JP NZ,IOERR ;I/O error - abort
09050 ;
09060 ; Stuff New LRL into directory entry
09070 ;
2676 C5 09080 PUSH BC ;Save drive & DEC
09090 ;
2677 E5 09100 PUSH HL ;HL => DIR+0 of dest
2678 3E04 09110 LD A,4 ;Posn to LRL byte
267A 85 09120 ADD A,L ;
267B 6F 09130 LD L,A ;HL => DIR+4 (LRL)
267C 3600 09140 GEOF1 LD (HL),$$ ;GEOF1+1 contains LRL
267E E1 09150 POP HL ;Restore HL
09160 ;
09170 ; Pick up the Clone Parameter
09180 ;
267F 11FFFF 09190 CPARM LD DE,-1 ;Default = ON
2682 7A 09200 LD A,D ;Was it changed ?
2683 B3 09210 OR E
2684 2811 09220 JR Z,GEOF2 ;CLONE = N
09230 ;
09240 ; CLONE = Yes, Transfer Attributes & Date
09250 ;
2686 EB 09260 EX DE,HL
2687 214B2C 09270 LD HL,CLONSAV ;HL => Attr, DE => DIR+0
268A 010300 09280 LD BC,3 ;Move in prot/date, etc
268D EDB0 09290 LDIR
09300 ;
09310 ; Transfer Password fields to entry
09320 ;
268F 3E0D 09330 LD A,13 ;Pt to dir pswd fields
2691 83 09340 ADD A,E
2692 5F 09350 LD E,A ;DE => DIR+16
2693 0E04 09360 LD C,4 ;BC = 4 bytes to xfer
2695 EDB0 09370 LDIR
09380 ;
09390 ; Write out Directory entry
09400 ;
2697 C1 09410 GEOF2 POP BC ;Rcvr drive & DEC

```

LBCOPYA - COPY Mainline

```

2698      09420      @DIRWR      ;Write Sector with entry
2698 3E58      00046      LD      A,88
269A EF      00047      RST      40
269B 180C      09430      JR      GEOF4      ;Go to Error check
          09440      ;
          09450      ;      CLOSE the destination file
          09460      ;
269D 21FFFF      09470 GEOF3A LD      HL,-1      ;Abort JCL
26A0 22B026      09480      LD      (RETCOD+1),HL      ; if <BREAK> hit
26A3 11B32B      09490 GEOF3 LD      DE,FCB2      ;DE => Destination FCB
26A6      09500      @CLOSE      ;Close the file
26A6 3E3C      00048      LD      A,60
26A8 EF      00049      RST      40
26A9 C21824      09510 GEOF4 JP      NZ,IOERR      ;I/O Error - Abort
          09520      ;
          09530      ;      Flash "Insert SYSTEM disk" & exit
          09540      ;
26AC CD1629      09550 GOHOME CALL    PMTSYS      ;Prompt SYSTEM if needed
26AF 210000      09560 RETCOD LD      HL,$-$      ;P/u return code 0=good
26B2 C34824      09570      JP      SAVESP      ;Finished
          09580      ;
          09590      ;      WRERN - Write a format sector on FILE #2
          09600      ;
26B5 11B32B      09610 WRERN LD      DE,FCB2      ;DE => File #2 FCB
26B8 78      09620      LD      A,B      ;Don't bother to write
26B9 B1      09630      OR      C      ; a sector if source
26BA C8      09640      RET      Z      ; is empty
          09650      ;
          09660      ;      Position to ERN of File #2
          09670      ;
26BB 0B      09680      DEC      BC      ;Adj for ERN
26BC      09690      @POSN      ;Position to ERN
26BC 3E42      00050      LD      A,66
26BE EF      00051      RST      40
26BF D5      09700      PUSH     DE      ;Save FCB ptr
          09710      ;
          09720      ;      Fill a buffer of X'E5's
          09730      ;
26C0 21002E      09740      LD      HL,BUF1      ;HL => I/O buffer
26C3 11012E      09750      LD      DE,BUF1+1      ;DE => I/O buffer+1
26C6 01FF00      09760      LD      BC,255      ;255+1 bytes to fill
26C9 36E5      09770      LD      (HL),0E5H      ;Format byte = X'E5'
26CB EDB0      09780      LDIR      ;Fill buffer
          09790      ;
          09800      ;      Write ERN of File #2
          09810      ;
26CD D1      09820      POP      DE      ;DE => FCB #2
26CE      09830      @WRITE      ;Write sector
26CE 3E4B      00052      LD      A,75
26D0 EF      00053      RST      40
26D1 C8      09840      RET      Z      ;RETurn if no error
26D2 C31824      09850      JP      IOERR      ;Error - abort
          09860      ;
          09870      ;      BYTEIO - OPEN Source or dest using byte I/O
          09880      ;
26D5 CD1B2A      09890 BYTEIO CALL    OPENSRC      ;OPEN source file
26D8 CD5B28      09900      CALL    PUTSOUR      ;Get source filespec
          09910      ;
          09920      ;      INIT the dest device with LRL from parm

```

LBCOPYA - COPY Mainline

```

00930 ;
26DB 3AA225 00940 LD A,(LPARM+1) ;P/u LRL from Parm
26DE 47 00950 LD B,A ;Open destination
26DF 11B32B 00960 LD DE,FCB2 ;DE => FCB #2
26E2 21002F 00970 LD HL,BUF2 ;Different buffer
26E5 3E3A 00980 LD A,@INIT ;@INIT SVC #
26E7 CD2C2A 00990 CALL GETFILE ;Issue it
26EA CD6128 10000 CALL PUTDEST ;Get dest devspec
26ED CD7228 10010 CALL CPYFILE ;"Copying/Appending : .."
26F0 AF 10020 XOR A ;Reset LRL = 0
26F1 32BC2B 10030 LD (FCB2+9),A ;For sector I/O
10040 ;
10050 ; Turn on cursor
10060 ;
26F4 0E0E 10070 BYTI00 LD C,14 ;Turn cursor on
26F6 CDCA29 10080 CALL DISPB ;Display byte
10090 ;
10100 ; BYTI01 Loop - File - Dev, Dev - File, Dev - Dev
10110 ; Was the <BREAK> key hit ?
10120 ;
10130 BYTI01
26F9 CD1229 10140 CALL CKBRK ;Was the <BREAK> key
26FC C29D26 10150 E_O_F JP NZ,GEOF3A ; hit ???
10160 ;
10170 ; The <BREAK> was not hit - get a character
10180 ;
26FF 11072C 10190 LD DE,FCB1 ;DE => Source FCB
2702 10200 @GET ;Get a byte
2702 3E03 00054 LD A,3
2704 EF 00055 RST 40
2705 280B 10210 JR Z,BYTI04 ;Good - stuff it
10220 ;
10230 ; If Error # = 0, then try @GET again
10240 ;
2707 B7 10250 OR A ;Error # = 0 ?
2708 28EF 10260 JR Z,BYTI01 ;Yes - @GET again
10270 ;
10280 ; Is the Error an "End of File" error ?
10290 ;
270A FE1C 10300 CP 1CH ;EOF?
270C CAA326 10310 JP Z,GEOF3 ;Yes - finished
270F C31824 10320 JP IOERR ;I/O error - abort
10330 ;
10340 ; Was the source character a <BREAK> ?
10350 ;
2712 FE80 10360 BYTI04 CP BREAK ;<BREAK> character ?
2714 2007 10370 JR NZ,BYTI04A ;No - @PUT it
10380 ;
10390 ; Source = <BREAK> --- is the BREAK bit set ?
10400 ;
2716 CD1229 10410 CALL CKBRK ;<BREAK> bit set ?
2719 20E1 10420 JR NZ,E_O_F ;Yes - stop
271B 3E80 10430 LD A,BREAK ;Restore A
10440 ;
10450 ; Output byte to destination
10460 ;
271D 11B32B 10470 BYTI04A LD DE,FCB2 ;DE => Dest. Device/File
2720 4F 10480 LD C,A ;Stuff byte in C for @PUT
2721 10490 @PUT ;Output byte

```

LBCOPYA - COPY Mainline

```

2721 3E04      00056      LD      A,4
2723 EF        00057      RST     40
2724 C21824    10500      JP      NZ,IOERR      ;NZ - I/O Error
                10510 ;
                10520 ;      Echo byte if parameter set
                10530 ;
2727 110000    10540 EPARM   LD      DE,$-$      ;P/u ECHO parm
272A 14        10550      INC     D              ;Specified ?
272B CCA29     10560      CALL    Z,DISPB        ;Echo byte
272E 18C9      10570      JR      BYTI01        ;Go til EOF or BREAK
                10580 ;
2730           00250 *GET    LBCOPYB:3
                10590 ;LBCOPYB/ASM - COPY/APPEND Common Routines
2730           10600      SUBTTL  '<LBCOPYB - COPY/APPEND Common Routines>'

```

LBCOPYB - COPY/APPEND Common Routines

```

2730      10610      PAGE
          10620 ;
          10630 ;      CVRTUC - Transfer partspec & convert to U/C
          10640 ;      HL => Buffer containing characters to parse
          10650 ;      DE => Destination of converted line
          10660 ;
2730 CD3827 10670 CVRTUC CALL CVRT      ;Convert line
2733 3E0D    10680 LD      A,CR      ;Ensure end-of-line
2735 12      10690 LD      (DE),A    ; with a carriage return
2736 C9      10700 RET
          10710 ;
          10720 ;      Position to First non-space character
          10730 ;
2737 23      10740 CVRT0 INC      HL
2738 7E      10750 CVRT LD      A,(HL) ;P/u possible dest char
2739 FE0D    10760 CP      CR      ;Exit on CR
273B C8      10770 RET      Z
273C FE20    10780 CP      ' '      ;Loop on space
273E 28F7    10790 JR      Z,CVRT0
2740 2B      10800 DEC      HL      ;Backup to 1st separator
          10810 ;
          10820 ;      HL => Filespec or Devspec, convert to U/C
          10830 ;
2741 0620    10840 LD      B,32      ;Max 32 chars
2743 7E      10850 COP1 LD      A,(HL) ;Transfer the partial
2744 FE61    10860 CP      'a'      ;Cvrt lc to uc
2746 3806    10870 JR      C,COP2
2748 FE7B    10880 CP      'z'+1
274A 3002    10890 JR      NC,COP2
274C CBAF    10900 RES      5,A
274E 12      10910 COP2 LD      (DE),A ;Filespec until paren
274F FE0D    10920 CP      CR      ;RETurn if terminator
2751 C8      10930 RET      Z
2752 FE28    10940 CP      '('      ;Parameter ?
2754 C8      10950 RET      Z
2755 23      10960 INC      HL      ;Bump ptr
2756 13      10970 INC      DE
2757 10EA    10980 DJNZ    COP1      ;32 characters max
2759 C9      10990 RET
          11000 ;
          11010 ;      PRSPC - Match Source & Dest. specs for defaults
          11020 ;
275A 21072C 11030 PRSPC LD      HL,FCB1 ;HL => Source FCB
275D 11B32B 11040 LD      DE,FCB2 ;DE => Destin FCB
          11050 ;
          11060 ;      Abort if first character is illegal
          11070 ;
2760 1A      11080 LD      A,(DE) ;P/u 1st char of dest
2761 FE0D    11090 CP      CR      ;Is it a C/R ?
2763 CA3524 11100 JP      Z,DSTREQ ;"Destination spec req"
2766 FE30    11110 CP      '0'      ;Numeric ?
2768 3805    11120 JR      C,CHKDEST
276A FE3A    11130 CP      '9'+1
276C DA3524 11140 JP      C,DSTREQ ;Yes - "Dest spec req"
          11150 ;
276F D5      11160 CHKDEST PUSH    DE ;Save Destination FCB ptr
          11170 ;
          11180 ;      If no dest. filename, xfer source filename
          11190 ;
2770 1A      11200 LD      A,(DE) ;P/u a dest char

```

LBCOPYB - COPY/APPEND Common Routines

```

2771 FE41      11210      CP      'A'          ;Filename present ?
2773 DCAA27    11220      CALL     C,PRSPC7      ;No - xfer src filename
2776 062F      11230      LD       B,'/'        ;With an alpha, init to
2778 CD8327    11240      CALL     PRSPC2        ; test for extension
277B 062E      11250      LD       B,'.'        ;Init to test for pswd
277D CD8327    11260      CALL     PRSPC2
2780 D1        11270      POP      DE
2781 C9        11280      RET

                11290 ;
                11300 ;      PRSPC2 - Transfer Field to destination
                11310 ;      DE => Start of Destination Field
                11320 ;      B = Delimiter to Position one byte after
                11330 ;
2782 13        11340 PRSPC1 INC      DE
                11350 ;
                11360 ;      Finished with Field (Delimiter hit) ?
                11370 ;
2783 1A        11380 PRSPC2 LD       A,(DE)      ;Is the next char the
2784 B8        11390      CP       B            ;Separator to look for
2785 280E      11400      JR       Z,PRSPC3
                11410 ;
                11420 ;      Skip over an existing destination field
                11430 ;
2787 FE41      11440      CP       'A'          ;Alphabetic ?
2789 30F7      11450      JR       NC,PRSPC1      ;Yes - skip it
278B FE30      11460      CP       '0'          ;Numeric ?
278D 3808      11470      JR       C,PRSPC4      ;No - use source field
278F FE3A      11480      CP       '9'+1        ;Numeric ?
2791 38EF      11490      JR       C,PRSPC1      ;Yes - skip it
                11500 ;
2793 1802      11510      JR       PRSPC4        ;Use source field
                11520 ;
                11530 ;      Hit the delimiter - Position to next char
                11540 ;
2795 13        11550 PRSPC3 INC      DE          ;Position to next field
2796 C9        11560      RET                  ; and RETurn
                11570 ;
                11580 ;      Scan Source spec & see if it contains field
                11590 ;
2797 E5        11600 PRSPC4 PUSH     HL          ;Save ptr to source
2798 7E        11610 PRSPC5 LD       A,(HL)      ;Grab a source char
2799 23        11620      INC      HL          ;Position to next char
                11630 ;
                11640 ;      Is the character a Terminator ?
                11650 ;
279A FE03      11660      CP       ETX          ;End of line ?
279C 280A      11670      JR       Z,PRSPC6      ;If so, not in source
279E FE0D      11680      CP       CR          ;End of line?
27A0 2806      11690      JR       Z,PRSPC6      ;If so, not in source
                11700 ;
                11710 ;      is the character the field delimiter ?
                11720 ;
27A2 B8        11730      CP       B            ;Delimiter ?
27A3 20F3      11740      JR       NZ,PRSPC5     ;Nope, continue
                11750 ;
                11760 ;      Transfer Source Field to Destination
                11770 ;
27A5 CDB627    11780      CALL     MVFLD1        ;Yes, xfer the SRC field
                11790 ;

```

LBCOPYB - COPY/APPEND Common Routines

```

27A8 E1      11800 PRSPC6 POP      HL          ;Rcvr source ptr
27A9 C9      11810          RET
              11820 ;
              11830 ;      PRSPC7 - Shoehorn source into destination field
              11840 ;
              11850 ;      Check if 1st char in (HL) is alphanumeric
              11860 ;
27AA 7E      11870 PRSPC7 LD        A,(HL)      ;P/u source char
27AB FE00    11880          CP      '0'        ;RET NC if alphanumeric
27AD D8      11890          RET      C         ;C - not alphanumeric
27AE FE3A    11900          CP      '9'+1      ;Numeric (0-9) ?
27B0 3803    11910          JR      C,MVFLD    ;Xfer if it is
27B2 FE41    11920          CP      'A'        ;Not numeric -
27B4 D8      11930          RET      C         ;Must be alphabetic
              11940 ;
              11950 ;      Shoehorn a source field byte into dest FCB
              11960 ;
27B5 23      11970 MVFLD  INC      HL          ;Bump source pointer
27B6 E5      11980 MVFLD1 PUSH     HL
27B7 62      11990          LD      H,D        ;Xfer dest ptr to HL
27B8 6B      12000          LD      L,E
              12010 ;
              12020 ;      P/u current char, & stuff in last char
              12030 ;
27B9 4E      12040 MVFLD2 LD      C,(HL)      ;P/u dest char
27BA 77      12050          LD      (HL),A     ;Stuff in last character
27BB 23      12060          INC      HL        ;Posn to next char
              12070 ;
              12080 ;      Finished with field ?
              12090 ;
27BC 79      12100          LD      A,C        ;Test dest for
27BD FE03    12110          CP      ETX       ;End of line ?
27BF 2804    12120          JR      Z,MVFLD3
              12130 ;
27C1 FE0D    12140          CP      CR        ;End of line ?
27C3 20F4    12150          JR      NZ,MVFLD2  ;Ripple the destination
              12160 ;
              12170 ;      Done stuffing 1 source byte into dest field
              12180 ;
27C5 77      12190 MVFLD3 LD      (HL),A     ;Stuff the terminator
27C6 E1      12200          POP      HL        ;Restore Source FCB
27C7 13      12210          INC      DE        ;Advance to next pos
27C8 18E0    12220          JR      PRSPC7     ;Go get next source byte
              12230 ;
              12240 ;      DOSVC - Get Filespec/Devspec & Issue @OPEN/@INIT
              12250 ;
27CA F5      12260 DOSVC  PUSH     AF         ;Save SVC #
27CB E5      12270          PUSH     HL        ;Buffer ptr
27CC D5      12280          PUSH     DE        ;FCB
              12290 ;
              12300 ;      Transfer Filespec/Devspec into TEMBUF
              12310 ;
27CD 21D32B  12320          LD      HL,TEMBUF
27D0 1A      12330 TLP    LD      A,(DE)      ;P/u byte from FCB
27D1 FE2F    12340          CP      '/'       ;Extension ?
27D3 2008    12350          JR      NZ,STUFCHR ;No - save the char
27D5 13      12360          INC      DE        ;Is the next character
27D6 1A      12370          LD      A,(DE)    ; valid ?
27D7 FE41    12380          CP      'A'

```

LBCOPYB - COPY/APPEND Common Routines

```

27D9 3802      12390      JR      C,STUFCHR      ;No - don't output it
27DB 1B        12400      DEC      DE              ;Back one
27DC 1A        12410      LD      A,(DE)           ;P/u slash
27DD 77        12420      STUFCHR LD      (HL),A      ;Xfer to TEMBUF
27DE 23        12430      INC      HL
27DF 13        12440      INC      DE
27E0 FE0E      12450      CP      CR+1             ;Done ?
27E2 3808      12460      JR      C,DUN
27E4 FE3A      12470      CP      ':'
27E6 2804      12480      JR      Z,DUN
27E8 FE2E      12490      CP      '.'
27EA 20E4      12500      JR      NZ,TLP
                12510 ;
                12520 ;      Found valid terminator - Is this a device ?
                12530 ;
27EC 2B        12540      DUN      DEC      HL              ;Back up to term
27ED D1        12550      POP      DE              ;DE => FCB+0
27EE 1A        12560      LD      A,(DE)           ;Device ?
27EF FE2A      12570      CP      '*'
27F1 2807      12580      JR      Z,DUN2            ;Yes - done
27F3 363A      12590      LD      (HL),':'         ;No - overwrite with ":"
27F5 23        12600      INC      HL              ;Bump
27F6 225828    12610      LD      (DSPEC+1),HL      ;Save drivespec location
27F9 23        12620      INC      HL              ;Bump
27FA 3603      12630      DUN2    LD      (HL),ETX      ;End with X'03'
27FC E1        12640      POP      HL              ;HL => Disk I/O buffer
27FD F1        12650      POP      AF              ;A = SVC #
27FE 323228    12660      LD      (SVCNUM+1),A      ;Save SVC #
                12670 ;
                12680 ;      Issue Supervisory Call
                12690 ;
2801 EF        12700      RST      28H              ;Do it
2802 F5        12710      PUSH     AF              ;Save SVC condition
2803 CD4A28    12720      CALL     GETDRIV          ;P/u drv (in C) if file
2806 2817      12730      JR      Z,DEVICE          ;Z - this is a device
                12740 ;
                12750 ;      This is a file - did we @INIT it ?
                12760 ;
2808 3A3228    12770      LD      A,(SVCNUM+1)      ;P/u SVC #
280B FE3A      12780      CP      @INIT            ;Is it @INIT ?
280D 2010      12790      JR      NZ,DEVICE          ;No - don't worry
                12800 ;
                12810 ;      Was the @INIT successful ?
                12820 ;
280F F1        12830      POP      AF              ;Don't perform
2810 F5        12840      PUSH     AF              ; a @CKDRV if you didn't
2811 2804      12850      JR      Z,DO_CKDR          ; successfully @INIT it.
2813 FE2A      12860      CP      42              ;LRL Open Fault ?
2815 2008      12870      JR      NZ,DEVICE          ;No - Don't @CKDRV
                12880 ;
                12890 ;      @INIT was ok - is this a Write Prot Disk?
                12900 ;
2817           12910      DO_CKDR @@CKDRV            ;Write Protected Disk ?
2817 3E21      00058      LD      A,33
2819 EF        00059      RST      40
281A 3E0F      12920      LD      A,15              ;Init WP disk error code
281C DA1824    12930      JP      C,IOERR            ;Yes - abort
                12940 ;
                12950 ;      Check status on the @INIT we did

```


LBCOPYB - COPY/APPEND Common Routines

```

12960 ;
281F F1      12970 DEVICE POP      AF      ;Recover status
2820 2803    12980 JR          Z,CHKPROT ;Check PROTection status
12990 ;
13000 ;      Ignore Error #42 - "LRL Open Fault"
13010 ;
2822 FE2A    13020 CP          42      ;Ignore this error
2824 C0      13030 RET          NZ      ;NZ - Abort
13040 ;
13050 ;      Check if File has proper Access
13060 ;
2825 DDCB007E 13070 CHKPROT BIT      7,(IX) ;Is this a filespec ?
2829 281D    13080 JR          Z,OKYDOKY ;No - don't check
282B DD7E01  13090 LD          A,(IX+1) ;P/u protection byte
282E E607    13100 AND          7      ;
2830 47      13110 LD          B,A      ;Xfer to B
2831 3E00    13120 SVCNUM LD        A,$-$ ;P/u SVC #
2833 FE3A    13130 CP          @INIT   ;@INIT ?
2835 78      13140 LD          A,B      ;P/u protection level
2836 280C    13150 JR          Z,INIT1  ;Z - Must be < 5
2838 FE06    13160 CP          6      ;Read Access ?
283A 380C    13170 JR          C,OKYDOKY ;Yes - set Z & RETurn
13180 ;
13190 ;      Illegal Access to protected file
13200 ;
283C         13210 ILLACC @@CLOSE      ;Close File
283C 3E3C    00060 LD          A,60
283E EF      00061 RST          40
283F 3E19    13220 LD          A,25     ;File Access Denied
2841 C31824  13230 JP          IOERR    ;Error - Regardless
13240 ;
2844 FE05    13250 INIT1 CP          5   ;Update Access ?
2846 30F4    13260 JR          NC,ILLACC ;No - Illegal Access
13270 ;
2848 AF      13280 OKYDOKY XOR        A   ;Set Z flag
2849 C9      13290 RET              ;And RETurn
13300 ;
13310 ;      GETDRIV - P/u Drive # from FCB & convert to ASCII
13320 ;
284A D5      13330 GETDRIV PUSH      DE   ;Get fcb to
284B DDE1    13340 POP          IX      ; here.
284D 1A      13350 LD          A,(DE)   ;Device ?
284E CB7F    13360 BIT          7,A     ;
2850 C8      13370 RET          Z       ;Device - RETurn
13380 ;
13390 ;      Stuff Drive # into Buffer
13400 ;
2851 DD7E06  13410 LD          A,(IX+6)
2854 4F      13420 LD          C,A      ;Xfer to C
2855 C630    13430 ADD          A,'0'    ;Convert to ASCII
2857 320000  13440 DSPEC LD          ($-$),A
285A C9      13450 RET              ;RETurn w/ condition
13460 ;
13470 ;      PUTSOUR/PUTDEST - Create Src/Dest File/Dev specs
13480 ;
285B E5      13490 PUTSOUR PUSH      HL   ;Save HL
285C 21E32B  13500 LD          HL,FROM  ;Xfer there
285F 1804    13510 JR          XBUFF
2861 E5      13520 PUTDEST PUSH      HL   ;Save HL

```

LBCOPYB - COPY/APPEND Common Routines

```

2862 21F72B 13530 LD HL,T02
          13540 ;
          13550 ; XBUFF - Transfer TEMBUF contents to User Buffer
          13560 ; HL => Destination of Filespec/Devspec
          13570 ;
2865 D5 13580 XBUFF PUSH DE ;Save FCB ptr
2866 11D32B 13590 LD DE,TEMBUF ;DE => Source
2869 EB 13600 EX DE,HL ;Swap for LDIR
286A 010F00 13610 LD BC,15 ;15 bytes Max
286D EDB0 13620 LDIR ;Xfer
286F D1 13630 POP DE ;DE => FCB+0
2870 E1 13640 POP HL ;Recover Buffer ptr
2871 C9 13650 RET ;Done
          13660 ;
          13670 ; CPYFILE - Display Copying/Appending Message
          13680 ;
2872 E5 13690 CPYFILE PUSH HL ;Save registers
2873 D5 13700 PUSH DE
2874 21332A 13710 LD HL,COPYMS ;"Copying : "
2877 3E01 13720 APPFLAG LD A,$-$+1 ;Append or Copy ?
2879 B7 13730 OR A
287A 2003 13740 JR NZ,ITSCOPY
287C 213D2A 13750 LD HL,APPDMS ;"Appending : "
287F CDD129 13760 ITSCOPY CALL DSPLY
2882 21E32B 13770 LD HL,FROM ;Source
2885 CDD129 13780 CALL DSPLY
2888 21F32B 13790 LD HL,TO ;" TO "
288B CDD129 13800 CALL DSPLY
288E 0E0D 13810 LD C,CR ;End line
2890 CDCA29 13820 CALL DISPB ;Display byte
2893 D1 13830 POP DE ;Restore Regs
2894 E1 13840 POP HL
2895 C9 13850 RET
          13860 ;
          13870 ; GETLRL - Get LRL from a Directory Entry
          13880 ; DE => FCB of filespec
          13890 ; IX <= FCB of filespec
          13900 ; A <= LRL of file
          13910 ;
2896 D5 13920 GETLRL PUSH DE ;Xfer fcb to IX
2897 DDE1 13930 POP IX
2899 DD7E07 13940 LD A,(IX+7) ;P/u DEC
289C E6E0 13950 AND 0E0H ;Calculate Record Number
289E C604 13960 ADD A,4 ;Pt to LRL
28A0 6F 13970 LD L,A
28A1 2600 13980 SYSBUF LD H,$-$ ;P/u hi-byte of SBUFF$
28A3 7E 13990 LD A,(HL) ;P/u LRL
28A4 C9 14000 RET
          14010 ;
          14020 ; GETCLON - Recover Cloning info from Source
          14030 ;
28A5 D5 14040 GETCLON PUSH DE
28A6 DDE1 14050 POP IX ;Xfer FCB to IX
28A8 DD7E07 14060 LD A,(IX+7) ;P/u source DEC
28AB E6E0 14070 AND 0E0H ;Pt to start of record
28AD 6F 14080 LD L,A ;Pt to core record
28AE 2600 14090 SYSBUF2 LD H,$-$ ;P/u hi-byte of SBUFF$
28B0 114B2C 14100 LD DE,CLONSAV ;Save clone info
28B3 010300 14110 LD BC,3

```

LBCOPYB - COPY/APPEND Common Routines

```

28B6 EDB0      14120      LDIR
28B8 3E0D      14130      LD      A,13          ;Pt to password fields
28BA 85        14140      ADD     A,L
28BB 6F        14150      LD      L,A
28BC 010400    14160      LD      BC,4          ;Save them also
28BF EDB0      14170      LDIR
28C1 C9        14180      RET
                14190 ;
                14200 ;      DOINIT - Initialization for APPEND or COPY
                14210 ;
28C2          14220 DOINIT EQU      $
                14230 ;
                14240 ;      Calculate high byte of available memory
                14250 ;
28C2 E5        14260      PUSH    HL          ;Save HL
28C3          14270      @@FLAGS          ;IY => System Flags
28C3 3E65      00062      LD      A,101
28C5 EF        00063      RST      40
28C6 210000    14280      LD      HL,0          ;P/u HIGH$
28C9 45        14290      LD      B,L
28CA FDCB024E  14300      BIT      1,(IY+CFLAG$) ;@CMNDR ?
28CE 2801      14310      JR      Z,USEHI      ;No - use HIGH$
28D0 04        14320      INC     B          ;Yes - use LOW$
28D1          14330 USEHI  @@HIGH$
28D1 3E64      00064      LD      A,100
28D3 EF        00065      RST      40
                14340 ;
                14350 ;      Stuff high byte into memory check locations
                14360 ;
28D4 23        14370      INC     HL          ; 256-byte block
28D5 25        14380      DEC     H
28D6 7C        14390      LD      A,H          ;Set up test bytes
28D7 323026    14400      LD      (RDREC3+1),A
28DA 324826    14410      LD      (RDREC5+1),A
28DD 32F728    14420      LD      (GE0F6+1),A
                14430 ;
                14440 ;      Pick up high byte of System Buffer (SBUFF$)
                14450 ;
28E0 010000    14460      LD      BC,0          ;DEC = 0, Drive = 0
28E3          14470      @@DIRRD          ;Read in BOOT/SYS
28E3 3E57      00066      LD      A,87
28E5 EF        00067      RST      40
28E6 7C        14480      LD      A,H          ;P/u high byte of SBUFF$
28E7 32A228    14490      LD      (SYSBUF+1),A ;Stuff away for LRL
28EA 32AF28    14500      LD      (SYSBUF2+1),A ;Stuff away for CLONE
                14510 ;
                14520 ;      Reset <PAUSE> & <ENTER> bits
                14530 ;
28ED CDBE29    14540      CALL    RESKFLG      ;Reset bits 1-2
28F0 E1        14550      POP     HL
28F1 C9        14560      RET
                14570 ;
                14580 ;      GE0F5 - Write out the last sector of destination
                14590 ;
28F2 7C        14600 GE0F5 LD      A,H          ;P/u hi-order pointer
28F3 FE2E      14610      CP      BUF1<-8      ;Same as start?
28F5 C8        14620      RET     Z          ;Ret if finished
                14630 ;
                14640 ;      At the end of buffer area ?

```

LBCOPYB - COPY/APPEND Common Routines

```

14650 ;
28F6 FE00 14660 GEOF6 CP    $-$      ;GEOF6+1 = msb of top
28F8 C8   14670 RET    Z          ;Return if finished
14680 ;
14690 ;      Flash "Insert Dest. Disk" & write remainder
14700 ;
28F9 CD4A29 14710 CALL    PMTDST      ;Prompt destination
28FC 44     14720 LD      B,H        ;Save highest used
28FD 21002E 14730 LD      HL,BUF1     ;Pt to start
14740 ;
14750 ;      Loop to write out remainder of dest. file
14760 ;
2900 22B62B 14770 GEOF7 LD      (FCB2+3),HL ;Set dest buffer address
2903 11B32B 14780 LD      DE,FCB2     ;DE => Dest. FCB
2906        14790 @@WRITE      ;Write a record
2906 3E4B   00068 LD      A,75
2908 EF     00069 RST      40
2909 C21824 14800 JP      NZ,IOERR      ;Jump on write error
14810 ;
14820 ;      Finished writing ?
14830 ;
290C 24     14840 INC      H          ;Bump buff ptr
290D 7C     14850 LD      A,H
290E B8     14860 CP      B          ;Finished?
290F 20EF   14870 JR      NZ,GEOF7   ;Loop if not
2911 C9     14880 RET              ;Return
14890 ;
14900 ;      CKBRK - Check for BREAK
14910 ;      NZ - <BREAK> key was depressed
14920 ;
14930 CKBRK
2912        14940 @@CKBRKC      ;<BREAK> hit ?
2912 3E6A   00070 LD      A,106
2914 EF     00071 RST      40
2915 C9     14950 RET              ;Z - No, NZ - Yes
14960 ;
14970 ;      PMTSYS - Prompt for a system disk
14980 ;
2916 3A2725 14990 PMTSYS LD      A,(XPARAM+1) ;X parameter entered ?
2919 B7     15000 OR      A
291A C8     15010 RET      Z          ;No - need to prompt
15020 ;
15030 ;      If Xfer drive number isn't 0 - don't prompt
15040 ;
291B 3E00   15050 XFRDRV LD      A,$-$      ;XFRDRV+1 contains drv #
291D B7     15060 OR      A          ;Is it zero ?
291E C0     15070 RET      NZ        ;Ret if not SYSTEM drive
15080 ;
15090 ;      Flash "Insert SYSTEM disk" message
15100 ;
291F E5     15110 PUSH    HL          ;Save HL
2920 21F32A 15120 LD      HL,PMTSYS$    ;"Insert SYSTEM disk"
2923 CD7729 15130 CALL    FLASH      ;Flash, & RET if <ENTER>
2926 E1     15140 POP      HL          ;Restore HL
2927 C9     15150 RET
15160 ;
15170 ;      PMTSRC - Prompt for Source Disk
15180 ;
2928 3A2725 15190 PMTSRC LD      A,(XPARAM+1) ;X parameter entered ?

```

LBCOPYB - COPY/APPEND Common Routines

```

292B B7      15200      OR      A
292C C8      15210      RET      Z                ;No - don't display it
              15220 ;
              15230 ;      Flash "Insert Source disk" message
              15240 ;
292D E5      15250      PUSH     HL
292E 21122B  15260      LD       HL,PMTSRC$        ;Init for source
2931 CD7729  15270      CALL     FLASH             ;Dsp msg & await reply
              15280 ;
              15290 ;      Read in the GAT from the Source Disk
              15300 ;
2934 3A1C29  15310      LD       A,(XFRDRV+1)      ;P/u drive
2937 4F      15320      LD       C,A              ;Stuff in C
2938 CDF629  15330      CALL     RDGAT             ;Get GAT
              15340 ;
              15350 ;      Is this the correct source disk ?
              15360 ;
293B 21CE2D  15370      LD       HL,GAT+0CEH       ;HL => Name & Date field
293E 11272C  15380      LD       DE,SRCTR         ;DE => Original Source
2941 0612    15390      LD       B,18             ;Same source disk ?
2943 CD6C29  15400      CALL     CPRHLDE          ;Z - same, NZ - different
2946 E1      15410      POP      HL              ;Restore HL
2947 20DF    15420      JR       NZ,PMTSRC        ;Re-request if not match
2949 C9      15430      RET                     ;Okay
              15440 ;
              15450 ;      PMTDST - Prompt for Destination disk
              15460 ;
294A 3A2725  15470 PMTDST LD       A,(XPARAM+1)    ;X parameter entered ?
294D B7      15480      OR       A
294E C8      15490      RET      Z                ;No - RETURN
              15500 ;
              15510 ;      Flash "Insert Destination disk" message
              15520 ;
294F E5      15530      PUSH     HL
2950 21472B  15540      LD       HL,PMTDST$        ;Pt to msg
2953 CD7729  15550      CALL     FLASH             ;Prompt & await reply
              15560 ;
              15570 ;      Read in GAT from original destination disk
              15580 ;
2956 3A1C29  15590      LD       A,(XFRDRV+1)      ;P/u drive #
2959 4F      15600      LD       C,A              ;Stuff in C
295A CDF629  15610      CALL     RDGAT             ;Get GAT
              15620 ;
              15630 ;      Is this the same destination disk ?
              15640 ;
295D 21CE2D  15650      LD       HL,GAT+0CEH       ;HL => Name & Date field
2960 11392C  15660      LD       DE,DSTSTR        ;DE => Original dest
2963 0612    15670      LD       B,18             ;Same destination disk ?
2965 CD6C29  15680      CALL     CPRHLDE          ;Z - Same, NZ - different
2968 E1      15690      POP      HL              ;Restore HL
2969 20DF    15700      JR       NZ,PMTDST        ;Re-request if wrong disk
296B C9      15710      RET
              15720 ;
              15730 ;      CPRHLDE - Compare string @ HL to string @ DE
              15740 ;      B => Number of characters to compare
              15750 ;      Z - Set if strings match
              15760 ;
296C 1A      15770 CPRHLDE LD       A,(DE)         ;P/u character @ DE
296D BE      15780      CP       (HL)            ;Same ?

```

LBCOPYB - COPY/APPEND Common Routines

```

296E C0      15790      RET      NZ          ;Ret (NZ) if no match
296F 23      15800      INC      HL          ;Bump each
2970 13      15810      INC      DE
2971 10F9    15820      DJNZ     CPRHLDE     ;Check B characters
2973 C9      15830      RET              ;Matched - RETURN Z
          15840 ;
          15850 ;      FLASH & FLASH0 - Flash a message string
          15860 ;      HL => Message string to flash
          15870 ;
2974 CDBE29  15880 FLASH0 CALL      RESKFLG      ;Reset Pause, Enter
          15890 ;
          15900 ;      Pause briefly
          15910 ;
2977 01FD41  15920 FLASH LD        BC,16893      ;Delay count
297A          15930      @@PAUSE
297A 3E10    00072      LD        A,16
297C EF      00073      RST        40
          15940 ;
          15950 ;      Wait for no Enter or Break
          15960 ;
297D FD7E0A  15970      LD        A,(IY+KFLAG$) ;P/u KFLAG$
2980 E605    15980      AND        411          ;Wait until no ENTER!BRK
2982 20F0    15990      JR         NZ,FLASH0     ;Still down, go flash
2984 CDBE29  16000      CALL      RESKFLG      ;Reset
          16010 ;
          16020 ;      Display the message & wait for 1/4 second
          16030 ;
2987 CDD129  16040 FLS1 CALL      DSPLY
298A 010040  16050      LD        BC,4000H          ;Delay 4000 iterations
298D CD9D29  16060      CALL      FLS2          ; & scan for Break,Enter
          16070 ;
          16080 ;      Erase the message & wait
          16090 ;
2990 0E1E    16100      LD        C,EL          ;Erase line
2992 CDCA29  16110      CALL      DISPB
2995 013333  16120      LD        BC,3333H          ;Delay 3333 iterations
2998 CD9D29  16130      CALL      FLS2          ; & scan for Break,Enter
299B 18EA    16140      JR         FLS1          ;Loop if neither
          16150 ;
          16160 ;      FLS2 - Delay BC loops & scan for <ENTER>
          16170 ;
299D CD1229  16180 FLS2 CALL      CKBRK          ;Check for <BREAK>
29A0 C24524  16190      JP         NZ,ABORT
          16200 ;
          16210 ;      Was the <ENTER> pressed ?
          16220 ;
29A3 FDCB0A56 16230      BIT        2,(IY+KFLAG$) ;<ENTER> hit ?
29A7 2006    16240      JR         NZ,FLS4          ;Go on ENTER down
          16250 ;
          16260 ;      Nothing hit - Count down
          16270 ;
29A9 0B      16280      DEC        BC          ;Decrement count
29AA 78      16290      LD        A,B          ;Done ?
29AB B1      16300      OR         C
29AC 20EF    16310      JR         NZ,FLS2          ;No - check again
29AE C9      16320      RET              ;Yes - RETURN
          16330 ;
          16340 ;      <ENTER> hit - POP ret addr & clr type ahead
          16350 ;

```

LBCOPYB - COPY/APPEND Common Routines

```

29AF F1      16360 FLS4   POP      AF          ;Pop return address
29B0         16370 CLRTYPE @@KBD          ;Clear type ahead
29B0 3E08    00074      LD      A,8
29B2 EF      00075      RST      40
29B3 28FB    16380      JR      Z,CLRTYPE    ;Good - get another
29B5 B7      16390      OR      A          ;O.K. ?
29B6 C21824  16400      JP      NZ,IOERR     ;No - I/O Error
                16410 ;
                16420 ;      Erase message line
                16430 ;
29B9 0E1E    16440      LD      C,EL        ;Erase line
29BB CDCA29  16450      CALL    DISPB       ;Fall into RESKFLG
                16460 ;
                16470 ;      RESKFLG - Reset <PAUSE> & <ENTER> bits in KFLAG
                16480 ;
29BE         16490 RESKFLG @@FLAGS        ;IY => Flag Table
29BE 3E65    00076      LD      A,101
29C0 EF      00077      RST      40
29C1 FD7E0A  16500      LD      A,(IY+KFLAG$) ;P/u KFLAG$
29C4 E6F9    16510      AND     0F9H       ;Reset bits 2 & 1
29C6 FD770A  16520      LD      (IY+KFLAG$),A ;Stuff in KFLAG$
29C9 C9      16530      RET              ;RETurn
                16540 ;
                16550 ;      DISPB - Output a byte to the Video
                16560 ;
29CA         16570 DISPB  @@DSP           ;Output byte
29CA 3E02    00078      LD      A,2
29CC EF      00079      RST      40
29CD C8      16580      RET      Z          ;Good - RETurn
29CE C31824  16590      JP      IOERR       ;Bad - I/O error
                16600 ;
                16610 ;      DSPLY - Display line to video
                16620 ;
29D1         16630 DSPLY  @@DSPLY         ;Display
                00080      IFEQ     00H,1
                00081      LD      HL,
                00082      ENDIF
29D1 3E0A    00083      LD      A,10
29D3 EF      00084      RST      40
29D4 C8      16640      RET      Z
29D5 C31824  16650      JP      IOERR       ;Bad - I/O Error
                16660 ;
                16670 ;      STOP - Display Transfer aborted & Abort
                16680 ;
29D8 217D2A  16690      LD      HL,STOP$    ;"Transfer Aborted"
29DB         16700      @@LOGOT          ;Log message
                00085      IFEQ     00H,1
                00086      LD      HL,
                00087      ENDIF
29DB 3E0C    00088      LD      A,12
29DD EF      00089      RST      40
29DE CD1629  16710      CALL    PMTSYS     ;"Insert SYSTEM disk"
29E1 C34524  16720      JP      ABORT      ;Abort
                16730 ;
                16740 ;      GETSYS2 - Bring in SYS2
                16750 ;
29E4 3E84    16760      GETSYS2 LD      A,84H
29E6 EF      16770      RST      28H
                16780 ;

```

LBCOPYB - COPY/APPEND Common Routines

```

16790 ;      GETSYS3 - Bring in SYS3
16800 ;
29E7 3E85 16810 GETSYS3 LD      A,85H
29E9 EF   16820      RST      28H
16830 ;
16840 ;      CKDEV - Is the source or Destination a device ?
16850 ;      Z - Either Source or Destination is a device
16860 ;
29EA 3A072C 16870 CKDEV LD      A,(FCB1)      ;Is source a device?
29ED FE2A   16880      CP      '*'
29EF C8     16890      RET      Z
29F0 3AB32B 16900      LD      A,(FCB2)      ;Is destination a device?
29F3 FE2A   16910      CP      '*'
29F5 C9     16920      RET
16930 ;
16940 ;      RDGAT - Read GAT of Drive C
16950 ;
29F6 D5     16960 RDGAT PUSH    DE            ;Save DE & HL
29F7 E5     16970      PUSH    HL
29F8 FDE5   16980      PUSH    IY            ;Save IY
16990 ;
17000 ;      Set D = Cyl, E = Sector, HL => Buffer
17010 ;
29FA       17020      @@GTDCT                ;Point IY to DCT
29FA 3E51   00090      LD      A,81
29FC EF     00091      RST      40
29FD FD5609 17030      LD      D,(IY+9)        ;P/u dir cyl in D
2A00 1E00   17040      LD      E,0            ;GAT is sector 0
2A02 21002D 17050      LD      HL,GAT          ;HL => GAT I/O Buffer
2A05       17060      @@RDSSC                ;Read Track D, Sector E
2A05 3E55   00092      LD      A,85
2A07 EF     00093      RST      40
17070 ;
17080 ;      Restore Registers
17090 ;
2A08 FDE1   17100      POP      IY            ;Restore IY
2A0A E1     17110      POP      HL            ;Restore HL & DE
2A0B D1     17120      POP      DE
2A0C 3E14   17130      LD      A,14H          ;Else reset to GAT error
2A0E C9     17140      RET                    ;RETurn with condition
17150 ;
17160 ;      OPENDES - OPEN Destination File
17170 ;
2A0F 21002F 17180 OPENDES LD      HL,BUF2      ;HL => Dest I/O Buffer
2A12 11B32B 17190      LD      DE,FCB2        ;DE => Dest FCB
2A15 1811   17200      JR      OPENFIL        ;Open the file
17210 ;
17220 ;      INITDES - INIT the destination file
17230 ;
2A17 3E3A   17240 INITDES LD      A,@INIT      ;A = @INIT SVC Number
2A19 180F   17250      JR      INITFIL        ;Go into OPEN routine
17260 ;
17270 ;      OPENSRC - Open Source File
17280 ;
2A1B 11072C 17290 OPENSRC LD      DE,FCB1      ;DE => Source FCB
17300 ;
17310 ;      Set the File OPEN inhibit flag
17320 ;
2A1E       17330      @@FLAGS                ;IY => System Flag Table

```


LBCOPYB - COPY/APPEND Common Routines

```

2A1E 3E65      00094      LD      A,101
2A20 EF        00095      RST      40
2A21 FDCB12C6  17340      SET      0,(IY+SFLAG$) ;Set file open inhibit
                17350 ;
2A25 21002E    17360 OPENS2 LD      HL,BUF1      ;HL => Source I/O buffer
2A28 3E3B      17370 OPENFIL LD      A,@OPEN    ;A = @OPEN SVC number
2A2A 0600      17380 INITFIL LD      B,0         ;B = LRL = 256
                17390 ;
                17400 ;      OPEN or INIT the File
                17410 ;
2A2C CDCA27    17420 GETFILE CALL    DOSVC      ;OPEN or INIT the file
2A2F C8        17430 RET      Z              ;RETurn with Z set
2A30 C31824    17440 JP      IOERR          ;NZ - I/O Error
                17450 ;
                17460 ;      Error & Informative message strings
                17470 ;
2A33 43        17480 COPYMS DB      'Copying: ',ETX
        6F 70 79 69 6E 67 3A 20
        03
2A3D 41        17490 APPDMS DB      'Appending: ',ETX
        70 70 65 6E 64 69 6E 67
        3A 20 03
                17500 ;
2A49 46        17510 DIFLRL$ DB      'Files have different LRLs',CR
        69 6C 65 73 20 68 61 76
        65 20 64 69 66 66 65 72
        65 6E 74 20 4C 52 4C 73
        0D
2A63 44        17520 DSTREQ$ DB      'Destination spec required',CR
        65 73 74 69 6E 61 74 69
        6F 6E 20 73 70 65 63 20
        72 65 71 75 69 72 65 64
        0D
2A7D 1D        17530 STOP$  DB      BL,'Transfer aborted',CR
        54 72 61 6E 73 66 65 72
        20 61 62 6F 72 74 65 64
        0D
2A8F 46        17540 SPCREQ$ DB      'File spec required',CR
        69 6C 65 20 73 70 65 63
        20 72 65 71 75 69 72 65
        64 0D
2AA2 49        17550 NOINDO$ DB      'Invalid command during <DO> '
        6E 76 61 6C 69 64 20 63
        6F 6D 6D 61 6E 64 20 64
        75 72 69 6E 67 20 3C 44
        4F 3E 20
2ABE 70        17560      DB      'processing',CR
        72 6F 63 65 73 73 69 6E
        67 0D
2AC9 53        17570 SAMERR$ DB      'Source and destination disks'
        6F 75 72 63 65 20 61 6E
        64 20 64 65 73 74 69 6E
        61 74 69 6F 6E 20 64 69
        73 6B 73
2AE5 20        17580      DB      ' are the same',CR
        61 72 65 20 74 68 65 20
        73 61 6D 65 0D
                17590 ;
2AF3 1D        17600 PMTSYS$ DB      BL,EL,' Insert SYSTEM disk <ENTER>'

```

LBCOPYB - COPY/APPEND Common Routines

```

      1E 20 49 6E 73 65 72 74
      20 53 59 53 54 45 4D 20
      64 69 73 6B 20 3C 45 4E
      54 45 52 3E
2B10 1D      17610      DB      BL,ETX
      03
      17620 ;
2B12 1D      17630 PMTSRC$ DB      BL,EL,' Insert SOURCE disk '
      1E 20 49 6E 73 65 72 74
      20 53 4F 55 52 43 45 20
      64 69 73 6B 20
2B28 69      17640      DB      'in drive :'
      6E 20 64 72 69 76 65 20
      3A
2B32 30      17650 SRC DR DB      ' and press <ENTER>'
      20 61 6E 64 20 70 72 65
      73 73 20 3C 45 4E 54 45
      52 3E
2B45 1D      17660      DB      BL,ETX
      03
      17670 ;
2B47 1D      17680 PMTDST$ DB      BL,EL,' Insert DESTINATION disk '
      1E 20 49 6E 73 65 72 74
      20 44 45 53 54 49 4E 41
      54 49 4F 4E 20 64 69 73
      6B 20
2B62 69      17690      DB      'in drive :'
      6E 20 64 72 69 76 65 20
      3A
2B6C 30      17700 DEST DR DB      ' and press <ENTER>'
      20 61 6E 64 20 70 72 65
      73 73 20 3C 45 4E 54 45
      52 3E
2B7F 1D      17710      DB      BL,ETX
      03
      17720 ;
      17730 ;
      17740 ;
      17750 ;
      17760 APPTBL DB      80H      ;Use new @PARAM
      17770 ;
2B82 55      17780      DB      FLAG!ABB!5
2B83 53      17790      DB      'STRIP'
      54 52 49 50
2B88 00      17800      DB      0
2B89 AD24    17810      DW      SPARM+1
      17820 ;
2B8B 54      17830      DB      FLAG!ABB!4
2B8C 45      17840      DB      'ECHO'
      43 48 4F
2B90 00      17850      DB      0
2B91 2827    17860      DW      EPARM+1
2B93 00      17870      DB      0
      17880 ;
      17890 ;
      17900 ;
2B94 80      17910 COPYTBL DB      80H      ;New @PARAM
      17920 ;
2B95 54      17930      DB      FLAG!ABB!4

```

LBCOPYB - COPY/APPEND Common Routines

```

2B96 45      17940      DB      'ECHO'
      43 48 4F
2B9A 00      17950      DB      0
2B9B 2827    17960      DW      EPARM+1
      17970 ;
2B9D 93      17980      DB      NUM!ABB!3
2B9E 4C      17990      DB      'LRL'
      52 4C
2BA1 00      18000      DB      0
2BA2 A225    18010      DW      LPARM+1
      18020 ;
2BA4 55      18030      DB      FLAG!ABB!5
2BA5 43      18040      DB      'CLONE'
      4C 4F 4E 45
2BAA 00      18050      DB      0
2BAB 8026    18060      DW      CPARM+1
      18070 ;
2BAD 41      18080      DB      FLAG!1
2BAE 58      18090      DB      'X'
2BAF 00      18100      DB      0
2BB0 2725    18110      DW      XPARM+1
2BB2 00      18120      DB      0
      18130 ;
      18140 ;
      18150 ;
      I/O & Storage Buffers
2BB3 00      18160 FCB2   DB      0
      18170   DS      31
      18180 TEMBUF  DS      16
      18190 FROM    DS      16
2BF3 20      18200 TO     DB      ' to '
      74 6F 20
      18210 T02      DS      16
2C07 00      18220 FCB1   DB      0
      18230   DS      31
      18240 SRCSTR  DS      18
      18250 DSTSTR  DS      18
      18260 CLONSAV DS      7
      18270 ;
2D00      18280      ORG      $<-8+1<+8
      18290 ;
      18300 GAT      DS      256
      18310 BUF1     DS      256
      18320 BUF2     DS      256
      18330 ;
      00260 ;
3000      00270      SUBTTL  <>
2400      00280      END      COPY
;End of COPY parm table
;Show closed on LOAD

```

@@1	0000 @@2	0000 @@3	0000
@@4	0000 @INIT	003A @MOD2	0000
@MOD4	FFFF @OPEN	003B ABB	0010
ABORT	2445 AP	0027 APCODE	244F
APND1	246E APND2	24B9 APND2A	24CF
APND3	24D8 APPDMS	2A3D APPEND	2403
APPENDA	240C APPFLAG	2877 APPTBL	2B81
BL	001D BREAK	0080 BS	0008
BUF1	2E00 BUF2	2F00 BYTE IO	26D5
BYTIO0	26F4 BYTIO1	26F9 BYTIO4	2712
BYTIO4A	271D CFLAG\$	0002 CHKDEST	276F
CHKPROT	2825 CKBRK	2912 CKDEV	29EA
CLONSAV	2C4B CLRTYPE	29B0 COP1	2743
COP2	274E COPY	2400 COPY1	251D
COPYCD	2508 COPYMS	2A33 COPYST	24F5
COPYSTA	24FE COPYTBL	2B94 CPARAM	267F
CPRHLDE	296C CPYF ILE	2872 CR	000D
CVRT	2738 CVRT0	2737 CVRTUC	2730
DEST DR	2B6C DEVICE	281F DFLAG\$	0003
DIFLRL	2431 DIFLRL\$	2A49 DISPB	29CA
DOINIT	28C2 DOSVC	27CA DO CKDR	2817
DSPEC	2857 DSPLY	29D1 DSTREQ	2435
DSTREQ\$	2A63 DSTSTR	2C39 DUN	27EC
DUN2	27FA EL	001E EPARM	2727
ETX	0003 EXIT	2413 E O F	26FC
FCB1	2C07 FCB2	2BB3 FLAG	0040
FLASH	2977 FLASH0	2974 FLS1	2987
FLS2	299D FLS4	29AF FROM	2BE3
GAT	2D00 GEOF1	267C GEOF2	2697
GEOF3	26A3 GEOF3A	269D GEOF4	26A9
GEOF5	28F2 GEOF6	28F6 GEOF7	2900
GETCLON	28A5 GETDRIV	284A GETFILE	2A2C
GETLRL	2896 GETSYS2	29E4 GETSYS3	29E7
GOHOME	26AC GOTE0F	264D ILLACC	283C
INIT1	2844 INITDES	2A17 INITFIL	2A2A
IOERR	2418 ITSCOPY	287F KFLAG\$	000A
LF	000A LPARM	25A1 LRL2	24A3
MVFLD	27B5 MVFLD1	27B6 MVFLD2	27B9
MVFLD3	27C5 NOINDO	242D NOINDO\$	2AA2
NUM	0080 OKYDOKY	2848 OPENDES	2A0F
OPENFIL	2A28 OPENSRC2	2A25 OPENSRC	2A1B
OPNDST	25DF OPNSRC	2593 PAR ERR	002C
PMTDST	294A PMTDST\$	2B47 PMTSRC	2928
PMTSRC\$	2B12 PMTSYS	2916 PMTSYS\$	2AF3
PRSPC	275A PRSPC1	2782 PRSPC2	2783
PRSPC3	2795 PRSPC4	2797 PRSPC5	2798
PRSPC6	27A8 PRSPC7	27AA PUTDEST	2861
PUTSOUR	285B RDGAT	29F6 RDREC1	2617
RDREC2	262D RDREC3	262F RDREC4	2639
RDREC5	2647 RESKFLG	29BE RETCOD	26AF
SAMERR	2425 SAMERR\$	2AC9 SAVESP	2448
SFLAG\$	0012 SPARM	24AC SPCREQ	2429
SPCREQ\$	2A8F SRCSTR	2C27 SRC DR	2B32
STOP	29D8 STOP\$	2A7D STR	0020
STUFCHR	27DD SVCNUM	2831 SYSBUF	28A1
SYSBUF2	28AE TAB	0009 TEMBUF	2BD3
TLP	27D0 TO	2BF3 T02	2BF7
USEHI	28D1 USEREGC	25A8 VFLAG\$	0015

WRERN	26B5	XBUFF	2865	XF2	25FE
XFER	2538	XFER1	254D	XFER2	2559
XFER3	255D	XFER5	2611	XFRDRV	291B
XPARM	2526	@@ABORT	6D39	@@ADTSK	6DCC
@@BANK	72E4	@@BKSP	6FC4	@@BREAK	72FA
@@CHNIO	6D24	@@CKBRKC	7348	@@CKDRV	6E20
@@CKEOF	6FD9	@@CKTSK	6DB7	@@CLOSE	6FAF
@@CLS	7332	@@CMNDI	6D63	@@CMNDR	6D78
@@CTL	6B88	@@DATE	6CFA	@@DCSTAT	6E5F
@@DEBUG	6DA2	@@DECHEX	7264	@@DIRRD	71D1
@@DIRWR	71E6	@@DIV16	724F	@@DIV8	723A
@@DODIR	6E35	@@DSP	6B4C	@@DSPLY	6BEC
@@ERROR	6D8D	@@EXIT	6D4E	@@FEXT	713E
@@FLAGS	72CE	@@FNAME	7153	@@FSPEC	7129
@@GATRD	71BC	@@GATWR	71FB	@@GET	6B60
@@GTDCT	717D	@@GTDCT	7168	@@GTMOD	7192
@@HDFMT	6F07	@@HEX16	72A3	@@HEX8	728E
@@HEXDEC	7279	@@HIGH\$	72B8	@@INIT	6F85
@@KBD	6BC4	@@KEY	6B38	@@KEYIN	6BD8
@@KLTSK	6E0B	@@LOAD	70FF	@@LOC	6FEE
@@LOF	7003	@@LOGGER	6C23	@@LOGOT	6C38
@@MSG	6C6F	@@MUL16	7225	@@MUL8	7210
@@OPEN	6F9A	@@PARAM	6CE5	@@PAUSE	6CD0
@@PEOF	7018	@@POSN	702D	@@PRINT	6C84
@@PRT	6B9C	@@PUT	6B74	@@RAMDIR	6E4A
@@RDSEC	6EDD	@@RDSSC	71A7	@@READ	7042
@@REMOV	6F70	@@RENAM	6F5B	@@REW	7057
@@RMTSK	6DE1	@@RPTSK	6DF6	@@RREAD	706C
@@RSLCT	6EC8	@@RSTOR	6E89	@@RUN	7114
@@RWRTIT	7081	@@SEEK	6EB3	@@SEEKSC	7096
@@SKIP	70AB	@@SLCT	6E74	@@STEPI	6E9E
@@TIME	6D0F	@@VDCTL	6CBB	@@VER	70C0
@@VRSEC	6EF2	@@WEOF	70D5	@@WHERE	6BB0
@@WRITE	70EA	@@WRSEC	6F1C	@@WRSSC	6F31
@@WRTRK	6F46				

2400 is the transfer address

00000 Total errors

NOTES:

NOTES:

Command: CREATE

Library: SYS7/SYS

ISAM # : 13H


```

00100 ;LBCREATE/ASM - CREATE Command
0000 00110 TITLE <CREATE - LS-DOS 6.2>
00120 ;
0000 00130 *GET SVCMAC:3 ;SVC Macro equivalents
00010 ;SVC MAC/ASM - LS-DOS Version VI
00020 *LIST OFF
0000 00000 *LIST ON
00140 *GET VALUES:3 ;Misc. equates
00020 ;VALUES/ASM - Version 6
00030 *LIST OFF
00040 *LIST ON
00150 ;
2400 00160 ORG 2400H
00170 ;
2400 00180 CREATE EQU $
2400 00190 @@CKBRKC ;Break key down?
2400 3E6A 00001 LD A,106
2402 EF 00002 RST 40
2403 2804 00200 JR Z,BEGINA ;Ok if not
2405 21FFFF 00210 LD HL,-1 ; else abort
2408 C9 00220 RET
00230 ;
2409 ED731E25 00240 BEGINA LD (SAVE SP+1),SP ;Save SP address
240D CD2824 00250 CALL CREATCD ;Do the CREATE code
00260 ;
00270 ; Set ERN & offset in FCB = value at @INIT
00280 ;
00290 ;
2410 210000 00290 ERN LD HL,$-$ ;P/u ERN (probably 0)
2413 22B125 00300 LD (FCB+12),HL ;Stuff into FCB
2416 3E00 00310 OFFSET LD A,$-$ ;P/u offset byte
2418 32AD25 00320 LD (FCB+8),A ;Stuff into FCB
00330 ;
00340 ; CLOSE the file if possible
00350 ;
241B 11A525 00360 LD DE,FCB ;DE => FCB
241E 00370 @@CLOSE ;Close file
241E 3E3C 00003 LD A,60
2420 EF 00004 RST 40
2421 C21425 00380 JP NZ,IOERR ;NZ - I/O Error
00390 ;
00400 ; Exit Routine
00410 ;
2424 210000 00420 LD HL,0 ;Successful - HL = 0
2427 C9 00430 RET ;Return
00440 ;
00450 ; Parse the Filespec given
00460 ;
2428 11A525 00470 CREATCD LD DE,FCB ;Fetch filespec
242B 00480 @@FSPEC
242B 3E4E 00005 LD A,78
242D EF 00006 RST 40
242E C23725 00490 JP NZ,SPCREQ ;Quit on bad name
00500 ;
00510 ; Check out parameter input
00520 ;
2431 118525 00530 LD DE,PRMTBL$ ;Get parms
2434 00540 @@PARAM
2434 3E11 00007 LD A,17
2436 EF 00008 RST 40
2437 C21425 00550 JP NZ,IOERR ;Jump on parm error
00560 ;

```

```

00570 ;      Check If Rec or LRL were specified
00580 ;
243A 3A9925 00590      LD      A,(LRESP)      ;P/u LRL response
243D 47      00600      LD      B,A          ;Save in B
243E 3A9225 00610      LD      A,(RRESP)      ;P/u REC response
2441 B0      00620      OR      B            ;Either specified ?
2442 2807    00630      JR      Z,RPARM        ;No - check # records
00640 ;
00650 ;      If Size parm also was specified - Parameter Error
00660 ;
2444 3A8B25 00670      LD      A,(SRESP)      ;Size can't be used
2447 B7      00680      OR      A            ; with REC or LRL
2448 C21225 00690      JP      NZ,PRMERR       ;Specified ? - Error
00700 ;
00710 ;      Check Record count
00720 ;
244B 010000 00730      LD      BC,$-$          ;P/u # of records
244E 78      00740      LD      A,B
244F B1      00750      OR      C
2450 200D    00760      JR      NZ,LPARM
00770 ;
00780 ;      Zero Records - Use Size instead
00790 ;
2452 210000 00800      LD      HL,$-$          ;P/u size parm
2455 7C      00810      LD      A,H
2456 B5      00820      OR      L
2457 CA1225 00830      JP      Z,PRMERR        ;Err if size not entered
00840 ;
00850 ;      Multiply HL x 4 to get # of sectors
00860 ;
245A 29      00870      ADD     HL,HL          ;X 2
245B 29      00880      ADD     HL,HL          ;X 4
245C 224C24 00890      LD      (RPARM+1),HL   ;Pretend it's rec input
00900 ;
00910 ;      Make sure LRL input is valid
00920 ;
245F 010000 00930      LD      BC,$-$          ;P/u LRL
2462 78      00940      LD      A,B          ;Test for > 256
2463 B7      00950      OR      A            ;If hi-order = 0,
2464 2808    00960      JR      Z,LP1          ;Just use lo-order
2466 3D      00970      DEC     A            ;Test hi-order = 1
2467 C21225 00980      JP      NZ,PRMERR       ;Quit if any other
246A B1      00990      OR      C            ;P/u lo-order
246B C21225 01000      JP      NZ,PRMERR       ;Lo-order must be 0
246E B1      01010      LP1     OR      C            ;Merge lo-order
01020 ;
01030 ;      Open the File with the LRL specified
01040 ;
246F 11A525 01050      LD      DE,FCB          ;Open the file
2472 210026 01060      LD      HL,BUFFER
2475 47      01070      LD      B,A          ;LRL = 256, or user entry
2476        01080      @@INIT
2476 3E3A    00009      LD      A,58
2478 EF      00010      RST     40
2479 C21425 01090      JP      NZ,IOERR        ;Jump on init error
01100 ;
01110 ;      Display "Creating : Filespec" String
01120 ;
247C 117625 01130      LD      DE,FILESP      ;DE => Filespec
247F ED4BAB25 01140      LD      BC,(FCB+6)    ;P/u drive #, DEC
2483 3AA525 01150      LD      A,(FCB)        ;P/u to test device/file

```

```

2486 CD2125 01160 CALL $FNAME
2489 C21425 01170 JP NZ,IOERR ;Leave on error
248C 216C25 01180 LD HL,CREATE$ ;"Creating : "
248F 01190 @@DSPLY ;Display it
00011 IFEQ 00H,1
00012 LD HL,
00013 ENDIF
248F 3E0A 00014 LD A,10
2491 EF 00015 RST 40
2492 C21425 01200 JP NZ,IOERR ;Quit on dsply error
2495 0E0D 01210 LD C,CR ;End line
2497 01220 @@DSP
2497 3E02 00016 LD A,2
2499 EF 00017 RST 40
249A C21425 01230 JP NZ,IOERR
01240 ;
01250 ; Save ERN & offset from FCB for later
01260 ;
249D 2AB125 01270 LD HL,(FCB+12) ;P/u ERN
24A0 221124 01280 LD (ERN+1),HL
24A3 3AAD25 01290 LD A,(FCB+8) ;P/u offset byte
24A6 321724 01300 LD (OFFSET+1),A
01310 ;
01320 ; Check if the New Size > Old Size ?
01330 ;
24A9 11A525 01340 BIGGER LD DE,FCB ;DE=> FCB+0
24AC 01350 @@LOF ;Get length of file
24AC 3E40 00018 LD A,64
24AE EF 00019 RST 40
24AF 60 01360 LD H,B ;Move len to HL
24B0 69 01370 LD L,C
24B1 ED4B4C24 01380 LD BC,(RPARM+1) ;P/u # of records
24B5 AF 01390 XOR A ;Clear carry
24B6 E5 01400 PUSH HL ;Save ERN
24B7 ED42 01410 SBC HL,BC ;Is new ERN > old ERN?
24B9 E1 01420 POP HL ;HL = ERN
24BA D23B25 01430 JP NC,BADSIZ ;Go if not
01440 ;
01450 ; Position FCB to Ending Record Number
01460 ;
24BD 0B 01470 DEC BC ;Reduce to offset from 0
24BE 01480 @@POSN ;Position to new ERN
24BE 3E42 00020 LD A,66
24C0 EF 00021 RST 40
01490 ;
01500 ; Fill a 256 byte buffer with X'E5' bytes
01510 ;
24C1 210026 01520 LD HL,BUFFER ;Buffer area
24C4 110126 01530 LD DE,BUFFER+1 ;"format" a sector
24C7 01FF00 01540 LD BC,255
24CA 36E5 01550 LD (HL),0E5H
24CC EDB0 01560 LDIR
01570 ;
01580 ; Write the last Record of the file
01590 ;
24CE 11A525 01600 LD DE,FCB ;Write the new ERN sector
24D1 25 01610 DEC H ;Set HL = buffer start
24D2 23 01620 INC HL
24D3 CD0D25 01630 CALL WRITE ;Write the last record
24D6 01640 @@REW ;Rewind File
24D6 3E44 00022 LD A,68

```

```

24D8 EF      00023      RST      40
              01650 ;
              01660 ;      Read in the directory entry
              01670 ;

24D9 ED4BAB25 01680      LD        BC,(FCB+6)      ;Get drive # & DEC
24DD          01690      @DIRRD      ;Read in record
24DD 3E57      00024      LD        A,87
24DF EF      00025      RST      40
24E0 2032      01700      JR        NZ,IOERR      ;Jump on read error
              01710 ;
              01720 ;      Set the CREATE bit, and write it back out
              01730 ;

24E2 23        01740      INC        HL            ;Point to FCB+1 &
24E3 CBFE      01750      SET        7,(HL)        ; set the CREATE bit
24E5           01760      @DIRWR      ;Write entry back
24E5 3E58      00026      LD        A,88
24E7 EF      00027      RST      40
24E8 202A      01770      JR        NZ,IOERR      ;Jump on write error
              01780 ;
              01790 ;      Do we have to Fill the file ?
              01800 ;

24EA 010001    01810 FILL      LD        BC,0100H      ;P/u FILL parm
24ED 05        01820      DEC        B
24EE C8        01830      RET        Z            ;RETurn if no Fill
              01840 ;
              01850 ;      Create a Buffer with the FILL bytes
              01860 ;

24EF D5        01870      PUSH       DE            ;Save FCB pointer
24F0 210026    01880      LD        HL,BUFFER      ;I/O buffer
24F3 71        01890      LD        (HL),C        ;Byte to xfer
24F4 110126    01900      LD        DE,BUFFER+1
24F7 01FF01    01910      LD        BC,255+256      ;Hit both buffers
24FA EDB0      01920      LDIR              ;Xfer into buffer
              01930 ;
              01940 ;      Pt HL => User Buff, DE => FCB, BC = last Rec
              01950 ;

24FC D1        01960      POP        DE            ;Restore FCB pointer
24FD ED4B4C24 01970      LD        BC,(RPARM+1)      ;P/u last record
2501 210027    01980      LD        HL,UBUFF      ;User Buffer
              01990 ;
              02000 ;      Loop to write logical records
              02010 ;

2504 78        02020 WRLOOP   LD        A,B            ;Is rec cnt = 0 ?
2505 B1        02030      OR        C
2506 C8        02040      RET        Z            ;Yes - done
2507 CD0D25    02050      CALL      WRITE          ;Write Record
250A 0B        02060      DEC        BC            ;Dec one
250B 18F7      02070      JR        WRLOOP        ;Do til BC = 0
              02080 ;
              02090 ;      Write the buffer contents
              02100 ;

250D          02110 WRITE     @WRITE              ;Write buffer
250D 3E4B      00028      LD        A,75
250F EF      00029      RST      40
2510 C8        02120      RET        Z            ;Good - RETurn
2511 21        02130      DB        21H          ;Skip LD A,## instruction
2512 3E2C      02140 PRMERR   LD        A,PAR_ERR      ;Parameter Error
              02150 ;
              02160 ;      I/O error display & abort routine
              02170 ;

2514 6F        02180 IOERR    LD        L,A            ;Save error # in HL

```

```

2515 2600      02190      LD      H,0
2517 F6C0      02200      OR       0C0H      ;Short error message
2519 4F        02210      LD       C,A      ;Stuff in C for @ERROR
251A          02220      @@ERROR      ;Display error message
251A 3E1A      00030      LD       A,26
251C EF        00031      RST      40
251D 310000    02230 SAVESP LD      SP,$-$      ;P/u original SP
2520 C9        02240      RET              ;Done, RETurn
                02250      ;
                02260      ;      Routine to pick up device/file name
                02270      ;
2521 CB7F      02280 $FNAME BIT      7,A      ;Test device/file
2523 2804      02290      JR       Z,FNAME1    ;Go if device
2525          02300      @@FNAME
2525 3E50      00032      LD       A,80
2527 EF        00033      RST      40
2528 C9        02310      RET
2529 3E2A      02320 FNAME1 LD      A,'*'      ;Stuff device indicator
252B 12        02330      LD      (DE),A
252C 13        02340      INC      DE
252D 79        02350      LD      A,C      ;Stuff 1st character
252E 12        02360      LD      (DE),A
252F 13        02370      INC      DE
2530 78        02380      LD      A,B      ;Stuff 2nd character
2531 12        02390      LD      (DE),A
2532 13        02400      INC      DE
2533 3E03      02410      LD      A,3      ;Stuff ETX
2535 12        02420      LD      (DE),A
2536 C9        02430      RET
                02440      ;
                02450      ;      Error Message Display routine
                02460      ;
2537 214625    02470 SPCREQ LD      HL,SPCREQ$
253A DD        02480      DB      0DDH
253B 215925    02490 BADSIZ LD      HL,BADSIZ$
                02500      ;
                02510      ;      Log Error Message & Abort
                02520      ;
253E          02530      @@LOGOT      ;Log error message
                00034      IFEQ      00H,1
                00035      LD       HL,
                00036      ENDIF
253E 3E0C      00037      LD       A,12
2540 EF        00038      RST      40
2541 21FFFF    02540      LD      HL,-1      ;Set abort code
2544 18D7      02550      JR       SAVESP      ;Exit
                02560      ;
                02570      ;      Messages
                02580      ;
2546 46        02590 SPCREQ$ DB      'File spec required',CR
                69 6C 65 20 73 70 65 63
                20 72 65 71 75 69 72 65
                64 0D
2559 46        02600 BADSIZ$ DB      'File exists larger',CR
                69 6C 65 20 65 78 69 73
                74 73 20 6C 61 72 67 65
                72 0D
256C 43        02610 CREATE$ DB      'Creating: '
                72 65 61 74 69 6E 67 3A
                20
000F          02620 FILESP DS      15

```

```

02630 ;
02640 ;PARAMETER TABLE
02650 ;
2585 80 02660 PRMTBL$ DB 80H ;6.x Parameter Table
02670 ;
02680 ; SIZE (S) - Accept Numeric Input only
02690 ;
2586 94 02700 DB NUM!ABB!4
2587 53 02710 DB 'SIZE'
49 5A 45
258B 00 02720 SRESP DB 0
258C 5324 02730 DW SPARM+1
02740 ;
02750 ; REC (R) - Accept Numeric input only
02760 ;
258E 93 02770 DB NUM!ABB!3
258F 52 02780 DB 'REC'
45 43
2592 00 02790 RRESP DB 0
2593 4C24 02800 DW RPARM+1
02810 ;
02820 ; LRL (L) - Accept Numeric input only
02830 ;
2595 93 02840 DB NUM!ABB!3
2596 4C 02850 DB 'LRL'
52 4C
2599 00 02860 LRESP DB 0
259A 6024 02870 DW LPARM+1
02880 ;
02890 ; FILL (F) - Accept Numeric or Flag input
02900 ;
259C D4 02910 DB FLAG!NUM!ABB!4
259D 46 02920 DB 'FILL'
49 4C 4C
25A1 00 02930 FRESP DB 0
25A2 EB24 02940 DW FILL+1
25A4 00 02950 DB 0
02960 ;
02970 ; I/O buffer
02980 ;
25A5 00 02990 FCB DB 0
001F 03000 DS 31
2600 03010 ORG $<-8+1<8
0100 03020 BUFFER DS 256
0100 03030 UBUFF DS 256
03040 ;
2400 03050 END CREATE

```

\$FNAME	2521	@@1	0000	@@2	0000
@@3	0000	@@4	0000	@MOD2	0000
@MOD4	FFFF	ABB	0010	AP	0027
BADSI\$	253B	BADSI\$	2559	BEGINA	2409
BIGGER	24A9	BREAK	0080	BS	0008
BUFFER	2600	CFLAG\$	0002	CR	000D
CREATCD	2428	CREATE	2400	CREATE\$	256C
DFLAG\$	0003	ERN	2410	ETX	0003
FCB	25A5	FILESP	2576	FILL	24EA
FLAG	0040	FNAME1	2529	FRESP	25A1
IOERR	2514	KFLAG\$	000A	LF	000A
LP1	246E	LPARM	245F	LRESP	2599
NUM	0080	OFFSET	2416	PAR ERR	002C
PRMERR	2512	PRMTBL\$	2585	RPARM	244B
RRESP	2592	SAVESP	251D	SFLAG\$	0012
SPARM	2452	SPCREQ	2537	SPCREQ\$	2546
SRESP	258B	STR	0020	TAB	0009
UBUFF	2700	VFLAG\$	0015	WRITE	250D
WLOOP	2504	@@ABORT	8400	@@ADTSK	8493
@@BANK	89AB	@@BKSP	868B	@@BREAK	89C1
@@CHNIO	83EB	@@CKBRKC	8A0F	@@CKDRV	84E7
@@CKEOF	86A0	@@CKTSK	847E	@@CLOSE	8676
@@CLS	89F9	@@CMNDI	842A	@@CMNDR	843F
@@CTL	824F	@@DATE	83C1	@@DCSTAT	8526
@@DEBUG	8469	@@DECH\$X	892B	@@DIRRD	8898
@@DIRWR	88AD	@@DIV16	8916	@@DIV8	8901
@@DODIR	84FC	@@DSP	8213	@@DSPLY	82B3
@@ERROR	8454	@@EXIT	8415	@@FEXT	8805
@@FLAGS	8995	@@FNAME	881A	@@FSPEC	87F0
@@GATRD	8883	@@GATWR	88C2	@@GET	8227
@@GTDCB	8844	@@GTDCT	882F	@@GTMOD	8859
@@HDFMT	85CE	@@HEX16	896A	@@HEX8	8955
@@HEXDEC	8940	@@HIGH\$	897F	@@INIT	864C
@@KBD	828B	@@KEY	81FF	@@KEYIN	829F
@@KLTSK	84D2	@@LOAD	87C6	@@LOC	86B5
@@LOF	86CA	@@LOGGER	82EA	@@LOGOT	82FF
@@MSG	8336	@@MUL16	88EC	@@MUL8	88D7
@@OPEN	8661	@@PARAM	83AC	@@PAUSE	8397
@@PEOF	86DF	@@POSN	86F4	@@PRINT	834B
@@PRT	8263	@@PUT	823B	@@RAMDIR	8511
@@RDSEC	85A4	@@RDSSC	886E	@@READ	8709
@@REMOV	8637	@@RENAM	8622	@@REW	871E
@@RMTSK	84A8	@@RPTSK	84BD	@@RREAD	8733
@@RSLCT	858F	@@RSTOR	8550	@@RUN	87DB
@@RWRT	8748	@@SEEK	857A	@@SEEKSC	875D
@@SKIP	8772	@@SLCT	853B	@@STEPI	8565
@@TIME	83D6	@@VDCTL	8382	@@VER	8787
@@VRSEC	85B9	@@WEOF	879C	@@WHERE	8277
@@WRITE	87B1	@@WRSEC	85E3	@@WRSSC	85F8
@@WRTK	860D				

2400 is the transfer address

00000 Total errors

NOTES:

NOTES:

Command: DATE, TIME

Library: SYS7/SYS

ISAM # : 15H, 16H

```

00100 ;LBDATE/ASM - Date/Time Commands
0000 00110 TITLE <DATE/TIME - LS-DOS 6.2>
00120 ;
0000 00130 *GET SVCMAC:3 ;SVC Macro equivalents
00010 ;SVMAC/ASM - LS-DOS Version VI
00020 *LIST OFF
00300 *LIST ON
00140 *LIST OFF ;Get LDOS60/EQU
00160 *LIST ON
00170 ;
000D 00180 CR EQU 13
001D 00190 @ADTSK EQU 29 ;Add Task SVC #
001E 00200 @RMTSK EQU 30 ;Remove Task SVC #
0005 00210 CLK_SLT EQU 5 ;Clock Task Slot #
00220 ;
00230 ;
00240 ; DATE$ Storage
00250 ;
00260 ; DATE$+0 Year (80-87)
00270 ; DATE$+1 Day of the month (1-31)
00280 ; DATE$+2 Month (1-12)
00290 ; DATE$+3 Bits 0-7 of Day of Year
00300 ; Date$+4 Bit 0 = bit 8 of Day of Year
00310 ; Bits 1-3 contain Day of Week
00320 ; Bit 7 set if a leap year
00330 ;
00340 ; TIME$ Storage
00350 ;
00360 ; TIME$+0 Seconds (0-59)
00370 ; TIME$+1 Minutes (0-59)
00380 ; TIME$+2 Hours (0-23)
00390 ;
00400 ;
00410 ;
2400 00420 ORG 2400H
00430 ;
00440 ; Branch to TIME entry point
00450 ;
2400 C30A25 00460 JP TIME ;Time entry point
00470 ;
2403 00480 SUBTTL '<LBDATE - DATE Code>'
00490 ;
2403 00500 DATE @@CKBRKC ;Break key down?
2403 3E6A 00001 LD A,106
2405 EF 00002 RST 40
2406 2804 00510 JR Z,BEGINA ;Ok if not
2408 21FFFF 00520 LD HL,-1 ; else abort
240B C9 00530 RET
00540 ;
00550 ; DATE - Pick up DATE$+0 pointer & stuff in IY
00560 ;
240C E5 00570 BEGINA PUSH HL ;Save command ptr
240D 218E26 00580 LD HL,DUMBUF ;HL => Dummy Buffer
2410 00590 @@DATE ;DE <= DATE$+0
2410 3E12 00003 LD A,18
2412 EF 00004 RST 40
2413 E1 00600 POP HL ;Recover command ptr
2414 D5 00610 PUSH DE ;Xfer ptr to IY
2415 FDE1 00620 POP IY
00630 ;
00640 ; Was a Date entered on the command line ?

```

LBDATE - DATE Code

```

00650 ;
2417 7E 00660 LD A,(HL) ;P/u character
2418 FE0E 00670 CP CR+1 ;Date entry ?
241A DA5B24 00680 JP C,DSPDATE ;No - display date
00690 ;
00700 ; Date entered - check if legal format
00710 ;
241D 0E30 00720 LD C,'0' ;Init separator
241F CDA025 00730 CALL PARSDAT ;Parse entry
2422 C20325 00740 JP NZ,BADFMT ;Bad entry - abort
00750 ;
00760 ; Legal Date - If Intl date - swap DTBUF+1 & 2
00770 ;
00780 IF @INTL
00790 INC DE ;DE => DTBUF+1
00800 LD H,D ;HL => DTBUF+1
00810 LD L,E
00820 LD C,(HL) ;C = Intl MONTH
00830 INC HL ;HL => DTBUF+2
00840 LD A,(HL) ;A = Intl DAY
00850 LD (DE),A ;Set DTBUF+1 = DAY
00860 LD (HL),C ;Set DTBUF+2 = MONTH
00870 DEC DE ;DE => DTBUF+0
00880 ELSE
2425 00 00890 DC 9,0 ;Pad US ver to match Intl
00 00 00 00 00 00 00 00
00900 ENDIF
00910 ;
00920 ; Is the year legal ?
00930 ;
242E 1A 00940 LD A,(DE) ;P/u year entry
242F D650 00950 SUB 80 ;Accept only 80-87
2431 DA0325 00960 JP C,BADFMT ;Less than 80 - bad
2434 FE08 00970 CP 8 ;Greater than 87 ?
2436 3018 00980 JR NC,BADFMT2 ;Yes - bad
00990 ;
01000 ; If Year is 1980 or 84 then set FEB = 29 days
01010 ;
2438 E603 01020 AND 3 ;0 or 4 ?
243A 214D26 01030 LD HL,MAXDAYS+2 ;Set Feb to have 29 days
243D 2001 01040 JR NZ,NOTLEAP ;No - don't inc
243F 34 01050 INC (HL) ;Leap year - inc max days
01060 ;
01070 ; Check Range of month - must be 1 - 12
01080 ;
2440 3A8D26 01090 NOTLEAP LD A,(DTBUF+2) ;P/u month
2443 3D 01100 DEC A ;Set month = 0-11
2444 FE0C 01110 CP 12 ;Valid Month ?
2446 3008 01120 JR NC,BADFMT2 ;Abort if 0 or >12
01130 ;
01140 ; Valid month - point HL to max days/month
01150 ;
2448 2B 01160 DEC HL ;HL => Max day table
2449 85 01170 ADD A,L ;Add month # to start
244A 6F 01180 LD L,A ;HL => Max days for month
01190 ;
01200 ; Check for Day entry is valid
01210 ;
244B 3A8C26 01220 LD A,(DTBUF+1) ;P/u day entry

```

LBDATE - DATE Code

```

244E 3D      01230      DEC      A      ;Reduce for test (0->FF)
244F BE      01240      CP      (HL)    ;More than max days ?
2450 D20325  01250 BADFMT2 JP      NC,BADFMT ;Go if too large (or 0)
                01260 ;
                01270 ;      Transfer Date into buffer
                01280 ;
2453 EB      01290      EX      DE,HL    ;Point HL to DTBUF
2454 FDE5    01300      PUSH    IY      ;Point DE = DATE$
2456 D1      01310      POP     DE
2457 0E03    01320      LD      C,3      ;BC = 3 chars to xfer
2459 EDB0    01330      LDIR           ;Xfer 3 chars
                01340 ;
                01350 ;      Display "No Date in System" if illegal Date
                01360 ;
245B FD7E02  01370 DSPDATE LD      A,(IY+2) ;P/u month
245E 215826  01380      LD      HL,NODATE$ ;"No Date in system"
2461 B7      01390      OR      A      ;Better not be zero
2462 CA0625  01400      JP      Z,LOGABRT ;Log & abort
2465 47      01410 GOTDATE LD      B,A    ;Xfer month to B
2466 214D26  01420      LD      HL,MAXDAYS+2 ;Adjust February if
2469 7E      01430      LD      A,(HL)    ; year is leap year
246A D61D    01440      SUB     29      ; & not already adjusted
246C 2808    01450      JR      Z,PUDAY   ; in parsing date entry
                01460 ;
                01470 ;      Pick up Year & increment max days if leap yr
                01480 ;
246E FD7E00  01490      LD      A,(IY)    ;P/u year
2471 E603    01500      AND     3      ;1980 and 1984 are lp yrs
2473 2001    01510      JR      NZ,PUDAY  ;Not leap year - fine
2475 34      01520      INC     (HL)    ;Bump to 29
                01530 ;
                01540 ;      Set HL = day # this month, DE => Max table
                01550 ;
2476 FD6E01  01560 PUDAY  LD      L,(IY+1) ;P/u day # this month
2479 2600    01570      LD      H,0      ; in HL
247B 114B26  01580      LD      DE,MAXDAYS ;DE => Max day table
                01590 ;
                01600 ;      Loop to Count up total # of days up to now
                01610 ;
247E 1A      01620 DAYLP  LD      A,(DE)  ;P/u max day
247F 85      01630      ADD     A,L      ;Add to HL
2480 6F      01640      LD      L,A
2481 8C      01650      ADC     A,H
2482 95      01660      SUB     L
2483 67      01670      LD      H,A
2484 13      01680      INC     DE      ;Bump days ptr
2485 10F7    01690      DJNZ    DAYLP    ;B months of max days
                01700 ;
                01710 ;      Stuff days (9 bits) into DATE$
                01720 ;
2487 FD7503  01730      LD      (IY+3),L ;Stuff in lsb
248A 7C      01740      LD      A,H      ;Get bit "8"
248B FDB604  01750      OR      (IY+4)   ; and OR it in
248E FD7704  01760      LD      (IY+4),A ;Then put it back
                01770 ;
                01780 ;      Pick up year in E (0-7)
                01790 ;
2491 FD7E00  01800      LD      A,(IY)    ;P/u year
2494 D650    01810      SUB     80      ;Offset from 80

```

LBDATE - DATE Code

```

2496 5F      01820 LD      E,A          ;Put 0-7 in E
2497 C603    01830 ADD      A,3          ;Ck for year >= 84
2499 0F      01840 RRCA
249A 0F      01850 RRCA
249B E603    01860 AND      3          ;Keep bits 0,1
249D 83      01870 ADD      A,E          ;Add back to year
249E 5F      01880 LD      E,A          ; & save in DE
249F 1600    01890 LD      D,0
24A1 19      01900 ADD      HL,DE        ;Add to days in year
24A2 23      01910 INC      HL          ;To start in right place
          01920 ;
          01930 ;
          01940 ;
          HL = desired number to divide by seven
24A3 010700  01950 LD      BC,7          ;Now divide by 7
24A6 AF      01960 XOR      A
24A7 ED42    01970 DIV7  SBC      HL,BC        ;Subtract weeks (7-days)
24A9 30FC    01980 JR      NC,DIV7        ; until under flow
          01990 ;
          02000 ;
          02010 ;
          Correct # for division, & put in bits 1-3
24AB 7D      02020 LD      A,L
24AC C608    02030 ADD      A,8          ;Add back to get 1-7
24AE 47      02040 LD      B,A          ;Save day of week
24AF 07      02050 RLCA          ;Shift to bits 1-3
24B0 4F      02060 LD      C,A          ; to store in DATE$
          02070 ;
          02080 ;
          02090 ;
          Merge day of week with bit 9 of day of year
24B1 FD7E04  02100 LD      A,(IY+4)        ;P/u DATE$ + 4
24B4 E6F1    02110 AND      0FH          ;Keep lp yr bit & bit 0
24B6 B1      02120 OR      C          ;Merge day of week
24B7 FD7704  02130 LD      (IY+4),A        ;Stuff back in
          02140 ;
          02150 ;
          02160 ;
          Transfer Day string into display buffer
24BA 210026  02170 LD      HL,DAYTBL        ;HL => Day string table
24BD 113926  02180 LD      DE,DATEBUF        ;Date display buffer
24C0 D5      02190 PUSH     DE          ;Save start
24C1 CD9125  02200 CALL    DSPMDY        ;Write out the day
          02210 ;
          02220 ;
          02230 ;
          Position DE to month destination in buffer
24C4 13      02240 INC      DE          ;Bump
24C5 13      02250 INC      DE
          02260 ;
          02270 ;
          02280 ;
          Pick up month, & stuff string into buffer
24C6 FD7E02  02290 LD      A,(IY+2)        ;P/u month number
24C9 47      02300 LD      B,A          ;Stuff in B
24CA 211526  02310 LD      HL,MONTBL        ;HL => Month string table
24CD CD9125  02320 CALL    DSPMDY        ;Write out the month name
          02330 ;
          02340 ;
          02350 ;
          P/u day of the month & convert to ASCII
24D0 13      02360 INC      DE          ;DE => Day destination
24D1 FD7E01  02370 LD      A,(IY+1)        ;P/u day
24D4 06FF    02380 LD      B,-1          ;Init # of tens to -1
          02390 ;
          02400 ;
          Divide day of the month by 10

```

LBDATE - DATE Code

```

02410 ;
24D6 04      02420 DIV10 INC B ;Divide by 10
24D7 D60A    02430 SUB 10 ; with quotient in B
24D9 30FB    02440 JR NC,DIV10 ;Subtract until carry
02450 ;
02460 ; Convert to tens digit to ASCII
02470 ;
24DB F5      02480 PUSH AF ;Save (10-remainder)
24DC 78      02490 LD A,B ;P/u quotient
24DD C630    02500 ADD A,'0' ;Change to ASCII
02510 ;
02520 ; Change to a space if it's a leading zero
02530 ;
24DF FE30    02540 CP '0' ;Zero?
24E1 2002    02550 JR NZ,NOTLD0 ;No - use it
24E3 3E20    02560 LD A,' ' ;Change leading 0 to ' '
24E5 12      02570 NOTLD0 LD (DE),A ;Stuff in buffer
02580 ;
02590 ; Convert remainder to ASCII & stuff in buffer
02600 ;
24E6 13      02610 INC DE ;DE => ones destination
24E7 F1      02620 POP AF ;Get back remainder
24E8 C63A    02630 ADD A,3AH ;Change to ASCII
24EA 12      02640 LD (DE),A ;Stuff in buffer
02650 ;
02660 ; P/u year & stuff lower digit + "0" in buffer
02670 ;
24EB FD7E00  02680 LD A,(IY) ;Form last year digit
24EE D650    02690 SUB 80 ;A = 0-7
24F0 C630    02700 ADD A,'0' ;Convert to ASCII
24F2 324926  02710 LD (DATEBUF+16),A ;Stuff year
02720 ;
02730 ; Set B = 0 (Normal Exit)
02740 ;
24F5 0600    02750 LOGDT LD B,0 ;B = 0 (normal exit)
24F7 E1      02760 POP HL ;HL => Date/Time string
02770 ;
02780 ; Display Date or Time String
02790 ;
24F8 C5      02800 LOGMSG PUSH BC ;Save Error #, B (exit)
24F9         02810 @@LOGOT ;Log message
000005      IFEQ 00H,1
000006      LD HL,
000007      ENDIF
24F9 3E0C    000008 LD A,12
24FB EF      000009 RST 40
24FC C1      02820 POP BC ;B = exit condition
02830 ;
02840 ; If B = 0 then exit HL = 0, otherwise HL = -1
02850 ;
24FD 60      02860 LD H,B ;Set HL = -1 or 0
24FE 68      02870 LD L,B ;
24FF         02880 @@CKBRKC ;Clear any break
24FF 3E6A    000010 LD A,106
2501 EF      000011 RST 40
2502 C9      02890 RET ;RETurn with condition
02900 ;
02910 ; Bad Format - display error & abort
02920 ;

```

LBDATE - DATE Code

2503 216B26	02930	BADFMT	LD	HL,BADDAT\$;Illegal Date/Time
2506 06FF	02940	LOGABRT	LD	B,-1	;Abort Condition
2508 18EE	02950		JR	LOGMSG	;Log Message
	02960	;			
250A	02970	SUBTTL '<LBDATE - TIME code>'			

LBDATE - TIME code

```

250A      02980      PAGE
          02990 ;
250A      03000 TIME  @@CKBRKC      ;Break key down?
250A 3E6A      00012      LD      A,106
250C EF      00013      RST      40
250D 2804      03010      JR      Z,BEGINB      ;Ok if not
250F 21FFFF      03020      LD      HL,-1      ; else abort
2512 C9      03030      RET
          03040 ;
          03050 ;      TIME entry point - Any Parms entered ?
          03060 ;
2513 E5      03070 BEGINB  PUSH     HL      ;Save pointer
2514 7E      03080 TLOOP  LD      A,(HL)      ;P/u character
2515 FE28      03090      CP      '('      ;Any parameters ?
2517 2807      03100      JR      Z,GETPRMS      ;Yes - get 'em
2519 FE0D      03110      CP      CR      ;End of line ?
251B 2809      03120      JR      Z,CLRSTK      ;Yes - go check time
251D 23      03130      INC     HL      ;Bump ptr
251E 18F4      03140      JR      TLOOP      ;Do til terminator
          03150 ;
          03160 ;      Process any Parameters
          03170 ;
2520 11D925      03180 GETPRMS LD      DE,PRMTBL$      ;DE => Parameter Table
2523      03190      @@PARAM      ;Get parameters
2523 3E11      00014      LD      A,17
2525 EF      00015      RST      40
          03200 ;
          03210 ;      Stuff "Illegal Time" Mess in error routine
          03220 ;
2526 217B26      03230 CLRSTK LD      HL,BADTIM$      ;Chg "Bad date format"
2529 220425      03240      LD      (BADFMT+1),HL      ; to "Bad time..."
252C E1      03250      POP     HL      ;Recover command ptr
252D 280A      03260      JR      Z,GDPARMS      ;Z - ok to continue
          03270 ;
          03280 ;      Parameter Error - Display & abort
          03290 ;
252F 6F      03300 IOERR  LD      L,A      ;Xfer errcod to HL
2530 2600      03310      LD      H,0
2532 F6C0      03320      OR      C0H      ;Short error
2534 4F      03330      LD      C,A      ;Xfer to C
2535      03340      @@ERROR      ;Log error
2535 3E1A      00016      LD      A,26
2537 EF      00017      RST      40
2538 C9      03350      RET
          03360 ;
          03370 ;      Was there a TIME string entered ?
          03380 ;
2539 7E      03390 GDPARMS LD      A,(HL)      ;P/u char
253A FE28      03400      CP      '('      ;Parms only?
253C CA8425      03410      JP      Z,DSPTIME      ;Display old time
253F FE0D      03420      CP      CR      ;End of line?
2541 CA8425      03430      JP      Z,DSPTIME      ;Display old time
          03440 ;
          03450 ;      Requested time set - Check if legal format
          03460 ;
2544 0E30      03470      LD      C,'0'      ;Init separator
2546 CDA025      03480      CALL   PARSDAT      ;Parse entry
2549 20B8      03490      JR      NZ,BADFMT      ;Bad - abort
          03500 ;
          03510 ;      Legal Format - Check if Hours are legal

```

LBDATE - TIME code

```

03520 ;
254B 218D26 03530 LD HL,DTBUF+2 ;HL => Hours byte
254E 3E17 03540 LD A,23 ;Greater than 23 ?
2550 BE 03550 CP (HL)
2551 38B0 03560 JR C,BADFMT ;Yes - bad format
03570 ;
03580 ; Hours legal - Check if minutes legal
03590 ;
2553 2B 03600 DEC HL ;HL => Minutes
2554 3E3B 03610 LD A,59 ;Greater than 59 ?
2556 BE 03620 CP (HL)
2557 38AA 03630 JR C,BADFMT ;Yes - bad format
03640 ;
03650 ; Minutes legal - Check if seconds legal
03660 ;
2559 2B 03670 DEC HL ;HL => Seconds
255A BE 03680 CP (HL) ;Greater than 59 ?
255B 38A6 03690 JR C,BADFMT ;Yes - bad format
03700 ;
03710 ; Legal input - transfer to TIME$ storage area
03720 ;
255D E5 03730 PUSH HL ;Save TIME buffer ptr
255E 218E26 03740 LD HL,DUMBUF ;HL => dummy buffer
2561 03750 @@TIME ;DE <= TIME$+0
2561 3E13 00018 LD A,19
2563 EF 00019 RST 40
2564 E1 03760 POP HL ;Recover TIME buffer ptr
2565 010300 03770 LD BC,3 ;3 bytes to xfer
2568 EDB0 03780 LDIR ;Xfer
03790 ;
03800 ; Was the CLOCK (C) parameter entered ?
03810 ;
256A 210000 03820 DOCLOCK LD HL,0 ;HL = 0 (Normal Exit)
256D 3AE025 03830 LD A,(CRESP) ;P/u response
2570 B7 03840 OR A
2571 C8 03850 RET Z ;RETurn if no response
03860 ;
03870 ; CLOCK (C) parameter entered - ON or OFF ?
03880 ;
2572 110000 03890 CLOCK LD DE,$-$ ;P/u parm = FFFF or 0000
2575 03900 @@FLAGS ;IY => System Flags
2575 3E65 00020 LD A,101
2577 EF 00021 RST 40
03910 ;
03920 ; Just Set/Reset CLOCK bit if Model IV version
03930 ;
03940 IF @MOD4
2578 FDCB15E6 03950 SET 4,(IY+'V'-'A') ;Set Clock bit
257C 1C 03960 INC E ;Return if CLOCK = YES
257D C8 03970 RET Z ;
257E FDCB15A6 03980 RES 4,(IY+'V'-'A') ;Otherwise Reset bit
2582 1D 03990 DEC E ;Set Z flag
2583 C9 04000 RET ;Done - RETurn
04010 ENDIF
04020 ;
04030 ; Also Add or Remove Task if Model II Version
04040 ;
04050 IF @MOD2
04060 RES 4,(IY+'V'-'A') ;Reset clock bit

```

LBDATE - TIME code

```

04070 LD C,CLK_SLT ;Set C = Clock Slot #
04080 LD A,@RMTSK ;A = Remove Task SVC #
04090 INC E ;Clock Off ?
04100 JR NZ,CLOFF ;Yes - remove task
04110 CLON LD DE,DO_CLOCK ;Clock on - DE => Address
04120 SET 4,(IY+'V'-'A') ;Set clock bit
04130 LD A,@ADTSK ;A = Add Task SVC #
04140 CLOFF RST 40 ;Issue SVC
04150 XOR A ;Set Z for no error
04160 LD H,A ;Pass to HL
04170 LD L,A ;For normal exit
04180 RET
04190 ENDIF
04200 ;
04210 ; Display the Time
04220 ;
2584 214226 04230 DSPTIME LD HL,DATEBUF+9 ;Pt to space for time str
2587 E5 04240 PUSH HL ;Save pointer
2588 04250 @@TIME ;Xfer time into buffer
2588 3E13 00022 LD A,19
258A EF 00023 RST 40
258B CD6A25 04260 CALL DO_CLOCK ;Set/Reset Clock bit
258E C3F524 04270 JP LOGDT ;Log it & exit
2591 04280 SUBTTL '<LBDATE - DATE/TIME Common Routines>'

```

LBDATE - DATE/TIME Common Routines

```

2591      04290      PAGE
          04300 ;
          04310 ;      DSPMDY - Xfer 3 char string from table to buffer
          04320 ;
          04330 ;      B => Entry # in table to display
          04340 ;      HL => Table to fetch data from
          04350 ;      DE => Buffer to receive string
          04360 ;
2591 05      04370 DSPMDY DEC      B      ;B = entry, 0-6
2592 7D      04380      LD      A,L      ;P/u lsb of table start
2593 80      04390      ADD      A,B
2594 80      04400      ADD      A,B      ;Entries 3 bytes long
2595 80      04410      ADD      A,B
2596 6F      04420      LD      L,A      ;HL => Table entry
          04430 ;
          04440 ;      Transfer string into buffer
          04450 ;
2597 0603    04460      LD      B,3      ;Three chars to xfer
2599 7E      04470 DSPM1  LD      A,(HL)  ;P/u char from table
259A 12      04480      LD      (DE),A    ;Stuff into buffer
259B 23      04490      INC      HL      ;Bump
259C 13      04500      INC      DE
259D 10FA    04510      DJNZ     DSPM1     ;Three chars to xfer
259F C9      04520      RET              ;Done - RETURN
          04530 ;
          04540 ;      PARSDAT - Parse TIME/DATE string entry
          04550 ;
          04560 ;      HL => Buffer containing string to parse
          04570 ;      C => Delimiter (<"0" = DATE, <"0"or=":" = TIME)
          04580 ;
          04590 ;      DTBUF-DTBUF+2 <= Data in compressed format
          04600 ;      Z - Set if successful
          04610 ;
25A0 118D26  04620 PARSDAT LD      DE,DTBUF+2 ;Point to buf end
25A3 0603    04630      LD      B,3      ;Process 3 fields
          04640 ;
          04650 ;      Parse a field - Return NZ if bad
          04660 ;
25A5 D5      04670 PRS1  PUSH     DE      ;Save pointer
25A6 CDBC25  04680      CALL     PRS2     ;Get a digit pair
25A9 D1      04690      POP      DE      ;Recover pointer
25AA C0      04700      RET      NZ      ;Ret if bad digit pair
          04710 ;
          04720 ;      Good field - Stuff in buff, dec ptr, & count
          04730 ;
25AB 12      04740      LD      (DE),A    ; else stuff the value
25AC 05      04750      DEC      B      ;Loop countdown
25AD C8      04760      RET      Z      ;Do for 3 fields
25AE 1B      04770      DEC      DE      ;Backup the pointer
          04780 ;
          04790 ;      Parsed a field - is the separator valid ?
          04800 ;
25AF 7E      04810      LD      A,(HL)    ;P/u separator
25B0 23      04820      INC      HL      ;Bump pointer
25B1 FE3A    04830      CP      ':'      ;Check for ':'
25B3 28F0    04840      JR      Z,PRS1     ; loop if so
25B5 B9      04850      CP      C      ;Correct ?
25B6 3002    04860      JR      NC,PRSRET ;NC = bad
25B8 18EB    04870      JR      PRS1     ;Loop if OK
          04880 ;

```

LBDATE - DATE/TIME Common Routines

```

25BA B7      04890 PRSRET OR      A      ;Set NZ
25BB C9      04900 RET              ;Return bad entry
              04910 ;
              04920 ;      PRS2 - Parse a digit pair at HL
              04930 ;
25BC CDD225  04940 PRS2 CALL     PRS4      ;Get a digit
25BF 300F    04950 JR        NC,PRS3      ;Illegal - return
              04960 ;
              04970 ;      Legal Digit - Multiply by 10
              04980 ;
25C1 5F      04990 LD          E,A        ;Multiply by ten
25C2 07      05000 RLCA              ;X 2
25C3 07      05010 RLCA              ;X 4
25C4 83      05020 ADD          A,E        ;X 5
25C5 07      05030 RLCA              ;X 10
25C6 5F      05040 LD          E,A        ;Stuff in E
              05050 ;
              05060 ;      Get another digit
              05070 ;
25C7 CDD225  05080 CALL     PRS4      ;Get ones digit
25CA 3004    05090 JR        NC,PRS3      ;Bad - return NZ
              05100 ;
              05110 ;      Legal digit - Add to tens digit & set Z flag
              05120 ;
25CC 83      05130 ADD          A,E        ;Accumulate new digit
25CD 5F      05140 LD          E,A        ;Save 2-digit value
25CE BF      05150 CP          A          ;Clear flags
25CF C9      05160 RET              ;Return Z
              05170 ;
              05180 ;      Force NZ & Return
              05190 ;
25D0 B7      05200 PRS3 OR          A      ;Set NZ
25D1 C9      05210 RET              ;Return
              05220 ;
              05230 ;      Pick up a digit and convert to binary
              05240 ;
25D2 7E      05250 PRS4 LD          A,(HL)  ;P/u a digit &
25D3 23      05260 INC          HL        ; bump ptr
25D4 D630    05270 SUB          '0'      ;Convert to binary
25D6 FE0A    05280 CP          10        ;Legal ?
25D8 C9      05290 RET              ;C - legal, NC - illegal
              05300 ;
              05310 ;      Parameter table
              05320 ;
0080         05330 NUM      EQU      80H
0040         05340 FLAG     EQU      40H
0020         05350 STR      EQU      20H
0010         05360 ABB      EQU      10H
              05370 ;
25D9 80      05380 PRMTBL$ DB      80H      ;6.x Parameter Table
              05390 ;
25DA 55      05400 DB          FLAG!ABB!5
25DB 43      05410 DB          'CLOCK'
              4C 4F 43 4B
25E0 00      05420 CRESP    DB          0
25E1 7325    05430 DW          CLOCK+1
25E3 00      05440 DB          0
              05450 ;
2600         05460 ORG      $<-8+1<+8

```

LBDATE - DATE/TIME Common Routines

```

                05470 ;
2600 53          05480 DAYTBL DB      'SunMonTueWedThuFriSat'
        75 6E 4D 6F 6E 54 75 65
        57 65 64 54 68 75 46 72
        69 53 61 74
2615 4A          05490 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
2639 44          05500 DATEBUF DB    'Day, Mon xx, 198x',CR
        61 79 2C 20 4D 6F 6E 20
        78 78 2C 20 31 39 38 78
        0D
264B 00          05510 MAXDAYS DB    0,31,28,31,30,31,30,31,31,30,31,30,31
        1F 1C 1F 1E 1F 1E 1F 1F
        1E 1F 1E 1F
2658 44          05520 NODATE$ DB    'Date not in system',CR
        61 74 65 20 6E 6F 74 20
        69 6E 20 73 79 73 74 65
        6D 0D
266B 42          05530 BADDAT$ DB    'Bad Date format',CR
        61 64 20 44 61 74 65 20
        66 6F 72 6D 61 74 0D
267B 42          05540 BADTIM$ DB    'Bad Time format',CR
        61 64 20 54 69 6D 65 20
        66 6F 72 6D 61 74 0D
                05550 ;
                05560 ;
268B          05570 DTBUF EQU        $
268E          05580 DUMBUF EQU       $+3
                05590 ;
2403          05600                END    DATE

```

LBDATE - DATE/TIME Common Routines

@\$SYS	08F0	@@1	0000	@@2	0000
@@3	0000	@@4	0000	@ADTSK	0010
@BANK	0877	@BYTEIO	1300	@CHNIO	0689
@CKBRKC	0553	@CLS	0545	@CTL	0623
@DATE	07A8	@DIV16	06E3	@DSP	0642
@DSPLY	052D	@FRENCH	0000	@GERMAN	0000
@GET	0638	@HEX16	07BD	@HEX8	07C2
@HEXDEC	06F6	@HZ50	0000	@INTL	0000
@JCL	0630	@KBD	0635	@KEY	0628
@KEYIN	0585	@KITSK	0089	@LOGER	0503
@LOGOT	0500	@MOD2	0000	@MOD4	FFFF
@MSG	0530	@MUL16	06C9	@OPREG	0084
@PRINT	0528	@PRT	063D	@PUT	0645
@RTSK	001E	@RSTNMI	0FE9	@RSTREG	0680
@TIME	078D	@USA	FFFF	@VDCTL	0B99
@VDCTL3	0D38	@_VDCTL	0D42	ABB	0010
ADDR_2_ROWCOL	0DF1	BADDAT\$	266B	BADFMt	2503
BADFMt2	2450	BADTIM\$	267B	BAR\$	0201
BEGINA	240C	BEGINB	2513	BOOTST\$	439D
BUR\$	0200	CASHK\$	0A7B	CFLAG\$	006C
CLK_SLT	0005	CLOCK	2572	CLRSTK	2526
CORE\$	0300	CR	000D	CRESP	25E0
CRTBGN\$	F800	DATE	2403	DATE\$	0033
DATEBUF	2639	DAYLP	247E	DAYTBL	2600
DAYTBL\$	04C7	DCBKL\$	0031	DCT\$	0470
DFLAG\$	006D	DIS_DO_RAM	0846	DIV10	24D6
DIV7	24A7	DOCLOCK	256A	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	DSKTPY\$	04C0
DSPDATE	245B	DSPM1	2599	DSPMDY	2591
DSPTIME	2584	DTBUF	268B	DTPMT\$	04C2
DUMBUF	268E	DVREND\$	0FF4	DVRHI\$	0206
ENADIS_DO_RAM	0817	FDDINT\$	000E	FLAG	0040
FLGTAB\$	006A	GDPARMS	2539	GETPRMS	2520
GET_@_ROWCOL	0DAE	GOTDATE	2465	HERTZ\$	0750
HIGH\$	040E	IFLAG\$	0072	INBUF\$	0420
INTVC\$	003E	IOERR	252F	JCLCB\$	0203
JLDCB\$	0230	KCK@	07D6	KFLAG\$	0074
KIDATA\$	08FC	KIDCB\$	0208	LBANK\$	0202
LOGABRT	2506	LOGDT	24F5	LOGMSG	24F8
MAXDAY\$	0401	MAXDAYS	264B	MODOUT\$	0076
MONTBL	2615	MONTBL\$	04DC	NFLAG\$	0077
NODATE\$	2658	NOTLD0	24E5	NOTLEAP	2440
NUM	0080	OPREG\$	0078	OPREG_SV_AREA	086E
OPREG_SV_PTR	0835	PAKNAM\$	0410	PARSDAT	25A0
PAUSE@	0382	PCSAVE\$	07AF	PDRV\$	001B
PRDCB\$	0218	PRMTBL\$	25D9	PRS1	25A5
PRS2	25BC	PRS3	25D0	PRS4	25D2
PRSRET	25BA	PUDAY	2476	PUTA@DE	0DCD
PUT_@	0DCA	PUT_@_ROWCOL	0DC6	RFLAG\$	007B
ROWCOL_2_ADDR	0DD0	RSTOR\$	04C4	S1DCB\$	0238
SET_SCROLL	0CF3	SFLAG\$	007C	SIDCB\$	0220
SODCB\$	0228	STACK\$	0380	START\$	0000
STR	0020	TIME	250A	TIME\$	002D
TIMER\$	002C	TIMSL\$	002B	TIMTSK\$	0713
TLOOP	2514	TMPMT\$	04C3	TRACE_INT	07B1

LBDATE - DATE/TIME Common Routines

TYPHK\$	0A8F TYPTSK\$	0B26 VFLAG\$	007F
ZERO\$	0401 @@ABORT	9E09 @@ADTSK	9E9C
@@BANK	A3B4 @@BKSP	A094 @@BREAK	A3CA
@@CHNIO	9DF4 @@CKBRKC	A418 @@CKDRV	9EF0
@@CKEOF	A0A9 @@CKTSK	9E87 @@CLOSE	A07F
@@CLS	A402 @@CMNDI	9E33 @@CMNDR	9E48
@@CTL	9C58 @@DATE	9DCA @@DCSTAT	9F2F
@@DEBUG	9E72 @@DECHEX	A334 @@DIRRD	A2A1
@@DIRWR	A2B6 @@DIV16	A31F @@DIV8	A30A
@@DODIR	9F05 @@DSP	9C1C @@DSPLY	9CBC
@@ERROR	9E5D @@EXIT	9E1E @@FEXT	A20E
@@FLAGS	A39E @@FNAME	A223 @@FSPEC	A1F9
@@GATRD	A28C @@GATWR	A2CB @@GET	9C30
@@GTDCB	A24D @@GTDC	A238 @@GTMOD	A262
@@HDFMT	9FD7 @@HEX16	A373 @@HEX8	A35E
@@HEXDEC	A349 @@HIGH\$	A388 @@INIT	A055
@@KBD	9C94 @@KEY	9C08 @@KEYIN	9CA8
@@KLTSK	9EDB @@LOAD	A1CF @@LOC	A0BE
@@LOF	A0D3 @@LOGGER	9CF3 @@LOGOT	9D08
@@MSG	9D3F @@MUL16	A2F5 @@MUL8	A2E0
@@OPEN	A06A @@PARAM	9DB5 @@PAUSE	9DA0
@@PEOF	A0E8 @@POSN	A0FD @@PRINT	9D54
@@PRT	9C6C @@PUT	9C44 @@RAMDIR	9F1A
@@RDSEC	9FAD @@RDSSC	A277 @@READ	A112
@@REMOV	A040 @@RENAM	A02B @@REW	A127
@@RMTSK	9EB1 @@RPTSK	9EC6 @@RREAD	A13C
@@RSLCT	9F98 @@RSTOR	9F59 @@RUN	A1E4
@@RWRIT	A151 @@SEEK	9F83 @@SEEKSC	A166
@@SKIP	A17B @@SLCT	9F44 @@STEPI	9F6E
@@TIME	9DDF @@VDCTL	9D8B @@VER	A190
@@VRSEC	9FC2 @@WEOF	A1A5 @@WHERE	9C80
@@WRITE	A1BA @@WRSEC	9FEC @@WRSSC	A001
@@WRTRK	A016		

2403 is the transfer address
00000 Total errors

NOTES:

Command: DEBUG, VERIFY

Library: SYS7/SYS

ISAM # : 14H, 1BH

```

00100 ;LBDEBUG/ASM - DEBUG VERIFY Commands
00000 00110 TITLE <DEBUG/VERIFY - LS-DOS 6.2>
00120 ;
0000D 00130 CR EQU 13
00000 00140 *GET SVCMAC:3 ;SVC Macro equivalents
00010 ;SVC MAC/ASM - LS-DOS Version VI
00020 *LIST OFF
03900 *LIST ON
00150 *LIST OFF ;Get SYS0/EQU
00170 *LIST ON
00180 ;
2400 00190 ORG 2400H
00200 ;
2400 C37824 00210 JP VERIFY ;Verify entry point
00220 ;
00230 ; Debug entry
00240 ;
2403 11F124 00250 DEBUG LD DE,PRM1TBL ;Pt to y/n on/off parms
2406 00260 @PARAM ;Get parameters
2406 3E11 00001 LD A,17
2408 EF 00002 RST 40
2409 C2C324 00270 JP NZ,IOERR ;Jump on parm error
240C CDB624 00280 CALL NPARM ;Ck on yes or no entry
240F 2821 00290 JR Z,DBG2 ;Go if no (OFF)
2411 3A7324 00300 LD A,(EPARM) ;Ck on extended DEBUG
2414 B7 00310 OR A ; requested
2415 280A 00320 JR Z,DBG0 ;Go if not
2417 FDCB0246 00330 BIT 0,(IY+'C'-'A') ;Can't put extended DEBUG
241B C2CD24 00340 JP NZ,CANT ; if himem not available
241E CD7524 00350 CALL DBG3 ;Call if EXT entered
2421 FDCB12FE 00360 DBG0 SET 7,(IY+'S'-'A') ;Turn on DEBUG flag
2425 1608 00370 LD D,8 ;Set to mask bit 3
2427 CD9824 00380 CALL GET$D0
242A AF 00390 DBG0A XOR A ; & enable DEBUG
242B 329F19 00400 DBG1 LD (@DBGHK),A
242E 210000 00410 EXIT LD HL,0 ;Set no error
2431 C9 00420 RET
00430 ;
2432 FDCB12BE 00440 DBG2 RES 7,(IY+'S'-'A') ;Turn off DEBUG flag
2436 1680 00450 LD D,80H ;Set to mask bit 7
2438 CD9824 00460 CALL GET$D0
243B 3A7324 00470 DBG2A LD A,(EPARM) ;Ck on extended DEBUG
243E B7 00480 OR A ; requested
243F 282E 00490 JR Z,DBG4 ;Go if not
00500 ;
00510 ; User requested we remove extended DEBUG
00520 ;
2441 FDCB0246 00530 BIT 0,(IY+'C'-'A') ;Can we alter HIGH$?
2445 2028 00540 JR NZ,DBG4 ;Go if not
2447 ED5BA419 00550 LD DE,(EXTDBG$) ;Is extended resident?
244B 2124EB 00560 LD HL,-ORARET@ ;Ck if vector same as
244E ED5A 00570 ADC HL,DE ; the default
2450 281D 00580 JR Z,DBG4 ;Go if not resident
2452 210000 00590 LD HL,0 ;Set to get HIGH$
2455 45 00600 LD B,L
2456 00610 @@HIGH$ ;P/u current pointer
2456 3E64 00003 LD A,100
2458 EF 00004 RST 40
2459 23 00620 INC HL ;Point to assumed module
245A AF 00630 XOR A ; & check if extended
245B ED52 00640 SBC HL,DE ; debugger is lowest

```

```

245D 2010      00650      JR      NZ,DBG4      ;Can't release if not
245F 13      00660      INC      DE      ;Point to last byte used
2460 13      00670      INC      DE
2461 EB      00680      EX      DE,HL      ;This will be new
2462 5E      00690      LD      E,(HL)
2463 23      00700      INC      HL      ; HIGH$
2464 56      00710      LD      D,(HL)
2465 EB      00720      EX      DE,HL      ;New high to HL
2466      00730      @@HIGH$      ;Reset high$
2466 3E64      00005      LD      A,100
2468 EF      00006      RST      40
2469 21DC14      00740      LD      HL,ORARET0      ;Reset extended debugger
246C 22A419      00750      LD      (EXTDBG$),HL      ; vector to default
246F 3EC9      00760      DBG4      LD      A,0C9H      ;Turn off DEBUG vector
2471 18B8      00770      JR      DBG1
             00780 ;
2473 0000      00790      EPARM      DW      0
2475 3E9B      00800      DBG3      LD      A,9BH      ;Get SYS9 loaded & up
2477 EF      00810      RST      28H
             00820 ;
             00830 ;      Verify entry
             00840 ;
2478 110125      00850      VERIFY      LD      DE,PRM2TBL      ;Pt to ext + above
247B      00860      @@PARAM
247B 3E11      00007      LD      A,17
247D EF      00008      RST      40
247E 2043      00870      JR      NZ,IOERR      ;Quit on parm error
2480 CDB624      00880      CALL      NPARAM      ;Ck yes/no
2483 213115      00890      LD      HL,@WRITE      ;Init to vector WRITE
2486 FDCB0396      00900      RES      2,(IY+'D'-'A')      ;Indicate VERIFY=OFF
248A 2807      00910      JR      Z,VER1      ;Use WRITE if VER=(OFF)
248C 216015      00920      LD      HL,@VER      ;Init to vector VERIFY
248F FDCB03D6      00930      SET      2,(IY+'D'-'A')      ;Indicate VERIFY=ON
2493 229601      00940      VER1      LD      (75*2+SVCTAB$),HL      ;Change SVCTAB vector
2496 1896      00950      JR      EXIT
             00960 ;
             00970 ;      Routine to save or restore SPACE COMPRESSION state
             00980 ;
2498 D5      00990      GET$D0      PUSH      DE      ;Save mask value
2499 11B224      01000      LD      DE,$D0$
249C      01010      @@GTMOD
249C 3E53      00009      LD      A,83
249E EF      00010      RST      40
249F 210400      01020      LD      HL,4      ;Point to 1st data byte
24A2 19      01030      ADD      HL,DE
24A3 D1      01040      POP      DE      ;Rcvr mask value
24A4 C0      01050      RET      NZ      ;Return if not found
24A5 7E      01060      LD      A,(HL)      ;P/u existing state
24A6 A2      01070      AND      D      ;Strip state of space comp
24A7 07      01080      RLCA      ; & move to bit 7 or 3
24A8 07      01090      RLCA
24A9 07      01100      RLCA
24AA 07      01110      RLCA
24AB 5F      01120      LD      E,A      ;Tempy save
24AC 7E      01130      LD      A,(HL)
24AD E677      01140      AND      77H      ;Strip bits 3 and 7
24AF B3      01150      OR      E      ;Merge comp state
24B0 77      01160      LD      (HL),A
24B1 C9      01170      RET
             01180 ;
24B2 24      01190      $D0$      DB      '$D0',3

```

```

44 4F 03      01200 ;
               01210 ;
               01220 ;
               01230 NPARM LD BC,0 ;P/u "no" parm entry
24B6 010000    01240 @@FLAGS ;Point to flag table
24B9           00011 LD A,101
24BB EF        00012 RST 40
24BC 78        01250 LD A,B
24BD B1        01260 OR C ;If NO, make Z
24BE EEFF      01270 XOR 0FFH
24C0 C9        01280 RET
24C1 0000      01290 YPARM DW 0
               01300 ;
               01310 ;
               01320 ;
               01330 IOERR LD L,A ;Set HL = error code
24C3 6F        01340 LD H,0 ;
24C4 2600      01350 OR 0C0H ;Short error
24C6 F6C0      01360 LD C,A ;Xfer error # to C
24C8 4F        01370 @@ERROR ;Display error
24C9           00013 LD A,26
24CB EF        00014 RST 40
24CC C9        01380 RET ;Return to DOS
               01390 ;
               01400 ;
               01410 ;
               01420 CANT LD HL,CANT$
24CD 21D724    01430 @@LOGOT
24D0           00015 IFEQ 00H,1
               00016 LD HL,
               00017 ENDIF
24D0 3E0C      00018 LD A,12
24D2 EF        00019 RST 40
24D3 21FFFF    01440 LD HL,-1
24D6 C9        01450 RET
               01460 ;
24D7 4E        01470 CANT$ DB 'No memory space available',CR
        6F 20 6D 65 6D 6F 72 79
        20 73 70 61 63 65 20 61
        76 61 69 6C 61 62 6C 65
        0D
               01480 ;
24F1 45        01490 PRMITBL DB 'EXT '
        58 54 20 20 20
24F7 7324      01500 DW EPARM
24F9 45        01510 DB 'E '
        20 20 20 20 20
24FF 7324      01520 DW EPARM
2501 4F        01530 PRM2TBL DB 'ON '
        4E 20 20 20 20
2507 C124      01540 DW YPARM
2509 4F        01550 DB 'OFF '
        46 46 20 20 20
250F B724      01560 DW NPARM+1
2511 59        01570 DB 'Y '
        20 20 20 20 20
2517 C124      01580 DW YPARM
2519 4E        01590 DB 'N '
        20 20 20 20 20
251F B724      01600 DW NPARM+1

```

2521 00	01610	NOP	
	01620 ;		
2403	01630	END	DEBUG

\$A1	03B7	\$A2	03B8	\$A3	03B9
\$CKEOF	1470	\$DOS	24B2	@\$SYS	08F0
@@1	0000	@@2	0000	@@3	0000
@@4	0000	@ABORT	1B08	@ADTSK	1CDA
@BANK	0877	@BKSP	1486	@BREAK	196F
@BYTE IO	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
@GTDCEB	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
@LOGGER	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
@RWRTIT	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	CANT	24CD	CANT\$	24D7
CASHK\$	0A7B	CFCB\$	00E0	CFGFCB\$	00E0
CFLAG\$	006C	CKMOD@	1A7F	CKOPEN@	1568
CONF IG\$	203F	CORE\$	0300	CR	000D
CRTBGN\$	F800	CYL GRN	16AE	D@F8BYT8	1A26
DATE\$	0033	DAYTBL\$	04C7	DBG0	2421
DBG0A	242A	DBG1	242B	DBG2	2432
DBG2A	243B	DBG3	2475	DBG4	246F
DBGSV\$	00A0	DCBKL\$	0031	DCT\$	0470
DCTBYT8@	1A29	DCTFLD@	1A34	DEBUG	2403
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
DSKTP\$	04C0 DTPMT\$	04C2 DVREND\$	0FF4
DVRHI\$	0206 EFLAG\$	006E ENADIS DO_RAM	0817
EPARM	2473 EXIT	242E EXTDBG\$	19A4
FDDINT\$	000E FEMSK\$	006F FLGTAB\$	006A
GET\$DO	2498 GET @ ROWCOL	0DAE HERTZ\$	0750
HIGH\$	040E HKRES\$	1A6C IFLAG\$	0072
INBUF\$	0420 INTIM\$	003C INTMSK\$	003D
INTVC\$	003E IOERR	24C3 JCLCB\$	0203
JDCB\$	0024 JFCB\$	00C0 JLDCB\$	0230
JRET\$	0026 KCK0	07D6 KFLAG\$	0074
KIDATA\$	08FC KIDCB\$	0208 LBANK\$	0202
LDRV\$	0023 LFLAG\$	0075 LNKFCB0	1566
LOW\$	001E L SVC\$	000D MAXCOR\$	2400
MAXDAY\$	0401 MINCOR\$	3000 MODOUT\$	0076
MONTBL\$	04DC NFLAG\$	0077 NPARM	24B6
OPREG\$	0078 OPREG SV_AREA	086E OPREG SV_PTR	0835
ORARET0	14DC OSRLS\$	003B OSVER\$	0085
OVRLY\$	0069 PAKNAM\$	0410 PAUSE0	0382
PCSAVE\$	07AF PDRV\$	001B PHIGH\$	001C
PRDCB\$	0218 PRM1TBL	24F1 PRM2TBL	2501
PUTA0DE	0DCD PUT @	0DCA PUT @ ROWCOL	0DC6
RFLAG\$	007B ROWCOL 2_ADDR	0DD0 RST380	1BFF
RSTOR\$	04C4 RWRITE	13A2 SIDCB\$	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
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
VER1	2493 VERIFY	2478 VFLAG\$	007F
WRINT\$	0080 YPARM	24C1 ZERO\$	0401
ZEROA0	13A0 @@ABORT	798E @@ADTSK	7A21
@@BANK	7F39 @@BKSP	7C19 @@BREAK	7F4F
@@CHNIO	7979 @@CKBRKC	7F9D @@CKDRV	7A75
@@CKEOF	7C2E @@CKTSK	7A0C @@CLOSE	7C04
@@CLS	7F87 @@CMNDI	79B8 @@CMNDR	79CD
@@CTL	77DD @@DATE	794F @@DCSTAT	7AB4
@@DEBUG	79F7 @@DECHEX	7EB9 @@DIRRD	7E26
@@DIRWR	7E3B @@DIV16	7EA4 @@DIV8	7E8F
@@DODIR	7A8A @@DSP	77A1 @@DSPLY	7841
@@ERROR	79E2 @@EXIT	79A3 @@FEXT	7D93
@@FLAGS	7F23 @@FNAME	7DA8 @@FSPEC	7D7E
@@GATRD	7E11 @@GATWR	7E50 @@GET	77B5
@@GTDCB	7DD2 @@GTDCI	7DBD @@GTMOD	7DE7
@@HDFMT	7B5C @@HEX16	7EF8 @@HEX8	7EE3
@@HEXDEC	7ECE @@HIGH\$	7F0D @@INIT	7BDA
@@KBD	7819 @@KEY	778D @@KEYIN	782D
@@KLTSK	7A60 @@LOAD	7D54 @@LOC	7C43
@@LOF	7C58 @@LOGGER	7878 @@LOGOT	788D
@@MSG	78C4 @@MUL16	7E7A @@MUL8	7E65
@@OPEN	7BEF @@PARAM	793A @@PAUSE	7925
@@PEOF	7C6D @@POSN	7C82 @@PRINT	78D9
@@PRT	77F1 @@PUT	77C9 @@RAMDIR	7A9F
@@RDSEC	7B32 @@RDSSC	7DFC @@READ	7C97
@@REMOV	7BC5 @@RENAM	7BB0 @@REW	7CAC

@@RMTSK	7A36 @@RPTSK	7A4B @@RREAD	7CC1
@@RSLCT	7B1D @@RSTOR	7ADE @@RUN	7D69
@@RWRTIT	7CD6 @@SEEK	7B08 @@SEEKSC	7CEB
@@SKIP	7D00 @@SLCT	7AC9 @@STPI	7AF3
@@TIME	7964 @@VDCTL	7910 @@VER	7D15
@@VRSEC	7B47 @@WEOF	7D2A @@WHERE	7805
@@WRITE	7D3F @@WRSEC	7B71 @@WRSSC	7B86
@@WRTRK	7B9B		

2403 is the transfer address

00000 Total errors

NOTES:

NOTES:

Command: DEVICE

Library: SYS6/SYS

ISAM # : 61H

```

00100 ;LBDEVICE/ASM - DEVICE Command
0000 00110 TITLE <DEVICE - LS-DOS 6.2>
00120 ;
000A 00130 LF EQU 10
000D 00140 CR EQU 13
0000 00150 *GET SVCMAC:3 ;SVC Macro equivalents
00010 ;SVC MAC/ASM - LS-DOS Version VI
00020 *LIST OFF
03900 *LIST ON
00160 ;
2400 00170 ORG 2400H
00180 ;
00190 DEVICE
2400 00200 @@CKBRKC ;Check for break
2400 3E6A 00001 LD A,106
2402 EF 00002 RST 40
2403 2804 00210 JR Z,DEVICEA ;Continue if not
2405 21FFFF 00220 LD HL,-1 ; else abort
2408 C9 00230 RET
00240 ;
2409 ED73CD27 00250 DEVICEA LD (SAVE SP+1),SP ;Save stack pointer
240D 11B928 00260 LD DE,PRMTBL$ ;First check for user parms
2410 00270 @@PARAM
2410 3E11 00003 LD A,17
2412 EF 00004 RST 40
2413 C29527 00280 JP NZ,IOERR ;Go if parm error
2416 00290 @@FLAGS ;Get flag table pointer
2416 3E65 00005 LD A,101
2418 EF 00006 RST 40
2419 CDBD27 00300 CALL RESKFL ;Reset Pause and Enter
241C FB 00310 EI ;Make sure they're on
241D 21FFFF 00320 DPARM LD HL,-1 ;Check Drive parameter
2420 7C 00330 LD A,H
2421 B5 00340 OR L
2422 CA7F25 00350 JP Z,DEND ;Go if D=NO
2425 0E00 00360 LD C,0 ;Init to drive 0
2427 C5 00370 DEV1 PUSH BC ;Save drive #
2428 AF 00380 XOR A ;Reset flag stuff
2429 327C24 00390 LD (WPTEST+1),A ; location
242C 00400 @@GTDCT ;Get DCT address
242C 3E51 00007 LD A,81
242E EF 00008 RST 40
242F FD7E00 00410 LD A,(IY+0) ;Is this drive disabled?
2432 FEC3 00420 CP 0C3H
2434 C27725 00430 JP NZ,POPDRV ;Ignore if it is
2437 00440 @@CKDRV ;This drive available?
2437 3E21 00009 LD A,33
2439 EF 00010 RST 40
243A 2022 00450 JR NZ,DEV2 ;Go if no diskette
243C 1F 00460 RRA ;Shift C-flag to bit-7
243D 327C24 00470 LD (WPTEST+1),A ; & save for WP test
2440 210029 00480 LD HL,BUFFER ;Pick up the GAT for the
2443 FD5609 00490 LD D,(IY+9) ; pack name
2446 5D 00500 LD E,L
2447 00510 @@RDSSC
2447 3E55 00011 LD A,85
2449 EF 00012 RST 40
244A 3E14 00520 LD A,20 ;"GAT read error
244C C29527 00530 JP NZ,IOERR
244F 21D829 00540 LD HL,BUFFER+0D8H ;Shove bracket ETX
2452 365D 00550 LD (HL),' '

```

The Source	LIBRARY Files	DEVICE - LS-DOS 6.2	Page 00002
2454 2C	00560	INC L	
2455 3620	00570	LD (HL),' '	
2457 2C	00580	INC L	
2458 3603	00590	LD (HL),3	
245A 2ED0	00600	LD L,0D0H	;Point to start of name
245C 180B	00610	JR DEV2A	
	00620 ;		
	00630 ;		
	00640 ;	Drive info for this active drive	
245E 21D827	00650 DEV2	LD HL,NOPACK\$;Display pack name
2461 11D029	00660	LD DE,BUFFER+0D0H	
2464 010C00	00670	LD BC,12	
2467 EDB0	00680	LDIR	
2469 3E3A	00690 DEV2A	LD A,':'	;Output the colon
246B CD8227	00700	CALL BYTOUT	
246E C1	00710	POP BC	;Get drive # back
246F C5	00720	PUSH BC	
2470 79	00730	LD A,C	;Get drive # converted
2471 C630	00740	ADD A,'0'	; to ASCII & display it
2473 CD8227	00750	CALL BYTOUT	
2476 3E20	00760	LD A,' '	;Space out one
2478 CD8227	00770	CALL BYTOUT	
247B 3E00	00780 WPTST	LD A,0	;P/u CKDRV FDC status
247D 07	00790	RLCA	;Hardware write protect?
247E 380A	00800	JR C,DEV2B	;Force "WP" if it is
2480 FDCB037E	00810	BIT 7,(IY+3)	;Test software WP
2484 3E20	00820	LD A,' '	;Output ' ' for read &
2486 0620	00830	LD B,' '	; write access or
2488 2804	00840	JR Z,\$+6	
248A 3E57	00850 DEV2B	LD A,'W'	; WP for read only
248C 0650	00860	LD B,'P'	
248E CD8227	00870	CALL BYTOUT	
2491 78	00880	LD A,B	;Xfer the 2nd char
2492 CD8227	00890	CALL BYTOUT	; & display it
2495 3E20	00900	LD A,' '	
2497 CD8227	00910	CALL BYTOUT	
249A 3E5B	00920	LD A,'['	;Left bracket
249C CD8227	00930	CALL BYTOUT	
249F 21D029	00940	LD HL,BUFFER+0D0H	;Write the pack name
24A2 CD7227	00950	CALL LINOUT	
	00960 ;		
	00970 ;	Determine if 5" or 8"	
	00980 ;		
24A5 FDCB036E	00990	BIT 5,(IY+3)	;Test 5"/8" drive
24A9 3E35	01000	LD A,'5'	;Init to 5
24AB 2802	01010	JR Z,\$+4	;Bypass if not 8
24AD 3E38	01020	LD A,'8'	; else init to 8
24AF CD8227	01030	CALL BYTOUT	
24B2 FDCB035E	01040	BIT 3,(IY+3)	;Test rigid/floppy
24B6 216028	01050	LD HL,FLOPY\$;Init to floppy
24B9 2803	01060	JR Z,\$+5	;Bypass if that kind
24BB 216B28	01070	LD HL,RIGID\$; else is hard
24BE CD7227	01080	CALL LINOUT	
24C1 FD7E04	01090	LD A,(IY+4)	;Output drive select addr
24C4 E60F	01100	AND 0FH	; in ASCII
24C6 C690	01110	ADD A,90H	
24C8 27	01120	DAA	
24C9 CE40	01130	ADC A,40H	
24CB 27	01140	DAA	
24CC CD8227	01150	CALL BYTOUT	
24CF FD6E06	01160 DEV3	LD L,(IY+6)	;P/u highest cylinder


```

2557 CD7227 01780 CALL LINOUT
255A FDCB0356 01790 BIT 2,(IY+3) ;Test off/on
255E 3E20 01800 LD A,' ' ;1 sec if DELAY=ON
2560 0631 01810 LD B,'1'
2562 2804 01820 JR Z,$+6
2564 3E2E 01830 LD A,'.' ;0.5 sec if DELAY=OFF
2566 0635 01840 LD B,'5'
2568 CD8227 01850 CALL BYTOUT
256B 78 01860 LD A,B
256C CD8227 01870 CALL BYTOUT
256F 3E73 01880 LD A,'s' ;Indicate seconds
2571 CD8227 01890 CALL BYTOUT
2574 CDA027 01900 ENDLINE CALL CKPAWS ;Check pause of display
2577 C1 01910 POPDRV POP BC ;Recover drive #
2578 0C 01920 INC C ;Bump to next drive
2579 79 01930 LD A,C
257A FE08 01940 CP 8 ;Loop thru all 8
257C C22724 01950 JP NZ,DEV1
257F 01960 DEND EQU $
01970 ;
01980 ; Byte I/O devices
01990 ;
257F 210000 02000 BPARAM LD HL,$-$ ;Check B parameter
2582 7C 02010 LD A,H
2583 B5 02020 OR L
2584 CAC826 02030 JP Z,BEND ;Go if B=NO (default)
02040 ;
02050 ; Display the device vectoring
02060 ;
2587 114B49 02070 LD DE,'IK' ;Start of device tables
258A 02080 @@GTDCB
258A 3E52 00013 LD A,82
258C EF 00014 RST 40
258D C29527 02090 JP NZ,IOERR
2590 7E 02100 LOGDCB LD A,(HL) ;Bypass this device if
2591 B7 02110 OR A ; table shows spare
2592 CA4A26 02120 JP Z,DVRB2
2595 11002A 02130 LD DE,STRBUF ;Pt to string buffer
2598 E5 02140 PUSH HL ;Save origin ptr
2599 CDB826 02150 CALL MOVNAM ;Move dev name -> strbuf
259C E1 02160 POP HL ;Rcvr org of table
259D E5 02170 PUSH HL
259E CB5E 02180 LOGDCB1 BIT 3,(HL) ;If NIL, don't show
25A0 201C 02190 JR NZ,DVRADDR ; any routes
25A2 CB66 02200 BIT 4,(HL) ;Is device routed?
25A4 2818 02210 JR Z,DVRADDR ;Bypass if not
02220 ;
02230 ; This device is routed
02240 ;
25A6 2C 02250 LOGRTE INC L ;Pt to vector & get it
25A7 7E 02260 LD A,(HL)
25A8 2C 02270 INC L
25A9 66 02280 LD H,(HL)
25AA 6F 02290 LD L,A
25AB CB7E 02300 BIT 7,(HL) ;Is the route to a file?
25AD C25426 02310 JP NZ,RTEFCB ;Jump if a file
25B0 E5 02320 PUSH HL ;Hang onto this vector
25B1 CD8226 02330 CALL DCBDIR ;Get device direction
25B4 CDB826 02340 CALL MOVNAM ;Move dev name -> strbuf
25B7 E1 02350 POP HL ;Rcvr org of routee
25B8 CB66 02360 BIT 4,(HL) ;Is routee also routed?

```


The Source	LIBRARY Files	DEVICE - LS-DOS 6.2	Page 00005
25BA 20EA	02370	JR NZ,LOGRTE	;Loop de loop if yes
25BC 187F	02380	JR DVRB1	; else go display the line
	02390 ;		
	02400 ;		
	02410 ;	Device has no routes - show its driver address	
25BE CD8226	02420	DVRADDR CALL DCBDIR	;Get device direction
25C1 CB5E	02430	BIT 3,(HL)	;Is this a NIL device
25C3 C27826	02440	JP NZ,MOVNIL	;No address if NIL
	02450 ;		
	02460 ;	If linked, show device name of link	
	02470 ;		
25C6 CB6E	02480	BIT 5,(HL)	;Any link DCB?
25C8 2824	02490	JR Z,DVRA0	;Go if none
25CA 2C	02500	INC L	;Get address of link DCB
25CB 7E	02510	LD A,(HL)	
25CC 2C	02520	INC L	
25CD 66	02530	LD H,(HL)	
25CE 6F	02540	LD L,A	
	02550 ;		
	02560 ;	Now move in the name of the linked DCB	
	02570 ;		
25CF E5	02580	PUSH HL	
25D0 E5	02590	PUSH HL	
25D1 CDB826	02600	CALL MOVNAM	;Move name of LINK DCB
25D4 3E7C	02610	LD A,' '	;Get separator for display and
25D6 12	02620	LD (DE),A	; put in the buffer
25D7 13	02630	INC DE	
25D8 FDE1	02640	POP IY	;Pop address to IY
25DA FD6E04	02650	LD L,(IY+4)	;P/u linked DCB address
25DD FD6605	02660	LD H,(IY+5)	
25E0 CDB826	02670	CALL MOVNAM	;Move name of linked DCB
25E3 E1	02680	POP HL	;Recover address
25E4 EB	02690	EX DE,HL	;Switch tempy, HL to
25E5 3620	02700	LD (HL),' '	; display buffer
25E7 23	02710	INC HL	
25E8 3626	02720	LD (HL),'&'	;Show the link
25EA 23	02730	INC HL	
25EB EB	02740	EX DE,HL	;Back to normal
25EC 18B0	02750	JR LOGDCB1	;Go ck this one
	02760 ;		
	02770 ;	If filtered, find the filter DCB	
	02780 ;		
25EE CB76	02790	DVRA0 BIT 6,(HL)	;If filtered, recover the
25F0 2835	02800	JR Z,DVRA0	; original data by
25F2 E5	02810	PUSH HL	; swapping back the
25F3 3E5B	02820	LD A,'['	
25F5 12	02830	LD (DE),A	
25F6 13	02840	INC DE	
25F7 D5	02850	PUSH DE	
25F8 54	02860	LD D,H	
25F9 5D	02870	LD E,L	
25FA 2C	02880	INC L	; 1st three bytes with
25FB 7E	02890	LD A,(HL)	; the FILTER DCB
25FC 2C	02900	INC L	
25FD 66	02910	LD H,(HL)	
25FE 6F	02920	LD L,A	
25FF 010400	02930	LD BC,4	;HL now points to the
2602 09	02940	ADD HL,BC	; entry point. Get its
2603 4E	02950	LD C,(HL)	; DCB address by peeking
2604 0C	02960	INC C	; past the name field
2605 09	02970	ADD HL,BC	

The Source	LIBRARY Files	DEVICE - LS-DOS 6.2	Page 00006
2606 7E	02980	LD A,(HL)	;Get low-order
2607 23	02990	INC HL	
2608 66	03000	LD H,(HL)	;Get hi-order
2609 6F	03010	LD L,A	
260A E5	03020	PUSH HL	;If DCB is itself, then
260B ED52	03030	SBC HL,DE	; bring in the "inactive
260D E1	03040	POP HL	
260E D1	03050	POP DE	;Recover string buf ptr
260F 200A	03060	JR NZ,DVRA1	
2611 21E327	03070	LD HL,INACT\$	
2614 010800	03080	LD BC,8	
2617 EDB0	03090	LDIR	
2619 1803	03100	JR DVRA2	
	03110 ;		
261B CDB826	03120 DVRA1	CALL MOVNAM	;Move name of filter DCB
261E 3E5D	03130 DVRA2	LD A,']'	;Put dsp chars into buffer
2620 12	03140	LD (DE),A	
2621 13	03150	INC DE	
2622 3E20	03160	LD A,' '	
2624 12	03170	LD (DE),A	
2625 13	03180	INC DE	
2626 E1	03190	POP HL	;Recover orig DCB ptr
	03200 ;		
	03210 ;		
	03220 ;		
		Routine to construct address "X'xxxx'"	
2627 3E58	03230 DVRB0	LD A,'X'	;Show address as
2629 12	03240	LD (DE),A	; X'dddd'
262A 13	03250	INC DE	
262B 3E27	03260	LD A,27H	;Single quote
262D 12	03270	LD (DE),A	
262E 13	03280	INC DE	
262F 2C	03290	INC L	
2630 7E	03300	LD A,(HL)	;P/u lo-order vector
2631 2C	03310	INC L	
2632 66	03320	LD H,(HL)	;P/u hi-order vector
2633 6F	03330	LD L,A	;Put lo in place
2634 EB	03340	EX DE,HL	;Vector value to DE
2635	03350	@@HEX16	;Convert to hex digits
2635 3E63	00015	LD A,99	
2637 EF	00016	RST 40	
2638 EB	03360	EX DE,HL	;Restore strbuf ptr to DE
2639 3E27	03370	LD A,27H	;Closing '
263B 12	03380	LD (DE),A	
263C 13	03390	INC DE	
263D 3E0D	03400 DVRB1	LD A,CR	
263F 12	03410	LD (DE),A	;Stuff end-of-line
2640 21002A	03420	LD HL,STRBUF	;Display the info
2643 CD7227	03430	CALL LINOUT	
2646 CDA527	03440	CALL CKPAWS0	;Ck with no CR
2649 E1	03450	POP HL	;Rcvr table org
264A 7D	03460 DVRB2	LD A,L	;Advance to next table
264B C608	03470 TABLEN	ADD A,8	
264D 6F	03480	LD L,A	
264E DAC826	03490	JP C,SPARM	;Exit if finished
2651 C39025	03500	JP LOGDCB	; else loop
	03510 ;		
	03520 ;		
	03530 ;		
		Device routed to a file - grab its filespec	
2654 E5	03540 RTEFCB	PUSH HL	;Save control block org
2655 21B428	03550	LD HL,IO\$;Show 2-way device
2658 010500	03560	LD BC,5	

```

265B EDB0 03570 LDIR
265D E1 03580 POP HL
265E 7D 03590 LD A,L ;Pt to file route data
265F C606 03600 ADD A,6 ; by indexing into FCB
2661 6F 03610 LD L,A
2662 8C 03620 ADC A,H
2663 95 03630 SUB L
2664 67 03640 LD H,A ;HL = FCB+6
2665 4E 03650 LD C,(HL) ;P/u drive #
2666 23 03660 INC HL
2667 46 03670 LD B,(HL) ;P/u DEC
2668 D5 03680 PUSH DE
2669 03690 @@FNAME ;Fetch filename
2669 3E50 00017 LD A,80
266B EF 00018 RST 40
266C D1 03700 POP DE
266D C29527 03710 JP NZ,IOERR
2670 1A 03720 RTEF1 LD A,(DE) ;Find end of filename
2671 FE03 03730 CP 3
2673 28C8 03740 JR Z,DVRB1 ;Exit on ETX to put CR
2675 13 03750 INC DE
2676 18F8 03760 JR RTEF1
03770 ;
03780 ; Move in 'NIL' as driver address
03790 ;
2678 21B128 03800 MOVNIL LD HL,NIL$ ;Move in NIL
267B 010300 03810 LD BC,3
267E EDB0 03820 LDIR
2680 18BB 03830 JR DVRB1
03840 ;
03850 ; Routine to denote i/o direction
03860 ;
2682 3E20 03870 DCBDIR LD A,' ' ;1st need a space
2684 12 03880 LD (DE),A
2685 13 03890 INC DE
2686 CB46 03900 BIT 0,(HL) ;Test if input device
2688 2802 03910 JR Z,DCBD1 ;Put another space if not
268A 3E3C 03920 LD A,'<' ;Else show input capable
268C 12 03930 DCBD1 LD (DE),A
268D 13 03940 INC DE
268E 3E3D 03950 LD A,'=' ;Always need this
2690 CB76 03960 BIT 6,(HL) ;If a filter, then
2692 2802 03970 JR Z,$+4 ; reset to '#'
2694 3E23 03980 LD A,'#'
2696 12 03990 LD (DE),A
2697 13 04000 INC DE
2698 3E20 04010 LD A,' ' ;Init a space
269A CB4E 04020 BIT 1,(HL) ;Output device?
269C 2802 04030 JR Z,DCBD2 ;Use space if not
269E 3E3E 04040 LD A,'>' ;Else show output capable
26A0 12 04050 DCBD2 LD (DE),A
26A1 13 04060 INC DE
26A2 3E20 04070 LD A,' ' ;Close with a space
26A4 12 04080 LD (DE),A
26A5 13 04090 INC DE
26A6 C9 04100 RET
04110 ;
04120 ; Convert HL to 3-place decimal & stuff into (DE)
04130 ;
26A7 D5 04140 CVRTDEC PUSH DE ;Save place
26A8 110029 04150 LD DE,BUFFER

```

The Source	LIBRARY Files	DEVICE - LS-DOS 6.2	Page 00008
26AB	04160	@@HEXDEC	;Convert to decimal ASCII
26AB 3E61	00019	LD A,97	
26AD EF	00020	RST 40	
26AE 210229	04170	LD HL,BUFFER+2	;Skip leading spaces
26B1 D1	04180	POP DE	
26B2 010300	04190	LD BC,3	
26B5 EDB0	04200	LDIR	
26B7 C9	04210	RET	
	04220 ;		
	04230 ;	Move device name into string buffer	
	04240 ;		
26B8 7D	04250 MOVNAM	LD A,L	;Pt to name field
26B9 C606	04260	ADD A,6	
26BB 6F	04270	LD L,A	
26BC 3E2A	04280	LD A,'*'	;Stuff * in string buf
26BE 12	04290	LD (DE),A	
26BF 13	04300	INC DE	;Bump ptr to next pos
26C0 7E	04310	LD A,(HL)	;P/u 1st char of name
26C1 12	04320	LD (DE),A	; & stuff it
26C2 13	04330	INC DE	;Do the same for char 2
26C3 2C	04340	INC L	
26C4 7E	04350	LD A,(HL)	
26C5 12	04360	LD (DE),A	
26C6 13	04370	INC DE	
26C7 C9	04380	RET	
26C8	04390 BEND	EQU \$	
	04400 ;		
	04410 ;	Show high memory device drivers	
	04420 ;		
26C8 21FFFF	04430 SPARM	LD HL,-1	;Check S parameter
26CB 7C	04440	LD A,H	
26CC B5	04450	OR L	
26CD CAD327	04460	JP Z,EXIT	;Exit if through
26D0 21EB27	04470	LD HL,DVCHDR\$;Display header
26D3 CD7227	04480	CALL LINOUT	
26D6	04490	@@FLAGS	;Get flag table pointer
26D6 3E65	00021	LD A,101	
26D8 EF	00022	RST 40	
26D9 FD7E03	04500	LD A,(IY+'D'-'A')	;P/u device flag
26DC B7	04510	OR A	;Exit if none in use
26DD F5	04520	PUSH AF	;Save flag
26DE 282D	04530	JR Z,SHOWFS	;Go if nothing on
26E0 21F527	04540	LD HL,DVCS\$;Pt to word string
26E3 01FF08	04550	LD BC,8<8!0FFH	;Init for 8 flag bits
26E6 F1	04560 DOD1	POP AF	;Rcvr link
26E7 0F	04570	RRCA	;Test if active
26E8 F5	04580	PUSH AF	
26E9 301B	04590	JR NC,DOD3	;Bypass if inactive
26EB 0C	04600	INC C	;Do we do the comma?
26EC 3E2C	04610	LD A','	;End of word, do comma
26EE C48227	04620	CALL NZ,BYTOUT	
26F1 3E20	04630	LD A,' '	;Start with a space
26F3 CD8227	04640	CALL BYTOUT	
26F6 7E	04650 DOD2	LD A,(HL)	;Display word until carry
26F7 23	04660	INC HL	
26F8 F5	04670	PUSH AF	
26F9 E67F	04680	AND 7FH	;Strip possible carry
26FB CD8227	04690	CALL BYTOUT	;Display the char
26FE F1	04700	POP AF	
26FF 07	04710	RLCA	;Was carry set
2700 30F4	04720	JR NC,DOD2	;Loop if not

```

2702 10E2      04730      DJNZ      DOD1      ;Loop for 8 bits
2704 1807      04740      JR          SHOWFS    ;Exit the loop
2706 7E        04750      DOD3      LD          A,(HL)      ;Loop & ignore word
2707 23        04760      INC          HL
2708 07        04770      RLCA          ;Carry set on last char
2709 30FB      04780      JR          NC,DOD3
270B 10D9      04790      DJNZ      DOD1      ;Loop for 8 bits
270D FDCB125E 04800      SHOWFS    BIT          3,(IY+'S'-'A') ;Show FAST or SLOW
2711 2005      04810      JR          NZ,FAST
2713 212928    04820      LD          HL,SLOW$      ;Point to slow$
2716 1803      04830      JR          SHOWIT
2718 212228    04840      FAST      LD          HL,FAST$      ;Point to fast$
271B FD7E03    04850      SHOWIT   LD          A,(IY+'D'-'A') ;Check if others shown
271E B7        04860      OR          A
271F 2001      04870      JR          NZ,COMAOK
2721 23        04880      INC          HL      ;Bypass comma
2722 CD7227    04890      COMAOK    CALL         LINOUT
                04900      ;
                04910      ;      Display system modules resident
                04920      ;
2725 F1        04930      DORES     POP          AF      ;Stack integrity
2726 CDA027    04940      NOTON     CALL         CKPAWS
2729 113028    04950      LD          DE,RES$      ;Check if driver resident
272C           04960      @@GTMOD   ; in memory
272C 3E53      00023      LD          A,83
272E EF        00024      RST          40
272F C2D327    04970      JP          NZ,EXIT      ;Done if nothing res'd
2732 210500    04980      LD          HL,5
2735 19        04990      ADD          HL,DE      ;Point to hi-order table
2736 E5        05000      PUSH         HL
2737 213728    05010      LD          HL,SYSRES$   ;Display header
273A CD7227    05020      CALL         LINOUT
273D E1        05030      POP          HL
273E 01FF10    05040      LD          BC,16<8!0FFH ;Init for 16 modules
2741 7E        05050      DORES1    LD          A,(HL)      ;P/u a high-order vector
2742 23        05060      INC          HL      ;Bump pointer to next
2743 23        05070      INC          HL
2744 B7        05080      OR          A      ;Is this module resident?
2745 2824      05090      JR          Z,DORES3    ;Go if not
2747 0C        05100      INC          C
2748 3E2C      05110      LD          A,', '      ;Need comma if 2nd
274A C48227    05120      CALL         NZ,BYTOUT
274D 3E20      05130      LD          A,' '      ;Start with a space
274F CD8227    05140      CALL         BYTOUT
2752 3E10      05150      LD          A,16
2754 90        05160      SUB          B      ;Calculate module #
2755 16FF      05170      LD          D,-1
2757 14        05180      DORES2    INC          D
2758 D60A      05190      SUB          10
275A 30FB      05200      JR          NC,DORES2
275C F5        05210      PUSH         AF      ;Save units place
275D 7A        05220      LD          A,D      ;Test tens place
275E C630      05230      ADD          A,'0'      ; for non-zero
2760 FE30      05240      CP          '0'
2762 C48227    05250      CALL         NZ,BYTOUT   ;Output if non-zero
2765 F1        05260      POP          AF      ;Get units
2766 C63A      05270      ADD          A,'0'+10    ;Adjust to ASCII
2768 CD8227    05280      CALL         BYTOUT
276B 10D4      05290      DORES3    DJNZ      DORES1
276D CDA027    05300      CALL         CKPAWS      ;One last ck for CR
2770 1861      05310      JR          EXIT

```

```

05320 ;
05330 ;      Output display routines
05340 ;
2772 05350 LINOUT @@DSPLY
00025 IFEQ 00H,1
00026 LD HL,
00027 ENDIF
2772 3E0A 00028 LD A,10
2774 EF 00029 RST 40
2775 201E 05360 JR NZ,IOERR
2777 3A8A27 05370 LD A,(PPARM+1) ;Ck P-parm
277A B7 05380 OR A
277B C8 05390 RET Z
277C 05400 @@PRINT ;Also print if needed
00030 IFEQ 00H,1
00031 LD HL,
00032 ENDIF
277C 3E0E 00033 LD A,14
277E EF 00034 RST 40
277F 2014 05410 JR NZ,IOERR
2781 C9 05420 RET
05430 ;
2782 C5 05440 BYTOUT PUSH BC
2783 4F 05450 LD C,A
2784 05460 @@DSP ;Display it
2784 3E02 00035 LD A,2
2786 EF 00036 RST 40
2787 200A 05470 JR NZ,POPBC
2789 110000 05480 PPARM LD DE,0 ;P/u P-parm
278C 7B 05490 LD A,E
278D B2 05500 OR D
278E 2803 05510 JR Z,POPBC
2790 05520 @@PRT ;Print chr if needed
2790 3E06 00037 LD A,6
2792 EF 00038 RST 40
2793 C1 05530 POPBC POP BC
2794 C8 05540 RET Z
2795 6F 05550 IOERR LD L,A ;Save error code
2796 2600 05560 LD H,0
2798 F6C0 05570 OR 0C0H ;Abbrev & return
279A 4F 05580 LD C,A
279B 05590 @@ERROR
279B 3E1A 00039 LD A,26
279D EF 00040 RST 40
279E 182C 05600 JR SAVESP
05610 ;
05620 ;      Routine to ck on pause or break
05630 ;
27A0 3E0D 05640 CKPAWS LD A,CR ;End line first
27A2 CD8227 05650 CALL BYTOUT
27A5 05660 CKPAWS0 @@FLAGS ;Get flag table pointer
27A5 3E65 00041 LD A,101
27A7 EF 00042 RST 40
27A8 FD7E0A 05670 LD A,(IY+'K'-'A') ;P/u KFLAG
27AB CB47 05680 BIT 0,A ;Check for break
27AD 2017 05690 JR NZ,BREAK ; if so exit
27AF CB4F 05700 BIT 1,A ;Check for pause
27B1 C8 05710 RET Z ;Ret if not
27B2 05720 CKPAW1 @@KEY ;Wait for key input
27B2 3E01 00043 LD A,1
27B4 EF 00044 RST 40

```

```

27B5 FE60      05730      CP      60H
27B7 28F9      05740      JR      Z,CKPAW1      ;Loop on pause
27B9 FE80      05750      CP      80H            ;Abort on BREAK
27BB 2809      05760      JR      Z,BREAK
27BD FD7E0A     05770 RESKFL LD      A,(IY+'K'-'A') ;Reset Pause & Enter bits
27C0 E6F9      05780      AND     0F9H          ;
27C2 FD770A     05790      LD      (IY+'K'-'A'),A
27C5 C9        05800      RET
                05810 ;
                05820 ;      BREAK handler routine
                05830 ;
27C6 CDBD27     05840 BREAK CALL    RESKFL
27C9 21FFFF     05850      LD      HL,-1
27CC 310000     05860 SAVESP LD      SP,$-$      ;Restore the stack
27CF           05870      @CKBRKC      ;Clear any <BREAK>
27CF 3E6A      00045      LD      A,106
27D1 EF        00046      RST      40
27D2 C9        05880      RET
                ; and RETURN
27D3 210000     05890 EXIT  LD      HL,0          ;Init to no error
27D6 18F4      05900      JR      SAVESP        ;P/u stack & return
                05910 ;
                05920 ;      String area
                05930 ;
27D8 4E        05940 NOPACK$ DB      'No Disk]',3
        6F 20 20 44 69 73 6B 5D
        20 03
27E3 49        05950 INACT$  DB      'Inactive'
        6E 61 63 74 69 76 65
27EB 0A        05960 DVCHDR$ DB      LF,'Options:',3
        4F 70 74 69 6F 6E 73 3A
        03
27F5 53        05970 DVCSS$  DB      'SpooLe','r'!80H,'Typ','e'!80H
        70 6F 6F 6C 65 F2 54 79
        70 E5
2800 56        05980          DB      'Verif','y'!80H,'Smoot','h'!80H
        65 72 69 66 F9 53 6D 6F
        6F 74 E8
280C 4D        05990          DB      'Memdis','k'!80H,'Form','s'!80H
        65 6D 64 69 73 EB 46 6F
        72 6D F3
2818 4B        06000          DB      'KS','M'!80H,'Graphi','c'!80H
        53 CD 47 72 61 70 68 69
        E3
2822 2C        06010 FAST$   DB      ', Fast',3
        20 46 61 73 74 03
2829 2C        06020 SLOW$   DB      ', Slow',3
        20 53 6C 6F 77 03
2830 53        06030 RES$    DB      'SYSRES',3
        59 53 52 45 53 03
2837 53        06040 SYSRES$ DB      'System modules resident:',3
        79 73 74 65 6D 20 6D 6F
        64 75 6C 65 73 20 72 65
        73 69 64 65 6E 74 3A 03
2850 20        06050 STPRAT$ DB      ' 6122030 3 61015'
        36 31 32 32 30 33 30 20
        33 20 36 31 30 31 35
2860 22        06060 FLOPY$  DB      '" Floppy #',3
        20 46 6C 6F 70 70 79 20
        23 03
286B 22        06070 RIGID$  DB      '" Rigid #',3
        20 52 69 67 69 64 20 20

```

```

23 03
2876 2C      06080 CYLS$   DB      ', Cyls='
      20 43 79 6C 73 3D
287D 20      06090 COMMA$  DB      '  , ',3
      20 20 2C 20 03
2883 52      06100 REMOV$   DB      'Removable',3
      65 6D 6F 76 61 62 6C 65
      03
288D 46      06110 FIXED$   DB      'Fixed',3
      69 78 65 64 03
2893 64      06120 DEN$     DB      'den, Sides=',3
      65 6E 2C 20 53 69 64 65
      73 3D 03
289F 2C      06130 STEP$    DB      ', Step=',3
      20 53 74 65 70 3D 03
28A7 6D      06140 MS$      DB      'ms',3
      73 03
28AA 2C      06150 DLY$     DB      ', Dly=',3
      20 44 6C 79 3D 03
28B1 4E      06160 NIL$     DB      'Nil'
      69 6C
28B4 20      06170 IO$      DB      ' <=> '
      3C 3D 3E 20
28B9      06180 PRMTBL$ EQU    $
0080      06190 VAL      EQU    80H
0040      06200 SW       EQU    40H
0020      06210 STR      EQU    20H
0010      06220 SGL      EQU    10H
28B9 80      06230      DB      80H
28BA 56      06240      DB      SW!SGL!6,'BYTEIO',0
      42 59 54 45 49 4F 00
28C2 8025    06250      DW      BPARAM+1
28C4 56      06260      DB      SW!SGL!6,'DRIVES',0
      44 52 49 56 45 53 00
28CC 1E24    06270      DW      DPARAM+1
28CE 55      06280      DB      SW!SGL!5,'PRINT',0
      50 52 49 4E 54 00
28D5 8A27    06290      DW      PPARAM+1
28D7 56      06300      DB      SW!SGL!6,'STATUS',0
      53 54 41 54 55 53 00
28DF C926    06310      DW      SPARM+1
28E1 56      06320      DB      SW!SGL!6,'OPTION',0
      4F 50 54 49 4F 4E 00
28E9 C926    06330      DW      SPARM+1
28EB 00      06340      NOP
      06350 ;
2900      06360      ORG      $<-8+1<8
0100      06370 BUFFER   DS      256
2A00      06380 STRBUF   EQU      $
      06390 ;
2400      06400      END      DEVICE

```


@@1	0000 @@2	0000 @@3	0000
@@4	0000 @MOD2	0000 @MOD4	FFFF
BEND	26C8 BPARM	257F BREAK	27C6
BUFFER	2900 BYTOUT	2782 CKPAW1	27B2
CKPAWS	27A0 CKPAWS0	27A5 COMAOK	2722
COMMA\$	287D CR	000D CVRTDEC	26A7
CYLS\$	2876 DCBD1	268C DCBD2	26A0
DCBDIR	2682 DEN\$	2893 DEND	257F
DEV1	2427 DEV2	245E DEV2A	2469
DEV2B	248A DEV3	24CF DEV4	24E2
DEVICE	2400 DEVICEA	2409 DLY\$	28AA
DOD1	26E6 DOD2	26F6 DOD3	2706
DORES	2725 DORES1	2741 DORES2	2757
DORES3	276B DPARM	241D DVCHDR\$	27EB
DVCS\$	27F5 DVRA0	25EE DVRA1	261B
DVRA2	261E DVRADDR	25BE DVRB0	2627
DVRB1	263D DVRB2	264A ENDLINE	2574
EXIT	27D3 FAST	2718 FAST\$	2822
FIXED\$	288D FLOPPY	2505 FLOPY\$	2860
INACT\$	27E3 IO\$	28B4 IOERR	2795
LF	000A LINOUT	2772 LOGDCB	2590
LOGDCB1	259E LOGRTE	25A6 MOVNAM	26B8
MOVNIL	2678 MS\$	28A7 NIL\$	28B1
NOPACK\$	27D8 NOTON	2726 POPBC	2793
POPDRV	2577 PPARM	2789 PRMTBL\$	28B9
REMOV\$	2883 RES\$	2830 RESKFL	27BD
RIGID\$	286B RTEF1	2670 RTEFCB	2654
SAVESP	27CC SGL	0010 SHOWFS	270D
SHOWIT	271B SLOW\$	2829 SPARM	26C8
STEP\$	289F STPRAT\$	2850 STR	0020
STRBUF	2A00 SW	0040 SYSRES\$	2837
TABLEN	264B VAL	0080 WPTST	247B
@@ABORT	A46E @@ADTSK	A501 @@BANK	AA19
@@BKSP	A6F9 @@BREAK	AA2F @@CHNIO	A459
@@CKBRKC	AA7D @@CKDRV	A555 @@CKEOF	A70E
@@CKTSK	A4EC @@CLOSE	A6E4 @@CLS	AA67
@@CMNDI	A498 @@CMNDR	A4AD @@CTL	A2BD
@@DATE	A42F @@DCSTAT	A594 @@DEBUG	A4D7
@@DECHEX	A999 @@DIRRD	A906 @@DIRWR	A91B
@@DIV16	A984 @@DIV8	A96F @@DODIR	A56A
@@DSP	A281 @@DSPLY	A321 @@ERROR	A4C2
@@EXIT	A483 @@FEXT	A873 @@FLAGS	AA03
@@FNAME	A888 @@FSPEC	A85E @@GATRD	A8F1
@@GATWR	A930 @@GET	A295 @@GTDCB	A8B2
@@GTDCT	A89D @@GTMOD	A8C7 @@HDFMT	A63C
@@HEX16	A9D8 @@HEX8	A9C3 @@HEXDEC	A9AE
@@HIGH\$	A9ED @@INIT	A6BA @@KBD	A2F9
@@KEY	A26D @@KEYIN	A30D @@KLTSK	A540
@@LOAD	A834 @@LOC	A723 @@LOF	A738
@@LOGGER	A358 @@LOGOT	A36D @@MSG	A3A4
@@MUL16	A95A @@MUL8	A945 @@OPEN	A6CF
@@PARAM	A41A @@PAUSE	A405 @@PEOF	A74D
@@POSN	A762 @@PRINT	A3B9 @@PRT	A2D1
@@PUT	A2A9 @@RAMDIR	A57F @@RDSEC	A612
@@RDSSC	A8DC @@READ	A777 @@REMOV	A6A5
@@RENAM	A690 @@REW	A78C @@RMTSK	A516
@@RPTSK	A52B @@RREAD	A7A1 @@RSLCT	A5FD
@@RSTOR	A5BE @@RUN	A849 @@RWRT	A7B6
@@SEEK	A5E8 @@SEEKSC	A7CB @@SKIP	A7E0
@@SLCT	A5A9 @@STEPI	A5D3 @@TIME	A444

@@VDCTL	A3F0 @@VER	A7F5 @@VRSEC	A627
@@WEOF	A80A @@WHERE	A2E5 @@WRITE	A81F
@@WRSEC	A651 @@WRSSC	A666 @@WRTRK	A67B

2400 is the transfer address
 00000 Total errors

NOTES:

Command: DIR, CAT

Library: SYS6/SYS

ISAM # : 21H, 20H

```

00100 ;LBDIR/ASM - DIR / CAT Command
0000 00110 TITLE <DIR - LS-DOS 6.2>
00120 ;
2400 00130 ORG 2400H
00140 ;
2400 C3D92D 00150 ENTRY JP DIR ;Go if DIR
00160 ;
2403 E5 00170 CATBGN PUSH HL ;Here if CAT
2404 210000 00180 LD HL,0 ;Set the DIR (A
2407 220628 00190 LD (APARM+1),HL ; parameter to OFF
240A E1 00200 POP HL ; and do a DIR
240B 18F3 00210 JR ENTRY ; command
00220 ;
4296 00230 BLKHASH EQU 4296H ;Hash code of blank password
00240 ;
240D 00250 *GET SVCMAC:3 ;Get SVC Macro equivalents and
00010 ;SVMAC/ASM - LS-DOS Version VI
00020 *LIST OFF
03900 *LIST ON
240D 00260 *GET VALUES:3 ; other misc. equates
03920 ;VALUES/ASM - Version 6
03930 *LIST OFF
04200 *LIST ON
00270 ;
240D 00280 *GET LBDIRA:3
04210 ;LBDIRA/ASM - DIR main processing loop
240D 04220 SUBTTL '<LBDIRA - Mainline Program>'

```

LBDIRA - Mainline Program

```

240D      04230      PAGE
          04240 ;
          04250 ;      error processing
          04260 ;
240D 217A2D 04270 NOMEM LD      HL,NOMEM$
2410 DD      04280      DB      0DDH
2411 219E2D 04290 BADFMT LD      HL,BADFMT$
2414      04300      @@LOGOT
          00001      IFEQ      00H,1
          00002      LD        HL,
          00003      ENDIF
2414 3E0C      00004      LD      A,12
2416 EF      00005      RST      40
          04310 ;
2417 21FFFF 04320 ABORT LD      HL,-1      ;Set HL = -1
241A 1810      04330      JR      SAVESP      ;Abort
          04340 ;
          04350 ;      I/O Error Routine
          04360 ;
241C 3E20      04370 ERR32 LD      A,32      ;"Illegal Drive Number"
241E 6F      04380 IOERR LD      L,A      ;Set HL = Error #
241F 2600      04390      LD      H,0
2421 F6C0      04400      OR      0C0H      ;Set short error
2423 4F      04410      LD      C,A      ;Stuff in C
2424      04420      @@ERROR      ;Display error
2424 3E1A      00006      LD      A,26
2426 EF      00007      RST      40
2427 1803      04430      JR      SAVESP      ;Abort
          04440 ;
          04450 ;      Clear stack & Exit
          04460 ;
2429 210000 04470 EXIT LD      HL,0      ;Good exit
242C 310000 04480 SAVESP LD      SP,$-$      ;P/u old SP address
242F      04490 ABORT3 @@CKBRKC      ;Clear break
242F 3E6A      00008      LD      A,106
2431 EF      00009      RST      40
2432 C9      04500      RET      ;Go home now
          04510 ;
          04520 ;      Init to 4 files/line & Drive # in string
          04530 ;
2433 C5      04540 DIR4 PUSH     BC      ;Save drive #
2434 3E04      04550      LD      A,4      ;4 filespecs/line
2436 320E28 04560      LD      (DONAM9+1),A      ;Save
2439 79      04570      LD      A,C      ;P/u drive #
243A C630      04580      ADD     A,'0'      ;Convert to ASCII
243C 320E2C 04590      LD      (DRIVE),A      ; & stuff in message
243F 32222D 04600      LD      (NDRIVE),A      ;Also stuff in No Disk
          04610 ;
          04620 ;      Is the starting Drive available ?
          04630 ;
2442      04640      @@GTDCT      ;IY => DCT+0
2442 3E51      00010      LD      A,81
2444 EF      00011      RST      40
2445      04650      @@CKDRV      ;Drive alive ?
2445 3E21      00012      LD      A,33
2447 EF      00013      RST      40
2448 F5      04660      PUSH     AF      ;Save RETURN condition
2449 CD932A 04670      CALL     CKPAWS      ;<BREAK> hit ?
244C F1      04680      POP      AF      ;NZ - couldn't log drive
244D 2826      04690      JR      Z,GDCKDRV      ;Z - Logged drive succ

```

LBDIRA - Mainline Program

```

04700 ;
04710 ;      Is this Drive enabled ?
04720 ;
244F FD7E00 04730 LD      A,(IY)      ;P/u Enable/Disable byte
2452 FEC3   04740 CP      0C3H      ;Enabled ?
2454 2809   04750 JR      Z,NO_DISK  ;Yes - display No Disk
04760 ;
04770 ;      If this is not global - Illegal Drive #
04780 ;
2456 3A3626 04790 LD      A,(SPECIF+1) ;Specific drive # ?
2459 B7     04800 OR      A      ;
245A CA1C24 04810 JP      Z,ERR32    ;Yes - illegal drive #
245D 1813   04820 JR      NEXTDRV   ;No - get next drive
04830 ;
04840 ;      Enabled Drive - Display "No Disk" string
04850 ;
245F 32882A 04860 NO_DISK LD      (NOTITLE+1),A ;Turn off title
2462 CD652A 04870 CALL     CKPAGE      ;Check for scroll
2465 211B2D 04880 LD      HL,NODISK   ;HL => "No Disk" string
2468 CDB929 04890 CALL     LINOUT      ;Display line
246B CD652A 04900 CALL     CKPAGE      ;Check for scroll
246E AF     04910 XOR      A      ;Turn on Title
246F 32882A 04920 LD      (NOTITLE+1),A ;
04930 ;
2472 C33426 04940 NEXTDRV JP      CKHIT4    ;Get next drive
04950 ;
04960 ;      Calculate quantity of Sectors/Gran
04970 ;
2475 C5     04980 GDCKDRV PUSH     BC      ;Save Drive #
2476 FD7E08 04990 LD      A,(IY+8)    ;P/u # Sectors/Gran
2479 E61F   05000 AND      1FH      ;Mask off junk
247B 3C     05010 INC      A      ;Bump for zero offset
247C 32ED29 05020 LD      (CALCK1+1),A ;Stuff it
05030 ;
05040 ;      P/u # Cylinders from DCT & stuff in string
05050 ;
247F FD6E06 05060 LD      L,(IY+6)    ;P/u cyl count
2482 2C     05070 INC      L      ;Offset from 0
2483 2600   05080 LD      H,0      ;Stuff in HL
2485 111B2C 05090 LD      DE,CYLCNT   ;DE => Destination
2488 3E20   05100 LD      A,' '      ;Leading space
248A CD382A 05110 CALL     CVD100      ;Stuff in string
05120 ;
05130 ;      Create "DDEN" String or "HARD" string
05140 ;
248D 11242C 05150 LD      DE,DENSITY  ;Destination
2490 21FF2B 05160 LD      HL,DEN
2493 3E44   05170 LD      A,'D'
2495 FDCB0376 05180 BIT      6,(IY+3)    ;Ck density
2499 2002   05190 JR      NZ,DUBDEN
249B 3E53   05200 LD      A,'S'
249D 77     05210 DUBDEN LD      (HL),A
249E 010400 05220 LD      BC,4      ;4 chars to Xfer
24A1 FDCB035E 05230 BIT      3,(IY+3)    ;Hard Drive ?
24A5 2803   05240 JR      Z,DOLDIR
24A7 21032C 05250 LD      HL,HARD    ;HL => "HARD"
24AA EDB0   05260 DOLDIR LDIR      ;Xfer string
05270 ;
05280 ;      Drive logged in - Read in GAT

```

LBDIRA - Mainline Program

```

05290 ;
24AC C1 05300 POP BC ;Recover Drive #
24AD 21002E 05310 LD HL,GAT ;HL => GAT buffer
24B0 FD5609 05320 LD D,(IY+9) ;D = Directory Cyl
24B3 1E00 05330 LD E,0 ;E = Gat Sector
24B5 05340 @RDSSC ;Read Sector
24B5 3E55 00014 LD A,85
24B7 EF 00015 RST 40
24B8 3E14 05350 LD A,20 ;Init "GAT Read Error"
24BA C21E24 05360 JP NZ,IOERR
24BD CD932A 05370 CALL CKPAWS ;<BREAK> hit ?
05380 ;
05390 ;
05400 ; Calculate the FREE space on the disk
05410 ;
05420 ;
24C0 110000 05430 LD DE,0 ;DE = Gran count
24C3 2ECC 05440 LD L,0CCH ;HL => GAT + X'CC'
24C5 7E 05450 LD A,(HL) ;P/u excess cyl byte
24C6 C623 05460 ADD A,35 ;Cyl excess of 35
24C8 47 05470 LD B,A ;Set loop counter
24C9 6A 05480 LD L,D ;HL => GAT + X'00'
24CA C5 05490 PUSH BC ;Save cyl count in B
05500 ;
05510 ; HL => GAT, B = # of cyls, DE = Gran count
05520 ;
24CB 7E 05530 FS1 LD A,(HL) ;P/u a GAT byte & set
24CC 37 05540 FS2 SCF ;Carry so bit 7 stays 1
05550 ;
05560 ; Is the granule in use ?
05570 ;
24CD 1F 05580 RRA ;Shift gran bit -> carry
24CE 3801 05590 JR C,FS3 ;Don't inc if in use
05600 ;
05610 ; Free Granule - bump Free Granule Count
05620 ;
24D0 13 05630 INC DE ;Another spare gran
24D1 FEFF 05640 FS3 CP 0FFH ;Fin with this GAT byte?
24D3 20F7 05650 JR NZ,FS2 ;Loop if not
05660 ;
05670 ; Finished with GAT byte, advance to next
05680 ;
24D5 2C 05690 INC L ;Advance to next byte
24D6 10F3 05700 DJNZ FS1 ;B cylinders to check
05710 ;
05720 ; DE = Free Grans, Calculate # Grans/cyl
05730 ;
24D8 C1 05740 POP BC ;B = # of cylinders
24D9 FD7E08 05750 LD A,(IY+8) ;P/u DCT+8
24DC 07 05760 RLCA ;Move Grans/Cyl into
24DD 07 05770 RLCA ;Bits 0-2
24DE 07 05780 RLCA
24DF E607 05790 AND 7
24E1 3C 05800 INC A ;A = Grans/Cylinder
24E2 FDCB046E 05810 BIT 5,(IY+4) ;Double-bit set ?
24E6 2801 05820 JR Z,NOTDUB ;No - don't double
24E8 87 05830 ADD A,A ;Double grans/cylinder
05840 ;
05850 ; A = # Grans/Cyl, Calculate Total # of Grans

```


LBDIRA - Mainline Program

```

05860 ;
24E9 210000 05870 NOTDUB LD HL,0 ;Init HL = 0
24EC D5 05880 PUSH DE ;Save Free Grans
24ED 54 05890 LD D,H ;Set DE = # cyls
24EE 58 05900 LD E,B
24EF 47 05910 LD B,A ;B = Grans/Cyl
05920 ;
05930 ; Multiply Grans/Cyl (B) x # Cyls (DE)
05940 ;
24F0 19 05950 GPCLOOP ADD HL,DE ;Add cylinder count
24F1 10FD 05960 DJNZ GPCLOOP ;Grans/cyl times
05970 ;
05980 ; HL = # of grans/disk, Is this a hard drive ?
05990 ;
24F3 FDCB035E 06000 BIT 3,(IY+3) ;Hard Drive ?
24F7 201E 06010 JR NZ,SKIPLOC ;Yes-don't check lockout
06020 ;
06030 ; Floppy disk - check for locked out cylinders
06040 ;
24F9 43 06050 LD B,E ;B = cylinder count
24FA EB 06060 EX DE,HL ;Save total cnt in DE
24FB 21602E 06070 LD HL,GAT+60H ;HL => Lockout table
24FE 0E00 06080 LD C,0 ;C = Locked out cyl count
2500 F5 06090 PUSH AF ;Save Grans/Cyl in A
06100 ;
06110 ; Loop to count up Locked out cylinders in C
06120 ;
2501 3E01 06130 LKLOOP LD A,1 ;Init cyl checker
2503 A6 06140 AND (HL) ;Locked out ?
2504 2801 06150 JR Z,GOODCYL ;No - good cylinder
2506 0C 06160 INC C ;Bump locked out count
2507 2C 06170 GOODCYL INC L ;Bump ptr
2508 10F7 06180 DJNZ LKLOOP ;B cylinders
06190 ;
06200 ; Multiply Cylinders (BC) x Grans/Cyl
06210 ;
250A F1 06220 POP AF ;A = Grans/Cyl
250B F5 06230 PUSH AF ;Save it
250C 60 06240 LD H,B ;Init HL = 0
250D 68 06250 LD L,B
06260 ;
250E 09 06270 GTUSED ADD HL,BC ;Add cylinder count
250F 3D 06280 DEC A ;Grans/cyl times
2510 20FC 06290 JR NZ,GTUSED
2512 F1 06300 POP AF ;A = Grans/Cyl
06310 ;
06320 ; Subtract # of Grans locked out from total
06330 ;
2513 B7 06340 OR A ;Clear carry
2514 EB 06350 EX DE,HL
2515 ED52 06360 SBC HL,DE ;HL = Grans possible
2517 D1 06370 SKIPLOC POP DE ;Rcvr # of Free Grans
06380 ;
06390 ; HL = # Grans possible, DE = # Grans Free
06400 ;
2518 E5 06410 PUSH HL ;Save Grans used
2519 21302C 06420 LD HL,KFREE ;Convert Grans Free
251C CDE829 06430 CALL CALCK ; to ASCII K & stuff
251F D1 06440 POP DE ; into string.

```

LBDIRA - Mainline Program

```

06450 ;
06460 ; Calculate # of K used & stuff into header
06470 ;
2520 213C2C 06480 LD HL,KPOSS ;Pt to where to stuff
2523 CDE829 06490 CALL CALCK ;Calculate K & stuff
06500 ;
06510 ; Transfer Diskette Name into string buffer
06520 ;
2526 21D02E 06530 LD HL,GAT+0D0H ;HL => Diskette Name
2529 11112C 06540 LD DE,NAME ;Move pack name -> header
252C 0E08 06550 LD C,8 ;BC = 8 chars to xfer
252E EDB0 06560 LDIR ;Xfer into buff
06570 ;
06580 ; Clear out Date buffer
06590 ;
2530 114D2C 06600 LD DE,DATBUF ;DE => Start of buffer
2533 3E20 06610 LD A,' ' ;Space
2535 0609 06620 LD B,9 ;9 chars to clear
2537 12 06630 CLRLP LD (DE),A ;Stuff in space
2538 13 06640 INC DE ;Bump
2539 10FC 06650 DJNZ CLRLP
06660 ;
06670 ; HL => Date in mm/dd/yy format - p/u month
06680 ;
253B 7E 06690 LD A,(HL) ;P/u month
253C D630 06700 SUB '0' ;Convert tens to binary
253E 4F 06710 LD C,A ;Save in C
06720 ;
06730 ; Multiply first digit of month x 10
06740 ;
253F 87 06750 ADD A,A ;X 2
2540 87 06760 ADD A,A ;X 4
2541 81 06770 ADD A,C ;X 5
2542 87 06780 ADD A,A ;X 10
2543 4F 06790 LD C,A ;Stuff in C
06800 ;
06810 ; Pick up second digit of month & add to 10's
06820 ;
2544 23 06830 INC HL ;Bump to ones
2545 7E 06840 LD A,(HL) ;P/u ones of month
2546 D630 06850 SUB '0' ;Convert to binary
2548 81 06860 ADD A,C ;A = Month (1-12)
2549 282E 06870 JR Z,ILLDATE ;Abort if NO DATE
254B FE0D 06880 CP 13 ;Legal Month ?
254D 302A 06890 JR NC,ILLDATE ;No - illegal date
06900 ;
06910 ; Legal Month - Mult x 3 & pt to month string
06920 ;
254F 4F 06930 LD C,A ;Xfer month to C
2550 87 06940 ADD A,A ;X 2
2551 81 06950 ADD A,C ;X 3
2552 4F 06960 LD C,A ;BC = offset
2553 E5 06970 PUSH HL ;Save date pointer
2554 21AB2D 06980 LD HL,MONTBL-3 ;HL => Month String table
2557 09 06990 ADD HL,BC ;HL => Month String
07000 ;
07010 ; HL => Month String, Stuff into Buffer
07020 ;
2558 11502C 07030 LD DE,DATBUF+3 ;DE => Destination

```

LBDIRA - Mainline Program

```

255B 0E03      07040      LD      C,3          ;BC = 3 chars to xfer
255D EDB0      07050      LDIR          ;Xfer date to buffer
                07060 ;
                07070 ;      Transfer Day (00-31) into date buffer
                07080 ;

255F E1        07090      POP      HL          ;Recover ptr
2560 23        07100      INC      HL          ;Bump
2561 23        07110      INC      HL          ;HL => Day of month
2562 114D2C    07120      LD      DE,DATBUF    ;DE => date buffer
2565 0E02      07130      LD      C,2          ;Xfer into buffer
2567 EDB0      07140      LDIR
                07150 ;
                07160 ;      Transfer Year into buffer
                07170 ;

2569 23        07180      INC      HL          ;HL => Year (80-87)
256A 0E02      07190      LD      C,2          ;2 chars to xfer
256C 11542C    07200      LD      DE,DATBUF+7    ;DE => Destination
256F EDB0      07210      LDIR          ;Xfer into buffer
                07220 ;
                07230 ;      Stuff "-" after day & month
                07240 ;

2571 3E2D      07250      LD      A,'-'        ;Separator
2573 324F2C    07260      LD      (DATBUF+2),A
2576 32532C    07270      LD      (DATBUF+6),A
                07280 ;
                07290 ;      Display the files in the directory
                07300 ;      Init DIR rec ptr = mem start, count = 0
                07310 ;

2579 3C        07320      ILLDATE INC      A          ;Set flag
257A 32D82D    07330      LD      (FILFLAG),A    ;Set file alr disp flag
257D 210030    07340      LD      HL,MEMORY      ;Init DIRPTR to start
2580 22D92D    07350      LD      (DIRPTR),HL    ; of available memory
2583 AF        07360      XOR      A          ;Set File display
2584 ED62      07370      SBC      HL,HL        ;Set HL = 0
2586 22F725    07380      LD      (TFILES+1),HL ;Total Files = 0
2589 22D625    07390      LD      (COUNT+1),HL ;Count = 0
258C 220226    07400      LD      (TOTGRNS+1),HL ;Total Grans = 0
                07410 ;
                07420 ;      Read in the HIT of the disk
                07430 ;

258F C1        07440      POP      BC          ;Recover Drive # in C
2590 FD5609    07450      LD      D,(IY+9)      ;P/u directory cylinder
2593 1E01      07460      LD      E,1          ;Pt to HIT sector
2595 21002E    07470      LD      HL,HIT        ;HL => I/O buffer
2598          07480      @@RDSSC          ;Read System Sector
2598 3E55      00016      LD      A,85
259A EF        00017      RST      40
259B 3E16      07490      LD      A,16H        ;"HIT read error"?
259D C21E24    07500      JP      NZ,IOERR      ;Jump if read error
25A0 CD932A    07510      CALL   CKPAWS        ;<BREAK> hit ?
25A3 C34526    07520      JP      CKHIT5        ;Jump into middle of loop
                07530 ;
                07540 ;      Loop to Process HIT entries
                07550 ;

25A6 E1        07560      CKHIT POP      HL
25A7 C1        07570      CKHIT1 POP      BC          ;Recover HIT pointer lo
                07580 ;
                07590 ;      Point HL => Last HIT entry
                07600 ;

```

LBDIRA - Mainline Program

```

25A8 262E    07610      LD      H,HIT<-8      ;Set H = hi byte of HIT
25AA 68      07620      LD      L,B           ;HL => Last HIT entry
                07630 ;
                07640 ;      Position to next entry of the Record
                07650 ;
25AB 7D      07660 CKHIT2 LD      A,L           ;P/u current entry
25AC C620    07670      ADD     A,32          ;Add 32 (bytes/entry)
25AE 6F      07680      LD      L,A           ;HL => Next entry
25AF 30F2    07690      JR      NC,$JP0       ;Go to next record ?
                07700 ;
                07710 ;      Position to entry zero of next record
                07720 ;
25B1 2C      07730      INC     L             ;Posn to next record
25B2 CB6D    07740      BIT     5,L           ;Done with drive ?
25B4 28ED    07750      JR      Z,$JP0       ;No - process entry
                07760 ;
                07770 ;      Finished with drive - Sort data unless (O=N)
                07780 ;
25B6 3AF 926 07790      LD      A,(SORTPRM+1) ;If sort requested,
25B9 B7      07800      OR      A             ; then need to output
25BA C4DF 2A 07810      CALL    NZ,SORTIT      ; the sorted data
                07820 ;
                07830 ;      Were there any files displayed ?
                07840 ;
25BD 2AD625 07850      LD      HL,(COUNT+1) ;P/u displayed file count
25C0 7C      07860      LD      A,H           ;Any entered ?
25C1 B5      07870      OR      L             ;
25C2 200B    07880      JR      NZ,FILES      ;Yes - dsp under if (A)
                07890 ;
                07900 ;      Display Title & line feed
                07910 ;
25C4 21072C 07920      LD      HL,DSTRING    ;HL => Title
25C7 CDB929 07930      CALL    LINOUT        ;Display title
25CA CD652A 07940      CALL    CKPAGE        ;Check for scroll
25CD 184A    07950      JR      NOTAP        ;Get next drive
                07960 ;
                07970 ;      Get next drive # if the A parm was specified
                07980 ;
25CF 3A0628 07990 FILES LD      A,(APARM+1)    ;Don't display if A
25D2 B7      08000      OR      A             ;
25D3 2844    08010      JR      Z,NOTAP        ;Not A - Output C/R
                08020 ;
                08030 ;      Were there any files shown in directory ?
                08040 ;
25D5 210000 08050 COUNT LD      HL,$-$      ;P/u count
25D8 7C      08060      LD      A,H           ;Any files shown ?
25D9 B5      08070      OR      L             ;
25DA 284E    08080      JR      Z,TERMDRV     ;No - get next drive
                08090 ;
                08100 ;      Display Line of equal signs "="
                08110 ;
25DC 064F    08120      LD      B,79          ;Output 79 "="
25DE 3E3D    08130 D79EQ LD      A,'='      ;
25E0 CDCB29 08140      CALL    BYTOUT        ;Output "="
25E3 10F9    08150      DJNZ    D79EQ        ;
                08160 ;
                08170 ;      End line & check for scroll
                08180 ;
25E5 3E0D    08190      LD      A,CR          ;End line with C/R

```

LBDIRA - Mainline Program

```

25E7 CDCB29 08200 CALL BYTOUT
25EA CD652A 08210 CALL CKPAGE
                08220 ;
                08230 ; Stuff # of files used into footer string
                08240 ;
25ED C5 08250 PUSH BC ;Save Drive #
25EE 11E92C 08260 LD DE,FDISP ;DE => Destination
25F1 3E20 08270 LD A,' ' ;Stuff # of files
25F3 CD382A 08280 CALL CVD100 ; used in string.
                08290 ;
                08300 ; Pick up # of used files & stuff in string
                08310 ;
25F6 210000 08320 TFILES LD HL,$-$ ;P/u total files used
25F9 11FA2C 08330 LD DE,FUSED ;DE => Destination
25FC 3E20 08340 LD A,' '
25FE CD382A 08350 CALL CVD100 ;Stuff in string
                08360 ;
                08370 ; P/u Total # of Grans & stuff into string
                08380 ;
2601 110000 08390 TOTGRNS LD DE,$-$ ;P/u total # of Grans
2604 21102D 08400 LD HL,SPUSED ;HL => Destination
2607 CDE829 08410 CALL CALCK ;Stuff into string
260A C1 08420 POP BC ;C = drive #
                08430 ;
                08440 ; Display Footer String
                08450 ;
260B 21DE2C 08460 LD HL,FSTRING ;HL => Files disp string
260E CDB929 08470 CALL LINOUT ;Display line
2611 CD652A 08480 CALL CKPAGE ;Check for title
2614 CD652A 08490 CALL CKPAGE
2617 1811 08500 JR TERMDRV ;Get next drive
                08510 ;
                08520 ; A parm not spec'd, was a header displayed ?
                08530 ;
2619 3AD82D 08540 NOTAP LD A,(FILFLAG) ;Was a header displayed ?
261C B7 08550 OR A
261D 200B 08560 JR NZ,TERMDRV ;No - get next drive
                08570 ;
                08580 ; Output a C/R if a full line wasn't displayed
                08590 ;
261F 3A0E28 08600 LD A,(DONAM9+1) ;Full line ?
2622 FE04 08610 CP 4
2624 C41928 08620 CALL NZ,ENDLINE ;End line
2627 CD1928 08630 CALL ENDLINE ;Do a blank line
                08640 ;
                08650 ; Position to next drive - or exit if finished
                08660 ;
262A 3E00 08670 TERMDRV LD A,$-$ ;P/u term drive
262C 0C 08680 INC C ;Bump current drive #
262D B9 08690 CP C ;Done ?
262E D23324 08700 JP NC,DIR4 ;Loop if in range
2631 C32924 08710 JP EXIT ;Exit if NZ
                08720 ;
                08730 ; Get next drive unless drivespec specified
                08740 ;
2634 C1 08750 CKHIT4 POP BC ;Get drive # in C
2635 3E00 08760 SPECIF LD A,$-$ ;P/u specific flag
2637 B7 08770 OR A
2638 210000 08780 LD HL,0 ;Init in case exit

```

LBDIRA - Mainline Program

```

263B C8      08790      RET      Z      ;Not global
              08800 ;
              08810 ;      Bump Drive number
              08820 ;
263C 3A2B26  08830      LD      A,(TERMDRV+1) ;P/u term drive #
263F 0C      08840      INC      C      ;Bump
2640 B9      08850      CP      C      ;Finished ?
2641 D23324  08860      JP      NC,DIR4      ;Loop if more
2644 C9      08870      RET      ; else return
              08880 ;
              08890 ;      Is the HIT entry in use ?
              08900 ;
2645 7E      08910 CKHIT5 LD      A,(HL)      ;P/u HIT entry
2646 B7      08920      OR      A      ;In use ?
2647 CAAB25  08930      JP      Z,CKHIT2      ;No - get next entry
              08940 ;
              08950 ;      HIT entry in use - Point HL to that entry
              08960 ;
264A 45      08970      LD      B,L      ;Save DEC in B
264B C5      08980      PUSH     BC      ; & to stack
264C 7D      08990      LD      A,L      ;Point L to Entry posn
264D E6E0    09000      AND      0E0H
264F 6F      09010      LD      L,A
              09020 ;
              09030 ;      Do we need to Read in another sector ?
              09040 ;
2650 A8      09050      XOR      B      ;Done with 8 entries ?
2651 FEFF    09060 CKHIT6 CP      0FFH
2653 2810    09070      JR      Z,CKDIR1      ;No - check out entry
              09080 ;
              09090 ;      Read in the next directory sector
              09100 ;
2655 325226  09110      LD      (CKHIT6+1),A ;Stuff in last entry posn
2658         09120      @DIRRD ; & read it into buffer
2658 3E57    09018      LD      A,87
265A EF      09019      RST      40
265B C21E24  09130      JP      NZ,IOERR      ;Jump on read error
265E 7C      09140      LD      A,H      ;P/u high byte
265F 326626  09150      LD      (CKDIR1+1),A ; and save
2662 328629  09160      LD      (SBUFFER+1),A ; for later
              09170 ;
              09180 ;      Valid File (Alive & FPDE) ?
              09190 ;
2665 2600    09200 CKDIR1 LD      H,$-$      ;P/u high byte
2667 CB66    09210      BIT      4,(HL)      ;Alive ?
2669 CAA725  09220      JP      Z,CKHIT1      ;No - get next entry
266C CB7E    09230      BIT      7,(HL)      ;FPDE ?
266E C2A725  09240      JP      NZ,CKHIT1      ;No - get next entry
              09250 ;
              09260 ;      Alive FPDE - Bump Total File counter
              09270 ;
2671 E5      09280      PUSH     HL      ;Save ptr
2672 2AF725  09290      LD      HL,(TFILES+1) ;HL => Total Files
2675 23      09300      INC      HL      ;Bump total files
2676 22F725  09310      LD      (TFILES+1),HL
2679 E1      09320      POP      HL
              09330 ;
              09340 ;      Is this a SYStem File ?
              09350 ;

```

LBDIRA - Mainline Program

```

267A CB76      09360      BIT      6,(HL)      ;SYS file ?
267C 280A      09370      JR        Z,CKDIR3    ;No - continue
                09380      ;
                09390      ;      SYS file - don't check unless S parm entered
                09400      ;
267E 110000    09410      SPARM    LD        DE,$-$      ;P/u S-parm
2681 7A        09420      LD        A,D          ;Specified ?
2682 B3        09430      OR        E
2683 CAA725    09440      JP        Z,CKHIT1      ;No - don't check it
2686 180C      09450      JR        CKMOD        ;Skip INV check
                09460      ;
                09470      ;      Non-SYS file - Is the file Visible ?
                09480      ;
2688 CB5E      09490      CKDIR3  BIT      3,(HL)      ;Visible ?
268A 2808      09500      JR        Z,CKMOD        ;Yes - skip I check
                09510      ;
                09520      ;      File is invisible - was INV (I) specified ?
                09530      ;
268C 110000    09540      IPARM    LD        DE,$-$      ;I-parm
268F 7A        09550      LD        A,D          ;Ignore if I-parm not
2690 B3        09560      OR        E            ; entered as this file
2691 CAA725    09570      JP        Z,CKHIT1      ; is invisible
                09580      ;
                09590      ;      Was the MOD parm entered ?
                09600      ;
2694 110000    09610      CKMOD    LD        DE,$-$      ;P/u mod parm
2697 7A        09620      LD        A,D          ;Was it entered ?
2698 B3        09630      OR        E
2699 2807      09640      JR        Z,CKNAM        ;Go if MOD not entered
                09650      ;
                09660      ;      MOD parm entered - was this file modified ?
                09670      ;
269B 2C        09680      INC      L            ;HL => DIR + 1
269C CB76      09690      BIT      6,(HL)      ;Was the file modified ?
269E CAA725    09700      JP        Z,CKHIT1      ;No - get next entry
26A1 2D        09710      DEC      L            ;Adjust back to start
                09720      ;
                09730      ;      Attributes match - check if filespec matches
                09740      ;
26A2 E5        09750      CKNAM    PUSH     HL          ;Save ptr to record
26A3 7D        09760      LD        A,L          ;Pt to filename in dir
26A4 C605      09770      ADD      A,5
26A6 6F        09780      LD        L,A          ;HL => DIR filename
26A7 113A2D    09790      LD        DE,BLANKS      ;DE => Partspec input
26AA 060B      09800      LD        B,11          ;Ck name/ext (11-chars)
                09810      ;
                09820      ;      Loop to check if partspec matches dir name
                09830      ;
26AC 1A        09840      CKNAM1  LD        A,(DE)      ;P/u partspec
26AD FE24      09850      CP        '$'          ;Wild char?
26AF 2807      09860      JR        Z,CKNAM2      ;Yes - match
                09870      ;
                09880      ;      Does Directory char match partspec char ?
                09890      ;
26B1 BE        09900      CP        (HL)          ;Not global, char match?
26B2 2804      09910      JR        Z,CKNAM2      ;Ck more if match
                09920      ;
                09930      ;      Chars don't match - Dir char a space ?
                09940      ;

```

LBDIRA - Mainline Program

```

26B4 FE20      00950      CP      ' '      ;Blank = end of ck
26B6 200B      00960      JR      NZ,MFLG      ;If not blank, no match
                00970 ;
                00980 ;      Bump Dir ptr & Partspec ptr & continue loop
                00990 ;
26B8 23        10000 CKNAM2 INC      HL      ;Bump pointers
26B9 13        10010      INC      DE
26BA 10F0      10020      DJNZ     CKNAM1      ;Loop for 11 chars
                10030 ;
                10040 ;      Entries Match - Was the "-" Exclude given ?
                10050 ;
26BC 3AC426    10060      LD      A,(MFLG+1)      ;P/u flag
26BF FE2D      10070      CP      '-'      ; - exclude given ?
26C1 1803      10080      JR      CK2HIT      ;Yes - get next entry
                10090 ;
                10100 ;      Entries Don't match - Was exclude given ?
                10110 ;
26C3 3E00      10120 MFLG LD      A,$-$      ;P/u Exclude flag
26C5 B7        10130      OR      A      ;If no exclude given
26C6 CAA625    10140 CK2HIT JP      Z,CKHIT      ; get next entry
                10150 ;
                10160 ;      Recover DIR+0 pointer
                10170 ;
26C9 E1        10180 CKNAM2A POP     HL      ;Rcvr ptr to DIR+0
26CA E5        10190      PUSH    HL      ;Save
                10200 ;
                10210 ;      Unpack Date of Directory entry
                10220 ;
26CB 23        10230      INC      HL      ;HL => DIR+1
26CC CD192A    10240      CALL    UNPACK      ;Unpack date
                10250 ;
                10260 ;      Use Dates before user-specified date ?
                10270 ;
26CF 3AD72D    10280      LD      A,(FTFLG)      ;P/u From/To flag
26D2 07        10290      RLCA      ;Tst fm bit
26D3 300F      10300      JR      NC,CKNAM2B      ;No - check to
                10310 ;
                10320 ;      "FROM" flag set - does file have a date ?
                10330 ;
26D5 7A        10340      LD      A,D      ;Ignore if no date
26D6 B3        10350      OR      E      ; in DIR for file
26D7 CAA625    10360      JP      Z,CKHIT      ;No date - get next entry
                10370 ;
                10380 ;      Is the Specified date >= the file's date ?
                10390 ;
26DA 2ADD2D    10400      LD      HL,(FMPAKD)      ;P/u user date entry
26DD EB        10410      EX      DE,HL
26DE CD132A    10420      CALL    CPHLDE      ;Compare HL to DE
26E1 EB        10430      EX      DE,HL      ;File date < User date ?
26E2 3811      10440      JR      C,$JP1      ;Yes - get next entry
                10450 ;
                10460 ;      Use Dates after user-specified Date ?
                10470 ;
26E4 3AD72D    10480 CKNAM2B LD      A,(FTFLG)      ;P/u FROM/TO flag
26E7 0F        10490      RRCA      ;Test TO bit
26E8 300E      10500      JR      NC,SortPRM      ;Go if no TOPARM
                10510 ;
                10520 ;      "TO" Flag set - Does file have a date ?
                10530 ;

```


LBDIRA - Mainline Program

```

26EA 7A      10540      LD      A,D          ;File have a valid date ?
26EB B3      10550      OR       E
26EC CAA625   10560      JP       Z,CKHIT      ;No - get next entry
                10570 ;
                10580 ;      File has a date - Is spec'd date less ?
                10590 ;
26EF 2ADF2D   10600      LD      HL,(TOPAKD)    ;P/u user's packed date
26F2 CD132A   10610      CALL    CPHLDE      ;User date < File date ?
26F5 DAA625   10620 $JP1  JP       C,CKHIT      ;Yes - get next entry
                10630 ;
                10640 ;      Was the Sort Parameter turned off ?
                10650 ;
26F8 11FFFF   10660 SORTPRM LD      DE,-1      ;P/u default parm
26FB E1       10670      POP     HL          ;HL => DIR+0
26FC 7A       10680      LD      A,D          ;Default to SORT=ON
26FD B3       10690      OR       E
26FE 282D     10700      JR       Z,DODSP      ;Go display if no sort
                10710 ;
                10720 ;      SORT = ON --- Calculate allocation & extents
                10730 ;
2700 E5       10740      PUSH    HL          ;Save DIR + 0 ptr
2701 CD5B29   10750      CALL    ALL09A      ;Calc alloc & extents
2704 E1       10760      POP     HL          ;Recover DIR+0 ptr
                10770 ;
                10780 ;      Overwrite FPDE's 22-25 with # Grans & # exts
                10790 ;
2705 E5       10800      PUSH    HL          ;Point IX = DIR+22
2706 DDE1     10810      POP     IX
2708 DD7316   10820      LD      (IX+22),E      ;Stuff in # Grans
270B DD7217   10830      LD      (IX+23),D
270E DD7118   10840      LD      (IX+24),C      ;Stuff in # Extents
2711 DD7019   10850      LD      (IX+25),B
                10860 ;
                10870 ;      Transfer Record into Memory For Sort
                10880 ;
2714 ED5BD92D 10890      LD      DE,(DIRPTR)    ;P/u last used mem addr
2718 E5       10900      PUSH    HL          ;Save current DIR ptr
2719 012000    10910      LD      BC,32      ;Move record to buffer
271C ED80     10920      LDIR                     ;Xfer
271E ED53D92D 10930      LD      (DIRPTR),DE      ;Update the pointer
                10940 ;
                10950 ;      Is there an overflow of available memory ?
                10960 ;
2722 2ADB2D   10970      LD      HL,(MAXMEM)    ;P/u approximate hi-mem
2725 ED52     10980      SBC     HL,DE      ;Did it overflow ?
2727 D2A625   10990      JP       NC,CKHIT      ;No - get next entry
272A C30D24   11000      JP       NOMEM      ;Insuf mem for sort buff
                11010 ;
                11020 ;      Display A Filename
                11030 ;
272D CD3327   11040 DODSP  CALL    MATCH      ;Display entry
2730 C3A725   11050      JP       CKHIT1      ;Loop to next DIR entry
2733          00290 *GET LBDIRB:3
                11060 ;LBDIRB/ASM - Display Filespec & attributes
2733          11070      SUBTTL '<LBDIRC - File Attribute Output>'

```

LBDIRC - File Attribute Output

```

2733          11080 PAGE
                11090 ;
                11100 ; MATCH - Display a File's Name and Extension
                11110 ;
2733 E5        11120 MATCH PUSH HL ;Save HIT posn
2734 21D625    11130 LD HL,COUNT+1 ;Bump file count
2737 34        11140 INC (HL)
                11150 ;
                11160 ; Was the Drive Header Displayed ?
                11170 ;
2738 21D82D    11180 LD HL,FILFLAG ;HL => File Header flag
273B AF        11190 XOR A ;If (HL) is Non-Zero
273C BE        11200 CP (HL) ; then the header has not
273D 77        11210 LD (HL),A ; printed.
273E C4B12A    11220 CALL NZ,CKTITL ;Display title if NZ
                11230 ;
                11240 ; Position HL to Directory Entry Filename
                11250 ;
2741 E1        11260 ALRPRT POP HL ;Recover DEC
2742 7D        11270 LD A,L ;P/u DEC
2743 E6E0      11280 AND 0E0H ;Posn to entry
2745 C605      11290 ADD A,5 ;Pt to start of filename
2747 6F        11300 LD L,A ;HL => Filename field
                11310 ;
                11320 ; Init B=8 chars for filename, C=19 to col
                11330 ;
2748 0E13      11340 LD C,19 ;Chars to next column
274A 0608      11350 LD B,8 ;Filename
                11360 ;
                11370 ; Loop to Output the Filename
                11380 ;
274C 7E        11390 DONAM1 LD A,(HL) ;P/u character
274D 23        11400 INC HL ;Bump DIR ptr
274E FE20      11410 CP ' ' ;Space ?
2750 2807      11420 JR Z,DONAM2 ;Yes - done with filename
2752 CDCA29    11430 CALL BYTOUT2 ;No - output char
2755 10F5      11440 DJNZ DONAM1 ;Field loop
2757 1804      11450 JR DONAM3 ;Bypass ext calculation
                11460 ;
                11470 ; Filename has < 8 chars, Pt to extension
                11480 ;
2759 7D        11490 DONAM2 LD A,L ;P/u low byte
275A 80        11500 ADD A,B ;Add # of chars left
275B 3D        11510 DEC A ;Back one
275C 6F        11520 LD L,A ;HL => Extension
                11530 ;
                11540 ; Does this file have an extension ?
                11550 ;
275D 7E        11560 DONAM3 LD A,(HL) ;P/u first char
275E FE20      11570 CP ' ' ;Blank
2760 2812      11580 JR Z,DONAM5 ;Yes - no extension
                11590 ;
                11600 ; Output a "/" & Set up for Extension loop
                11610 ;
2762 3E2F      11620 LD A,'/' ;Display slash
2764 CDCA29    11630 CALL BYTOUT2
2767 0603      11640 LD B,3 ;3 chars max for EXT
                11650 ;
                11660 ; Loop to output the extension
                11670 ;

```

LBDIRC - File Attribute Output

```

2769 7E      11680 DONAM4 LD      A,(HL)      ;P/u char
276A 23      11690      INC      HL           ;Bump ptr
276B FE20    11700      CP        ' '         ;Space ?
276D 2805    11710      JR        Z,DONAM5     ;Exit on 1st blank
276F CDCA29  11720      CALL     BYTOUT2       ;Else display the char
2772 10F5    11730      DJNZ     DONAM4        ;Loop 3 chars
          11740 ;
          11750 ;      Was the (A) parameter specified ?
          11760 ;
2774 3A0628  11770 DONAM5 LD      A,(APARM+1)   ;A parm specified ?
2777 B7      11780      OR        A
2778 2813    11790      JR        Z,DONAM5A     ;No - continue
          11800 ;
          11810 ;      (A) parameter specified - Tab to column 14
          11820 ;
277A 79      11830      LD        A,C          ;P/u chars left to col 20
277B D606    11840      SUB        6           ;Adjust to column 14
277D 47      11850      LD        B,A          ;Stuff into B for DJNZ
277E CDE029  11860      CALL     OUTSPC        ;Output B spaces
          11870 ;
          11880 ;      Output mod flag (if modified) & tab to 19
          11890 ;
2781 7D      11900      LD        A,L          ;Pt HL => DIR+0
2782 E6E0    11910      AND        0E0H
2784 6F      11920      LD        L,A
2785 CD3729  11930      CALL     OUTMOD        ;Output "+" if mod
2788 0603    11940      LD        B,3          ;Output 3 spaces
278A CDE029  11950      CALL     OUTSPC        ;Output B spaces
          11960 ;
          11970 ;      Display the File's Attributes
          11980 ;
278D 0601    11990 DONAM5A LD      B,1          ;Set B=1 space
278F CDE029  12000      CALL     OUTSPC        ;After filespec.
          12010 ;
          12020 ;      Point HL => DIR+0 (Attributes)
          12030 ;
2792 7D      12040      LD        A,L          ;Pt to 1st byte of
2793 E6E0    12050      AND        0E0H        ;Directory record
2795 6F      12060      LD        L,A
          12070 ;
          12080 ;      Display "?" if File OPEN bit set
          12090 ;
2796 3E3F    12100      LD        A,'?'        ;"?" character
2798 23      12110      INC        HL          ;HL => DIR + 1
2799 CB6E    12120      BIT        5,(HL)      ;File Open ?
279B 2B      12130      DEC        HL          ;HL => DIR + 0
279C C4CA29  12140      CALL     NZ,BYTOUT2     ;Yes - output byte
          12150 ;
          12160 ;      Display an "*" if this is a PDS file
          12170 ;
279F 46      12180      LD        B,(HL)       ;P/u attributes byte
27A0 3E2A    12190      LD        A,'*'        ;Init for PDS display
27A2 CB68    12200      BIT        5,B
27A4 C4CA29  12210      CALL     NZ,BYTOUT2     ;Display if PDS
          12220 ;
          12230 ;      Display an "S" if file is a SYS file
          12240 ;
27A7 CB70    12250      BIT        6,B          ;Is it a SYS file?
27A9 3E53    12260      LD        A,'S'

```

LBDIRC - File Attribute Output

```

27AB C4CA29 12270 CALL NZ,BYTOUT2 ;Display S if so
12280 ;
12290 ; Display an "I" if file is invisible
12300 ;
27AE CB58 12310 BIT 3,B ;Is it an INV file?
27B0 3E49 12320 LD A,'I'
27B2 C4CA29 12330 CALL NZ,BYTOUT2 ;Display I if so
12340 ;
12350 ; Point HL => Password Hash (DIR+16)
12360 ;
27B5 E5 12370 PUSH HL ;Save ptr to 1st dir byte
27B6 7D 12380 LD A,L ;Pt to owner password
27B7 C610 12390 ADD A,16
27B9 6F 12400 LD L,A ;HL => DIR+16
12410 ;
12420 ; Pick up Password in DE
12430 ;
27BA 5E 12440 LD E,(HL) ;P/u in password in DE
27BB 2C 12450 INC L
27BC 56 12460 LD D,(HL)
12470 ;
12480 ; Is this a password protected File ?
12490 ;
27BD E5 12500 PUSH HL ;Save ptr to user psw
27BE 219642 12510 LD HL,BLKHASH ;Init to blanks hash
27C1 ED52 12520 SBC HL,DE ;Is password blanks?
27C3 E1 12530 POP HL
27C4 2814 12540 JR Z,DONAM6 ;Blanks - no "P"assword
12550 ;
12560 ; Password - Display "P" if access <> ALL
12570 ;
27C6 78 12580 LD A,B ;P/u attributes byte
27C7 E607 12590 AND 7 ;Get protection level
27C9 3E50 12600 LD A,'P' ;Init for protected
27CB 200F 12610 JR NZ,DONAM7 ;Stuff the 'P' if prot
12620 ;
12630 ; Access = ALL, Pick up USER password in DE
12640 ;
27CD 2C 12650 INC L ;HL => User Password
27CE 5E 12660 LD E,(HL) ;P/u in DE
27CF 2C 12670 INC L
27D0 56 12680 LD D,(HL)
12690 ;
12700 ; Set A = "P" is User Password <> blank
12710 ;
27D1 219642 12720 LD HL,BLKHASH ;Init to blanks hash
27D4 ED52 12730 SBC HL,DE
27D6 3E50 12740 LD A,'P' ;Init to Protected
27D8 2002 12750 JR NZ,DONAM7 ;Stuff 'P' if acc<>blank
27DA 3E20 12760 DONAM6 LD A,' ' ; else stuff a blank
12770 ;
12780 ; Set Password flag if protected & display "P"
12790 ;
27DC 322728 12800 DONAM7 LD (ALL02+1),A ;Stuff 'P' or blank
27DF FE20 12810 CP ' ' ;Space ?
27E1 C4CA29 12820 CALL NZ,BYTOUT2 ;Display char if needed
27E4 E1 12830 POP HL ;HL => DIR+0
12840 ;
12850 ; Display a "C" if the file was Created

```

LBDIRC - File Attribute Output

```

12860 ;
27E5 23 12870 INC HL ;HL => DIR+1
27E6 7E 12880 LD A,(HL) ;P/u attributes
27E7 2B 12890 DEC HL ;HL => DIR+0
27E8 07 12900 RLCA ;Created ?
27E9 3E43 12910 LD A,'C' ;"C"reate character
27EB DCCA29 12920 CALL C,BYTOUT2 ;Yes - output byte
12930 ;
12940 ; Display Mod flag here if (A) not specified
12950 ;
27EE 3A0628 12960 LD A,(APARM+1) ;P/u A-parm
27F1 B7 12970 OR A
27F2 F5 12980 PUSH AF ;Save condition
27F3 CC3729 12990 CALL Z,OUTMOD ;Output mod flag if -A
27F6 F1 13000 POP AF ;NZ - (A) parm
13010 ;
13020 ; If (A) parameter given - then tab to col 26
13030 ;
27F7 2804 13040 JR Z,DONAM8 ;Not A - go to 20
27F9 3E06 13050 LD A,6 ;Add 6 to column #
27FB 81 13060 ADD A,C
27FC 4F 13070 LD C,A ;C = # of spaces
13080 ;
13090 ; Position to Next designated column
13100 ;
27FD 3E20 13110 DONAM8 LD A,' ' ;Write a space
27FF CDCB29 13120 CALL BYTOUT ;Output byte
2802 0D 13130 DEC C ;Dec column counter
2803 20F8 13140 JR NZ,DONAM8 ;Display trailing spaces
13150 ;
13160 ; Display other things if (A) parm set
13170 ;
2805 11FFFF 13180 APARM LD DE,-1 ;P/u (A) parm
2808 7A 13190 LD A,D ;Specified ?
2809 B3 13200 OR E
280A C42528 13210 CALL NZ,ALL01 ;Full info if A-parm
13220 ;
13230 ; Check for end of line
13240 ;
280D 3E00 13250 DONAM9 LD A,0 ;Count down 4-across
280F 3D 13260 DEC A
2810 320E28 13270 LD (DONAM9+1),A ;Update count
2813 C0 13280 RET NZ ;Loop if more to go
2814 3E04 13290 LD A,4 ; else re-init to 4/line
2816 320E28 13300 LD (DONAM9+1),A
13310 ;
13320 ; Finished with one line - end with C/R
13330 ;
2819 3E0D 13340 ENDLINE LD A,CR ;End line
281B CDCB29 13350 CALL BYTOUT
281E CD652A 13360 CALL CKPAGE ;Check for page pause
2821 CD932A 13370 CALL CKPAWS ;Scan pause or break
2824 C9 13380 RET ;Loop
13390 ;
13400 ; ALL01 - Display Full Allocation of a file
13410 ;
2825 E5 13420 ALL01 PUSH HL ;Save pointer to 1st byte
2826 3E00 13430 ALL02 LD A,0 ;Bypass if not
2828 D620 13440 SUB 20H ; password protected

```

LBDIRC - File Attribute Output

```

282A 2803      13450      JR      Z,ALL03
282C 7E        13460      LD      A,(HL)      ;Get prot level &
282D E607      13470      AND      7          ; multiply by 4
282F 07        13480 ALL03 RLCA          ; to index string array
2830 07        13490      RLCA
2831 4F        13500      LD      C,A
2832 0600      13510      LD      B,0
2834 21462D    13520      LD      HL,PROTS$    ;Pt to 4-char abbrevs
2837 09        13530      ADD     HL,BC      ;Pt to proper one
2838 11A72C    13540      LD      DE,PLEVEL   ;Move into output line
283B 0E04      13550      LD      C,4
283D EDB0      13560      LDIR
283F E1        13570      POP     HL          ;Recover pointer to
2840 E5        13580      PUSH    HL          ; 1st byte of dir record
2841 2C        13590      INC     L
2842 2C        13600      INC     L
2843 2C        13610      INC     L
                13620 ;
                13630 ;      Pick up EOF offset byte & Stuff for later
                13640 ;
2844 7E        13650      LD      A,(HL)      ;P/u EOF offset byte
2845 329628    13660      LD      (EOFBYTE+1),A ;Stuff into LD DE,$-$
                13670 ;
                13680 ;      calculate EOF record according to the formula:
                13690 ;      EOFREC= ((ERN-1)*256+EOF+LRL-1)/LRL if ERN<>0
                13700 ;      EOFREC= 0 if ERN=0
                13710 ;
2848 7E        13720      LD      A,(HL)      ;P/u EOF offset byte
2849 F5        13730      PUSH    AF          ; & save it
284A 2C        13740      INC     L          ;Pt to LRL
284B 7E        13750      LD      A,(HL)      ;P/u LRL
284C 326A28    13760      LD      (ALL04+1),A ; & stuff it
                13770 ;
                13780 ;      get LRL into message
                13790 ;
284F E5        13800      PUSH    HL          ;Save ptr
2850 6F        13810      LD      L,A        ;Transfer LRL to HL
2851 2600      13820      LD      H,0
2853 B7        13830      OR      A          ;Test for <> 256
2854 2001      13840      JR      NZ,$+3
2856 24        13850      INC     H          ;Show 256
2857 11AE2C    13860      LD      DE,LRL     ;DE => LRL destination
285A 3E20      13870      LD      A,' '      ;Init the ASCII byte
285C CD382A    13880      CALL    CVD100     ;Cvrt to msg field
285F E1        13890      POP     HL
                13900 ;
                13910 ;      continue to calculate EOF
                13920 ;
2860 7D        13930      LD      A,L        ;Pt to ERN
2861 C610      13940      ADD     A,16
2863 6F        13950      LD      L,A
2864 5E        13960      LD      E,(HL)     ;P/u into reg DE
2865 2C        13970      INC     L
2866 56        13980      LD      D,(HL)
2867 C1        13990      POP     BC          ;Rcvr EOF byte in reg B
2868 EB        14000      EX      DE,HL     ;Xfer EOFREC -> reg HL
2869 3E00      14010 ALL04 LD      A,0      ;P/u LRL
286B B7        14020      OR      A
286C 2818      14030      JR      Z,TSTSIZ   ;Go use ERN if LRL=0

```

LBDIRC - File Attribute Output

```

286E 5F      14040      LD      E,A      ;Xfer LRL to reg E
286F 04      14050      INC      B      ;Test EOF
2870 05      14060      DEC      B
2871 2801    14070      JR      Z,DONTDEC ;Don't dec ERN if EOF=0
2873 2B      14080      DEC      HL      ;Reduce ERN for 0 offset
2874 CD3029  14090 DONTDEC CALL    DIVIDE
2877 4D      14100      LD      C,L
2878 54      14110      LD      D,H
2879 67      14120      LD      H,A
287A 68      14130      LD      L,B      ;P/u EOF
287B 7B      14140      LD      A,E
287C CD3029  14150      CALL    DIVIDE
287F 61      14160      LD      H,C
2880 B7      14170      OR      A
2881 2801    14180      JR      Z,DONTINC
2883 23      14190      INC      HL      ;Round up partial record
2884 7A      14200 DONTINC LD      A,D      ;Ck if overflow
2885 B7      14210      OR      A
2886 280D    14220 TSTSIZ JR      Z,EOFBYTE ;Use calc'd ERN if not
                14230 ;
                14240 ;      Overflow in # of Records - use "*****"
                14250 ;
2888 21D22D  14260      LD      HL,STARS
288B 11B32C  14270      LD      DE,RECORDS ;DE => Destination
288E 010500  14280      LD      BC,5
2891 EDB0    14290      LDIR
2893 181B    14300      JR      DIR_0
                14310 ;
                14320 ;      If # Records = 0 then set EOF = 0
                14330 ;
2895 110000  14340 EOFBYTE LD      DE,00      ;P/u EOF offset byte
2898 7C      14350      LD      A,H      ;# Records = 0 ?
2899 B5      14360      OR      L
289A 2002    14370      JR      NZ,KEEPEOF ;No - keep EOF
289C 1E01    14380      LD      E,1      ;Set EOF=1 (gets DECed)
289E E5      14390 KEEPEOF PUSH    HL      ;Save # Records
289F 21BB2C  14400      LD      HL,OFFSET ;HL => Destination
28A2 1D      14410      DEC      E      ;DE = EOF byte
28A3 EB      14420      EX      DE,HL      ;Swap for conversion
28A4 3E20    14430      LD      A,' '      ;Init
28A6 CD382A  14440      CALL    CVD100      ;Convert
                14450 ;
                14460 ;      Stuff # of Records used into string
                14470 ;
28A9 E1      14480      POP      HL      ;Recover # of Records
28AA 11B32C  14490      LD      DE,RECORDS ;DE => Destination
28AD CD2A2A  14500      CALL    CVDDEC      ; & stuff into message
                14510 ;
                14520 ;      Get # of extents & Granules used
                14530 ;
28B0 E1      14540 DIR_0 POP      HL      ;Rcvr ptr to 1st byte
28B1 E5      14550      PUSH    HL
28B2 CD4529  14560      CALL    ALL09      ;Get total grans in use
28B5 D5      14570      PUSH    DE
28B6 69      14580      LD      L,C      ;Xfer # extents
28B7 60      14590      LD      H,B
28B8 11CD2C  14600      LD      DE,EXTENTS
28BB 3E20    14610      LD      A,' '
28BD CD3E2A  14620      CALL    CVD10      ;Cvrt to decimal & stuff

```

LBDIRC - File Attribute Output

```

28C0 D1      14630      POP      DE
              14640 ;
              14650 ;      DE = # Grans used - Add to Grans Counter
              14660 ;
28C1 2A0226  14670      LD        HL,(TOTGRNS+1) ;P/u total grans
28C4 19      14680      ADD      HL,DE ;Add this file's count
28C5 220226  14690      LD        (TOTGRNS+1),HL ; & stuff into counter.
              14700 ;
28C8 21C02C  14710      LD        HL,KSIZE ;Pt to where to stuff
28CB CDE829  14720      CALL     CALCK ;Cvrt to K
28CE 21D32C  14730      LD        HL,DATEFLD-1 ;Blank out day-mo-yr
28D1 11D42C  14740      LD        DE,DATEFLD
28D4 010900  14750      LD        BC,9
28D7 EDB0    14760      LDIR
28D9 E1      14770      POP      HL ;Rcvr ptr to DIR+0
28DA 11D42C  14780      LD        DE,DATEFLD
28DD 23      14790      INC      HL
28DE 23      14800      INC      HL ;Advance to date field
28DF 7E      14810      LD        A,(HL)
28E0 B7      14820      OR        A
28E1 2841    14830      JR        Z,ALL08 ;Ignore if no date saved
28E3 0F      14840      RRCA ;Has date, get day
28E4 0F      14850      RRCA
28E5 0F      14860      RRCA
28E6 E61F    14870      AND      1FH
28E8 062F    14880      LD        B,2FH ;Convert day to decimal
28EA 04      14890 ALL06 INC      B ;By counting # of 10's
28EB D60A    14900      SUB      10 ;Sub 10 from day #
28ED 30FB    14910      JR        NC,ALL06
28EF C63A    14920      ADD      A,3AH ;Cvrt lo order to ASCII
28F1 F5      14930      PUSH     AF ;Save day low order
28F2 78      14940      LD        A,B ;Stuff day hi order
28F3 12      14950      LD        (DE),A
28F4 13      14960      INC      DE ;Bump
28F5 F1      14970      POP      AF ;Rcvr lo order day #
28F6 12      14980      LD        (DE),A ;Stuff low order
28F7 13      14990      INC      DE ;Bump pointer to msg
28F8 13      15000      INC      DE ; to pt to month field
28F9 E5      15010      PUSH     HL ;Save DIR ptr
28FA 2B      15020      DEC      HL ;Pt to DIR+1 (month+)
28FB 7E      15030      LD        A,(HL) ;P/u month etc
28FC E60F    15040      AND      0FH ;Strip off flags
28FE 3D      15050      DEC      A ;(mon-1)*3 to index
28FF 4F      15060      LD        C,A ;String conversion table
2900 07      15070      RLCA
2901 81      15080      ADD      A,C
2902 4F      15090      LD        C,A
2903 0600    15100      LD        B,0
2905 21AE2D  15110      LD        HL,MONTBL
2908 09      15120      ADD      HL,BC ;Add offset to table
2909 0E03    15130      LD        C,3
290B EDB0    15140      LDIR ;Move 3-char month
290D 13      15150      INC      DE ;Advance to year field
290E 3E38    15160      LD        A,'8' ;Stuff 8 of 1980
2910 12      15170      LD        (DE),A
2911 13      15180      INC      DE ;Bump msg ptr
2912 E1      15190      POP      HL ;Rcvr DIR+2
2913 E5      15200      PUSH     HL
2914 7E      15210      LD        A,(HL) ;P/u year field

```


LBDIRC - File Attribute Output

```

2915 E607      15220      AND      7      ;Remove day
2917 C630      15230      ADD      A,'0'    ;Cvrt to ASCII
2919 12        15240      LD      (DE),A    ;Stuff -> msg
291A E1        15250      POP      HL      ;Rcvr DIR+2
291B 2B        15260      DEC      HL      ;B/u to DIR+1
291C 3E2D      15270      LD      A,'-'    ;Else change to not cur
291E 32D62C    15280      LD      (DATEFLD+2),A ;Stuff indicator
2921 32DA2C    15290      LD      (DATEFLD+6),A ; between mo&day, day&yr
2924 21A72C    15300 ALL08 LD      HL,PLEVEL ;Pt to start of message
2927 CDB929    15310      CALL     LINOUT    ; & output entire string
292A 3E01      15320      LD      A,1      ;Show only one entry
292C 320E28    15330      LD      (DONAM9+1),A ; per line if A-parm
292F C9        15340      RET

15350 ;
15360 ;      DIVIDE - Divide HL by A
15370 ;
2930 C5        15380 DIVIDE PUSH     BC      ;Save BC
2931 4F        15390      LD      C,A      ;Xfer Divisor in C
2932          15400      @DIV16    ;Divide HL / C
2932 3E5E      00020      LD      A,94
2934 EF        00021      RST      40
2935 C1        15410      POP      BC      ;Restore BC
2936 C9        15420      RET

15430 ;
15440 ;      OUTMOD - Output a "+" if file has been modified
15450 ;
2937 23        15460 OUTMOD INC      HL      ;HL => DIR+1
2938 3E20      15470      LD      A,' '    ;Default to no mod
293A CB76      15480      BIT      6,(HL)   ;Test MOD flag
293C 2802      15490      JR      Z,OUTCHR  ;Output space
293E 3E2B      15500      LD      A,'+'    ;Mod flag char
2940 CDCA29    15510 OUTCHR CALL     BYTOUT2 ;Display '+' if MOD
2943 2B        15520      DEC      HL      ;Repoint to 1st byte
2944 C9        15530      RET

15540 ;
15550 ;
15560 ;      routine calculates total # of grans in use
15570 ;
2945 3AF926    15580 ALL09 LD      A,(SORTPRM+1) ;If sorted, then data
2948 B7        15590      OR      A      ; already calculated
2949 2810      15600      JR      Z,ALL09A ;Go if not sorted
294B E5        15610      PUSH     HL
294C DDE1      15620      POP      IX      ;P/u the saved data
294E DD5E16    15630      LD      E,(IX+22)
2951 DD5617    15640      LD      D,(IX+23) ;P/u Space used
2954 DD4E18    15650      LD      C,(IX+24)
2957 DD4619    15660      LD      B,(IX+25) ;P/u # of extents
295A C9        15670      RET

15680 ;
15690 ;      ALL09A - Calculate space allocated to a file
15700 ;      HL => DIR+0 of an FPDE
15710 ;      BC <= # of Extents in the file
15720 ;      DE <= # of Grans allocated to the file
15730 ;
295B 110000    15740 ALL09A LD      DE,0      ;Init gran counter to 0
295E 43        15750      LD      B,E      ;Init extent ctr to 0
295F 4B        15760      LD      C,E
15770 ;
15780 ;      Point to First Extent of a directory entry

```

LBDIRC - File Attribute Output

```

15790 ;
2960 7D 15800 ALL10 LD A,L ;P/u low byte
2961 C616 15810 ALL11 ADD A,22
2963 6F 15820 LD L,A ;HL => DIR + 22
15830 ;
15840 ; Is the Extent Field in Use ?
15850 ;
2964 7E 15860 ALL14 LD A,(HL) ;P/u cylinder
2965 2C 15870 INC L ;Bump to alloc info
2966 FEFE 15880 CP 0FEH ;Another extent or done ?
2968 300D 15890 JR NC,ALL15 ;Either X'FE' or X'FF'
15900 ;
15910 ; Extent Field is in use - Get allocation info
15920 ;
296A 03 15930 INC BC ;Bump extent counter
296B 7E 15940 LD A,(HL) ;P/u alloc info
296C 2C 15950 INC L ;Bump ptr to next extent
296D E61F 15960 AND 1FH ;Keep # of grans
296F 3C 15970 INC A ;Adj for zero offset
15980 ;
15990 ; A = # of contig grans, add to gran counter
16000 ;
2970 83 16010 ADD A,E ;Accumulate # of grans
2971 5F 16020 LD E,A
2972 30F0 16030 JR NC,ALL14 ;Forget hi if no carry
2974 14 16040 INC D ;Bump hi
2975 18ED 16050 JR ALL14 ;Get next extent field
16060 ;
16070 ; P/u DEC if (X'FE') or RET if done (X'FF')
16080 ;
2977 C0 16090 ALL15 RET NZ ;Ret if not extended
2978 7E 16100 LD A,(HL) ;P/u DEC of FXDE
16110 ;
16120 ; Point HL => Extended Directory Entry posn
16130 ;
2979 E61F 16140 AND 1FH ;Get dir sector of DEC
297B F5 16150 PUSH AF ;Save it
297C AE 16160 XOR (HL) ;Get dir record of FXDE
297D 6F 16170 LD L,A ;Save dir record position
297E F1 16180 POP AF ;Recover DEC of FXDE
16190 ;
16200 ; Is the Dir Sector with FXDE already in mem ?
16210 ;
297F E5 16220 PUSH HL ;Save ptr to 1st extent
2980 215226 16230 LD HL,CKHIT6+1 ;Do we have this dir
2983 BE 16240 CP (HL) ; sector in core?
2984 E1 16250 POP HL ;Restore ptr
2985 2600 16260 SBUFFER LD H,00 ;Buffer hi order
2987 28D7 16270 JR Z,ALL10 ;Jump if we have it
16280 ;
16290 ; Dir Sector not res - Is Ext buf resident ?
16300 ;
2989 FE00 16310 ALL16 CP 0 ;Same as extended area?
298B 262F 16320 LD H,BUF2<-8 ;Pt to extended buf area
298D 28D1 16330 JR Z,ALL10 ;Jump if we have it there
298F 328A29 16340 LD (ALL16+1),A ; else upd the test byte
16350 ;
16360 ; Set B = Directory Entry Code of FXDE
16370 ;

```

LBDIRC - File Attribute Output

```

2992 C5      16380      PUSH    BC          ;Save Gran counter
2993 D5      16390      PUSH    DE          ; & Extent counter
2994 B5      16400      OR      L           ;Combine sector & record
2995 47      16410      LD      B,A         ; pointers to retrieve DEC
                16420 ;
                16430 ;
                16440 ;
                Set C = Logical Drive #, D = Directory Cyl
2996 3A0E2C  16450      LD      A,(DRIVE)    ;P/u ASCII drive #
2999 D630    16460      SUB     '0'         ;Adjust to binary
299B 4F      16470      LD      C,A         ;Save in C
299C FD5609  16480      LD      D,(IY+9)     ;P/u Directory cyl in D
                16490 ;
                16500 ;
                16510 ;
                Set E = FXDE's Dir Sector, HL => I/O buffer
299F 78      16520      LD      A,B         ;P/u DEC
29A0 E61F    16530      AND     1FH        ;Get sector #
29A2 C602    16540      ADD     A,2         ;Adj for GAT & HIT
29A4 5F      16550      LD      E,A         ;Stuff in E
29A5 21002F  16560      LD      HL,BUF2     ;HL => I/O Buffer
                16570 ;
                16580 ;
                16590 ;
                Read in the FXDE's Directory Sector
29A8         16600      @@RDSEC              ;Read a sector
29A8 3E31    00022      LD      A,49
29AA EF      00023      RST     40
29AB FE06    16610      CP      6          ;Expecting Error #6
29AD 3E11    16620      LD      A,11H       ;Read error?
29AF C21E24  16630      JP      NZ,IOERR     ;Jump if got error
                16640 ;
                16650 ;
                16660 ;
                Set A = offset into Sector of entry
29B2 78      16670      LD      A,B         ;P/u FXDE DEC
29B3 E6E0    16680      AND     0E0H       ;Pt to dir record
29B5 D1      16690      POP     DE         ;Restore counters
29B6 C1      16700      POP     BC
29B7 18A8    16710      JR      ALL11       ;Loop through extents
                16720 ;
                16730 ;
                LINOUT - Output line to *DO/*PR
                16740 ;
                HL => Buffer to output
                16750 ;
29B9         16760      LINOUT @@DSPLY        ;Output line to *DO
                00024      IFEQ    00H,1
                00025      LD      HL,
                00026      ENDIF
29B9 3E0A    00027      LD      A,10
29BB EF      00028      RST     40
29BC 2008    16770      JR      NZ,IOER1     ;NZ - Abort
29BE 3AD329  16780      LD      A,(PPARM+1) ;Ck P-parm
29C1 B7      16790      OR      A
29C2 C8      16800      RET     Z          ;Not spec'd - don't print
29C3         16810      @@PRINT              ;Output line to *PR
                00029      IFEQ    00H,1
                00030      LD      HL,
                00031      ENDIF
29C3 3E0E    00032      LD      A,14
29C5 EF      00033      RST     40
29C6 C21E24  16820      IOER1 JP      NZ,IOERR ;NZ - Abort
29C9 C9      16830      RET
                16840 ;

```

LBDIRC - File Attribute Output

```

16850 ;      BYTOUT - Output a byte to *DO/*PR
16860 ;      A = Character to output
16870 ;
29CA 0D      16880 BYTOUT2 DEC      C      ;Decrement col #
29CB C5      16890 BYTOUT  PUSH     BC      ;Save BC
29CC 4F      16900          LD       C,A     ;Save char in C
29CD          16910          @@DSP          ;Display char
29CD 3E02     00034          LD       A,2
29CF EF      00035          RST      40
29D0 20F4     16920          JR       NZ,IOER1 ;NZ - Abort
29D2 110000   16930 PPARAM LD       DE,0    ;P/u P-param
29D5 1C       16940          INC      E      ;Specified ?
29D6 2006     16950          JR       NZ,NOPRT ;No - don't print
29D8          16960          @@PRT          ;Output byte
29D8 3E06     00036          LD       A,6
29DA EF      00037          RST      40
29DB 20E9     16970          JR       NZ,IOER1 ;NZ - Abort
29DD 79       16980          LD       A,C     ;Get back char
29DE C1       16990 NOPRT  POP      BC      ;Restore BC
29DF C9       17000          RET           ;And return
17010 ;
17020 ;      OUTSPC - Output B spaces
17030 ;
29E0 3E20     17040 OUTSPC LD       A,' '    ;Space char
29E2 CDCA29   17050          CALL    BYTOUT2 ;Output space
29E5 10F9     17060          DJNZ    OUTSPC
29E7 C9       17070          RET           ;RETurn
29E8          00300 *GET LBDIRC:3
17080 ;LBDIRC/ASM - DIR math, strings, & buffers
29E8          17090          SUBTTL  '<LBDIRC - Math Routines>'

```

LBDIRC - Math Routines

```

29E8          17100          PAGE
              17110 ;
              17120 ;      CALCK - Calculate the # of K given # of Grans
              17130 ;      DE => # of Granules
              17140 ;      HL => Destination of #K ASCII string
              17150 ;
29E8 22FD29   17160 CALCK LD      (CALCK2+1),HL ;Stuff dest address
              17170 ;
              17180 ;      Calc # of Free Sects (Sectors/Gran x Grans)
              17190 ;
29EB EB       17200          EX      DE,HL          ;HL = # of Free Grans
29EC 0E00     17210 CALCK1 LD      C,$-$          ;C = Sectors/Gran
29EE          17220          @MUL16          ;Mult HL x C
29EE 3E5B     00038          LD      A,91
29F0 EF       00039          RST      40
              17230 ;
              17240 ;      LA = Total # of Sectors - Divide by 4 for K
              17250 ;
29F1 F5       17260          PUSH    AF          ;Save offset
29F2 65       17270          LD      H,L          ;Set HL = LA
29F3 6F       17280          LD      L,A
29F4 CB3C     17290          SRL     H          ;Divide HL / 4
29F6 CB1D     17300          RR      L
29F8 CB3C     17310          SRL     H
29FA CB1D     17320          RR      L
              17330 ;
              17340 ;      P/u dest address & stuff in # of FULL K
              17350 ;
29FC 110000   17360 CALCK2 LD      DE,$-$          ;P/u destination address
29FF CD2A2A   17370          CALL    CVDDEC          ;Stuff in message
2A02 13       17380          INC     DE          ;DE => Hundredths
              17390 ;
              17400 ;      Stuff hundredths value into string
              17410 ;
2A03 F1       17420          POP     AF          ;Rcvr offset to
2A04 E603     17430          AND     3          ;Get offset
2A06 87       17440          ADD     A,A
2A07 0600     17450          LD      B,0
2A09 4F       17460          LD      C,A          ;BC = offset
2A0A 21722D   17470          LD      HL,HUNDTAB      ;HL => Hundredths table
2A0D 09       17480          ADD     HL,BC          ;HL => Hundredths offset
2A0E 0E02     17490          LD      C,2          ;BC = 2 characters
2A10 EDB0     17500          LDIR          ;Transfer to DE
2A12 C9       17510          RET
              17520 ;
              17530 ;      CPHLDE - Compare HL to DE
              17540 ;
2A13 7C       17550 CPHLDE LD      A,H          ;P/u high byte
2A14 BA       17560          CP      D          ;Same ?
2A15 C0       17570          RET     NZ          ;No - Return C or NC
2A16 7D       17580          LD      A,L          ;P/u low byte
2A17 BB       17590          CP      E          ;Less than or greater ?
2A18 C9       17600          RET          ;Return - C, NC, or Z
              17610 ;
              17620 ;      UNPACK - Unpack the Date from a directory entry
              17630 ;      HL => DIR+1
              17640 ;      DE <= Date in DATE$ format
              17650 ;
2A19 7E       17660 UNPACK LD      A,(HL)          ;P/u DIR+1
2A1A E60F     17670          AND     0FH          ;Bits 3-0 contain month

```

LBDIRC - Math Routines

```

2A1C 57      17680      LD      D,A          ;Save month in D
              17690 ;
              17700 ;      D = Month (1-12)
              17710 ;

2A1D 23      17720      INC      HL          ;HL => DIR+2
2A1E 7E      17730      LD      A,(HL)       ;P/u Day/Year byte
2A1F E6F8    17740      AND      0F8H       ;Mask off the year
2A21 5F      17750      LD      E,A         ;Save day in E
2A22 7E      17760      LD      A,(HL)       ;P/u Day/Year byte
2A23 AB      17770      XOR      E          ;Strip off day
2A24 0F      17780      RRCA              ;Year to bits 5-7
2A25 0F      17790      RRCA
2A26 0F      17800      RRCA
2A27 B2      17810      OR      D           ; and merge with month
2A28 57      17820      LD      D,A
2A29 C9      17830      RET

              17840 ;
              17850 ;      conversion of grans to decimal
              17860 ;

2A2A 3E20    17870      CVDDEC LD      A,' '
2A2C 011027  17880      LD      BC,10000    ;Init 1st power or 10
2A2F CD4A2A  17890      CALL     CVD0       ;Make ASCII
2A32 01E803  17900      LD      BC,1000     ;Continue until 1s
2A35 CD4A2A  17910      CALL     CVD0
2A38 016400  17920      CVD100 LD      BC,100
2A3B CD4A2A  17930      CALL     CVD0
2A3E 010A00  17940      CVD10  LD      BC,10
2A41 CD4A2A  17950      CALL     CVD0
2A44 7D      17960      LD      A,L
2A45 C630    17970      ADD     A,30H
2A47 12      17980      LD      (DE),A
2A48 13      17990      INC     DE
2A49 C9      18000      RET
2A4A D5      18010      CVD0  PUSH    DE
2A4B 5F      18020      LD      E,A         ;Fill character
2A4C 16FF    18030      LD      D,0FFH     ;Init to 0
2A4E AF      18040      XOR      A
2A4F 14      18050      CVD1  INC      D         ;Inc digit counter
2A50 ED42    18060      SBC     HL,BC
2A52 30FB    18070      JR      NC,CVD1     ;Sub power/10 til carry
2A54 09      18080      ADD     HL,BC       ;Add back the carry
2A55 7B      18090      LD      A,E
2A56 42      18100      LD      B,D         ;Get count this power
2A57 D1      18110      POP     DE
2A58 12      18120      LD      (DE),A     ;Default char to buffer
2A59 04      18130      INC     B
2A5A 05      18140      DEC     B         ;Was count = 0?
2A5B 2806    18150      JR      Z,CVD2     ;Go if so
2A5D 78      18160      LD      A,B
2A5E C630    18170      ADD     A,30H       ; else make count ASCII
2A60 12      18180      LD      (DE),A     ; and store in buffer
2A61 3E30    18190      LD      A,30H
2A63 13      18200      CVD2  INC     DE         ;Advance buffer
2A64 C9      18210      RET
              18220 ;
              18230 ;      CKPAGE - Check for Page Pause
              18240 ;

2A65 3E00    18250      CKPAGE LD      A,$-$   ;Ck for display pause
2A67 3D      18260      DEC     A          ;Count down

```

LBDIRC - Math Routines

```

2A68 32662A 18270 LD (CKPAGE+1),A ;Update
2A6B C0 18280 RET NZ ;Ret if not yet full
18290 ;
18300 ; Displayed a full page - Reset Counter
18310 ;
2A6C 3E16 18320 LD A,22 ;Reset to max lines/page
2A6E 32662A 18330 LD (CKPAGE+1),A
18340 ;
18350 ; Don't pause if NOPAUSE (N) parm entered
18360 ;
2A71 110000 18370 NPARM LD DE,0 ;P/u NOPAUSE parm
2A74 7B 18380 LD A,E ;Specified ?
2A75 B2 18390 OR D
2A76 C0 18400 RET NZ ;Nonstop if non-zero
18410 ;
18420 ; Non-Stop if <D0> in effect
18430 ;
2A77 3E00 18440 SFLAG LD A,$-$ ;P/u SFLAG$
2A79 E620 18450 AND 20H ;Strip all but <D0> bit
2A7B C0 18460 RET NZ ;Return if do in effect
18470 ;
18480 ; There isn't a <D0> in effect - Wait for key
18490 ;
2A7C 18500 @KEY ;Wait for key entry
2A7C 3E01 00040 LD A,1
2A7E EF 00041 RST 40
2A7F C21E24 18510 IOERR5 JP NZ,IOERR
18520 ;
18530 ; Clear Screen
18540 ;
2A82 18550 @CLS ;Clear Screen
2A82 3E69 00042 LD A,105
2A84 EF 00043 RST 40
2A85 20F8 18560 JR NZ,IOERR5
18570 ;
18580 ; If the NOTITLE flag is set - don't display
18590 ;
2A87 3E00 18600 NOTITLE LD A,$-$ ;P/u flag
2A89 B7 18610 OR A ;No title ?
2A8A C0 18620 RET NZ ;Then RETurn
18630 ;
18640 ; Display a title if there were matching files
18650 ;
2A8B 3AD82D 18660 LD A,(FILFLAG) ;Was a matching file
2A8E B7 18670 OR A ; displayed ?
2A8F C4B12A 18680 CALL NZ,CKTITL ;Yes - display title
2A92 C9 18690 RET ;Return
18700 ;
18710 ; CKPAWS - Check for <SHIFT><@> or <BREAK>
18720 ;
18730 CKPAWS
18740 ;
18750 ; Was the <BREAK> key hit ?
18760 ;
2A93 3A0000 18770 KFLAG LD A,($-$) ;P/u KFLAG$
2A96 0F 18780 RRCA ;<BREAK> hit ?
2A97 DA1724 18790 JP C,ABORT ;Yes - cease DIR
18800 ;
18810 ; Is the <PAUSE> bit set ?

```

LBDIRC - Math Routines

```

18820 ;
2A9A 0F      18830      RRCA                ;<PAUSE> bit set ?
2A9B D0      18840      RET      NC          ;Ret if not pause
18850 ;
18860 ;      The <PAUSE> bit is set - Wait for Char
18870 ;
2A9C      18880 CKPAW1  @@KEY                ;Scan keyboard
2A9C 3E01    00044      LD      A,1
2A9E EF      00045      RST      40
18890 ;
18900 ;      Character entered - Ignore it if <SHIFT><0>
18910 ;
2A9F FE60    18920 CKPAW2  CP      60H          ;<SHIFT><0> ?
2AA1 28F9    18930      JR      Z,CKPAW1      ;Yes - get another char
2AA3 FE80    18940      CP      BREAK
2AA5 CA1724  18950      JP      Z,ABORT
18960 ;
18970 ;      Reset <PAUSE> & <ENTER> bits
18980 ;
2AA8 3A0000  18990 RESKFL  LD      A,($-$)      ;P/u KFLAG$
2AAB E6F9    19000      AND      0F9H          ;Reset <PAUSE> & <ENTER>
2AAD 320000  19010 KFLAG1 LD      ($-$),A      ;Stuff into KFLAG$
2AB0 C9      19020      RET                    ; & RETURN
19030 ;
19040 ;      CKTITL - Display Title
19050 ;
19060 ;      Display Disk type Header
19070 ;
2AB1 21072C  19080 CKTITL LD      HL,DSTRING    ;HL => Heading
2AB4 CDB929  19090      CALL     LINOUT          ;Output line
2AB7 CD652A  19100      CALL     CKPAGE          ;Bump line count
2ABA CD652A  19110      CALL     CKPAGE          ; twice.
19120 ;
19130 ;      Display Attributes header if A parm spec'd
19140 ;
2ABD 3A0628  19150      LD      A,(APARM+1)      ;Was the A parm spec'd
2AC0 B7      19160      OR      A
2AC1 3E0D    19170      LD      A,CR            ;Output a CR if A
2AC3 CACB29  19180      JP      Z,BYTOUT        ; not specified.
19190 ;
2AC6 21572C  19200      LD      HL,HEADING      ;HL => Attr heading
2AC9 CDB929  19210      CALL     LINOUT          ;Output line
19220 ;
19230 ;      Display Underline
19240 ;
2ACC C5      19250      PUSH     BC              ;Save BC
2ACD 064F    19260      LD      B,79            ;Display underline
2ACF 3E2D    19270 D79L  LD      A,'-'
2AD1 CDCB29  19280      CALL     BYTOUT          ;Output byte
2AD4 10F9    19290      DJNZ     D79L            ; 79 times
2AD6 C1      19300      POP      BC              ;Restore BC
2AD7 3E0D    19310      LD      A,CR            ;One CR between
2AD9 CDCB29  19320      CALL     BYTOUT
2ADC C3652A  19330      JP      CKPAGE          ;Check page pause & RET
19340 ;
2ADF      19350      SUBTTL  '<LBDIRC - Sort Code>'
19360 ;
19370 ;      SORTIT - Set up Directory Records for Shell Sort
19380 ;

```


LBDIRC - Sort Code

```

2ADF 2AD92D 19390 SORTIT LD HL,(DIRPTR) ;Calculate # of records
2AE2 110030 19400 LD DE,MEMORY ;Point to buf start
2AE5 73 19410 LD (HL),E ;Prime the 1st index
2AE6 23 19420 INC HL ; in case there is
2AE7 72 19430 LD (HL),D ; only one record
2AE8 2B 19440 DEC HL ; to sort
2AE9 AF 19450 XOR A
2AEA ED52 19460 SBC HL,DE ;PTREND - PTRBGN
2AEC C8 19470 RET Z ;Ret if nothing
19480 ;
19490 ; Set HL = # of directory entries
19500 ;
2AED 0605 19510 LD B,5 ;Divide by
2AEF CB3C 19520 SORT1 SRL H ; 32 bytes/record
2AF1 CB1D 19530 RR L
2AF3 10FA 19540 DJNZ SORT1
19550 ;
19560 ; Set B = # of entries & init count
19570 ;
2AF5 45 19580 LD B,L ;Set loop counter
2AF6 C5 19590 PUSH BC ;Save it for printing
2AF7 22332B 19600 LD (COUNTM1),HL ;Init the count
19610 ;
19620 ; Skip sort if # of entries = 0
19630 ;
2AFA 7C 19640 LD A,H ;If length = 0
2AFB B5 19650 OR L ; then no need to sort
2AFC 2821 19660 JR Z,SORT2A
2AFE 29 19670 ADD HL,HL ;Make sure enuff room
2AFF EB 19680 EX DE,HL
2B00 2ADB2D 19690 LD HL,(MAXMEM)
2B03 AF 19700 XOR A
2B04 ED52 19710 SBC HL,DE
2B06 DA0D24 19720 JP C,NOMEM
2B09 2AD92D 19730 LD HL,(DIRPTR) ;Set up the index array
2B0C 110030 19740 LD DE,MEMORY ;Starting record pointer
2B0F 73 19750 SORT2 LD (HL),E ;Place record pointers
2B10 23 19760 INC HL ; into index array
2B11 72 19770 LD (HL),D
2B12 23 19780 INC HL
2B13 7B 19790 LD A,E ;Increment pointer by 32
2B14 C620 19800 ADD A,32
2B16 5F 19810 LD E,A
2B17 3001 19820 JR NC,$+3 ;Go if no overflow
2B19 14 19830 INC D ; else bump high order
2B1A 10F3 19840 DJNZ SORT2 ;Loop for all records
2B1C CD322B 19850 CALL SHELL ;Sort the dir records
2B1F C1 19860 SORT2A POP BC ;Recover loop counter
2B20 2AD92D 19870 LD HL,(DIRPTR) ;P/u starting record
2B23 5E 19880 SORT3 LD E,(HL) ;Grab its address
2B24 23 19890 INC HL
2B25 56 19900 LD D,(HL)
2B26 23 19910 INC HL
2B27 E5 19920 PUSH HL ;Save index pointer
2B28 C5 19930 PUSH BC ;Save loop counter
2B29 EB 19940 EX DE,HL ;Record address -> HL
2B2A CD3327 19950 CALL MATCH ;Display the record
2B2D C1 19960 POP BC ;Rcvr loop counter
2B2E E1 19970 POP HL ;Rcvr index pointer

```

LBDIRC - Sort Code

```

2B2F 10F2      19980      DJNZ      SORT3
2B31 C9        19990      RET
                20000 ;
                20010 ;      SHELL - Shell Sort Routine
                20020 ;
2B32 210000    20030 SHELL LD      HL,$-$      ;P/u count minus 1
2B33          20040 COUNTM1 EQU     $-2
2B35 22392B    20050 LD      (STORM),HL
                20060 ;
                20070 ;      Start Select & Compare
                20080 ;
2B38 110000    20090 CYCLE LD      DE,0      ;M = M / 2
2B39          20100 STORM EQU     $-2
2B3B CB3A      20110 SRL      D
2B3D CB1B      20120 RR      E
2B3F 7A        20130 LD      A,D      ;Return when M=0
2B40 B3        20140 OR      E
2B41 C8        20150 RET      Z
2B42 ED53392B  20160 LD      (STORM),DE
2B46 2A332B    20170 LD      HL,(COUNTM1) ;K = N - M
2B49 ED52      20180 SBC      HL,DE
2B4B 22BA2B    20190 LD      (STORK),HL
2B4E 210000    20200 LD      HL,0      ;J = 0
2B51 22552B    20210 LD      (STORJ),HL
2B54 210000    20220 AGAIN LD      HL,$-$    ;I = J
2B55          20230 STORJ EQU     $-2
2B57 225B2B    20240 LD      (STORI),HL
2B5A 210000    20250 REPEAT LD      HL,$-$    ;L = I + M
2B5B          20260 STORI EQU     $-2
2B5D ED5B392B  20270 LD      DE,(STORM)
2B61 19        20280 ADD      HL,DE
2B62 29        20290 ADD      HL,HL      ;L * 2 -> regHL
2B63 E5        20300 PUSH     HL      ;Save L
2B64 2A5B2B    20310 LD      HL,(STORI) ;I * 2 -> regHL
2B67 29        20320 ADD      HL,HL
2B68 ED4BD92D  20330 LD      BC,(DIRPTR) ;P/u string parm ptr
2B6C 09        20340 ADD      HL,BC      ;Pt to A$(I) parm
2B6D EB        20350 EX      DE,HL      ;Ptr -> DE
2B6E E1        20360 POP      HL      ;Pt to A$(L) parm
2B6F 09        20370 ADD      HL,BC      ;Ptr -> HL
2B70 E5        20380 PUSH     HL      ;Save ptr to A$(L)
2B71 D5        20390 PUSH     DE      ;Save ptr to A$(I)
2B72 060B      20400 LD      B,11.    ;Set compare length
2B74 C5        20410 PUSH     BC      ;Save cpr len & flag
2B75 7E        20420 LD      A,(HL)    ;P/u string2 ptr
2B76 23        20430 INC      HL
2B77 66        20440 LD      H,(HL)
2B78 6F        20450 LD      L,A
2B79 010500    20460 LD      BC,5      ;Key is 5 bytes in
2B7C 09        20470 ADD      HL,BC
2B7D EB        20480 EX      DE,HL      ;String2 ptr -> rDE
2B7E 7E        20490 LD      A,(HL)    ;P/u string1 ptr
2B7F 23        20500 INC      HL
2B80 66        20510 LD      H,(HL)
2B81 6F        20520 LD      L,A
2B82 09        20530 ADD      HL,BC      ;Key is 5 bytes in
2B83 C1        20540 POP      BC      ;Rcvr len & flag
2B84 1A        20550 BACK LD      A,(DE) ;Go swap if str1>str2
2B85 96        20560 SUB      (HL)

```

LBDIRC - Sort Code

```

2B86 3808      20570      JR      C,POP
2B88 2025      20580      JR      NZ,FINIS      ;Next str if str2>str1
2B8A 13        20590      INC      DE      ;Loop if this matches
2B8B 23        20600      INC      HL
2B8C 10F6      20610      DJNZ     BACK
2B8E 181F      20620      JR      FINIS      ;None really should match
2B90 D1        20630 POP     POP     DE      ;Else swap
2B91 E1        20640      POP     HL
2B92 0602      20650      LD      B,2      ;Swap 2-byte
2B94 4E        20660 SWAP   LD      C,(HL)    ;String pointer
2B95 EB        20670      EX      DE,HL
2B96 7E        20680      LD      A,(HL)
2B97 71        20690      LD      (HL),C
2B98 EB        20700      EX      DE,HL
2B99 77        20710      LD      (HL),A
2B9A 23        20720      INC      HL
2B9B 13        20730      INC      DE
2B9C 10F6      20740      DJNZ     SWAP
2B9E 2A392B    20750      LD      HL,(STORM)    ;P/u M
2BA1 EB        20760      EX      DE,HL
2BA2 2A5B2B    20770      LD      HL,(STORI)    ;P/u I
2BA5 AF        20780      XOR      A
2BA6 ED52      20790      SBC      HL,DE
2BA8 225B2B    20800      LD      (STORI),HL    ;I = I - M
2BAB 30AD      20810      JR      NC,REPEAT    ;Repeat if I => 0
2BAD 1802      20820      JR      EXITSRT    ;Else exit the loop
2BAF D1        20830 FINIS  POP     DE
2BB0 E1        20840      POP     HL
2BB1 2A552B    20850 EXITSRT LD      HL,(STORJ)
2BB4 23        20860      INC      HL      ;J = J + 1
2BB5 22552B    20870      LD      (STORJ),HL
2BB8 AF        20880      XOR      A
2BB9 110000    20890      LD      DE,$-$
2BBA          20900 STORK   EQU      $-2
2BBC ED52      20910      SBC      HL,DE      ;J - K
2BBE D2382B    20920      JP      NC,CYCLE    ;Cycle if J => K *
2BC1 C3542B    20930      JP      AGAIN     ;Else again
          20940 ;
2BC4          20950      SUBTTL  '<LBDIRC - Data>'
          20960 ;
2BC4 80        20970 PRMTBL$ DB      80H      ;6.x parameters
          20980 ;
          20990 ;      A - Flag input only
          21000 ;
2BC5 41        21010      DB      FLAG!1
2BC6 41        21020      DB      'A'
2BC7 00        21030      DB      0
2BC8 0628      21040      DW      APARM+1
          21050 ;
          21060 ;      INV (I) - Flag input only
          21070 ;
2BCA 53        21080      DB      FLAG!ABB!3
2BCB 49        21090      DB      'INV'
          4E 56
2BCE 00        21100      DB      0
2BCF 8D26      21110      DW      IPARM+1
          21120 ;
          21130 ;      P - Flag input only
          21140 ;

```

LBDIRC - Data

2BD1 41	21150	DB	FLAG!1
2BD2 50	21160	DB	'P'
2BD3 00	21170	DB	0
2BD4 D329	21180	DW	PPARM+1
	21190 ;		
	21200 ;		
	21210 ;		
			SYS (S) - Flag input only
2BD6 53	21220	DB	FLAG!ABB!3
2BD7 53	21230	DB	'SYS'
59 53			
2BDA 00	21240	DB	0
2BDB 7F26	21250	DW	SPARM+1
	21260 ;		
	21270 ;		
	21280 ;		
			N - Flag input only
2BDD 41	21290	DB	FLAG!1
2BDE 4E	21300	DB	'N'
2BDF 00	21310	DB	0
2BE0 722A	21320	DW	NPRM+1
	21330 ;		
	21340 ;		
	21350 ;		
			DATE (D) - Flag or String input
2BE2 74	21360	DB	FLAG!STR!ABB!4
2BE3 44	21370	DB	'DATE'
41 54 45			
2BE7 00	21380	DB	0
2BE8 982E	21390	DW	DATPRM+1
	21400 ;		
	21410 ;		
	21420 ;		
			MOD (M) - Flag input only
2BEA 53	21430	DB	FLAG!ABB!3
2BEB 4D	21440	DB	'MOD'
4F 44			
2BEE 00	21450	DB	0
2BEF 9526	21460	DW	CKMOD+1
	21470 ;		
	21480 ;		
	21490 ;		
			SORT (O) - Flag input only
2BF1 44	21500	DB	FLAG!4
2BF2 53	21510	DB	'SORT'
4F 52 54			
2BF6 00	21520	DB	0
2BF7 F926	21530	DW	SORTPRM+1
	21540 ;		
2BF9 41	21550	DB	FLAG!1
2BFA 4F	21560	DB	'O'
2BFB 00	21570	DB	0
2BFC F926	21580	DW	SORTPRM+1
	21590 ;		
	21600 ;		
2BFE 00	21610	DB	0
	21620 ;		
2BFF 78	21630	DB	'xDEN'
44 45 4E			
2C03 48	21640	DB	'Hard'
61 72 64			
	21650 ;		
2C07 44	21660	DB	'Drive :'
72 69 76 65 20 3A			

LBDIRC - Data

2C0E 64	21670 DRIVE DB	'd '
20 20		
2C11 64	21680 NAME DB	'diskname '
69 73 6B 6E 61 6D 65 20		
20		
2C1B 20	21690 CYLCNT DB	' Cyl, '
20 20 20 43 79 6C 2C 20		
2C24 6E	21700 DENSITY DB	'nDEN, Free ='
44 45 4E 2C 20 46 72 65		
65 20 3D		
2C30 20	21710 KFREE DB	' . K / '
20 20 20 20 2E 20 20 4B		
20 2F 20		
2C3C 20	21720 KPOSS DB	' . K, Date '
20 20 20 20 2E 20 20 4B		
2C 20 20 44 61 74 65 20		
2C4D 64	21730 DATBUF DB	'dd-mmm-yy',CR
64 2D 6D 6D 6D 2D 79 79		
0D		
	21740 ;	
2C57 46	21750 HEADING DB	'Filespec MOD Attr Prot LRL'
69 6C 65 73 70 65 63 20		
20 20 20 4D 4F 44 20 20		
20 41 74 74 72 20 20 20		
50 72 6F 74 20 20 20 4C		
52 4C		
2C7A 20	21760 DB	' #Recs EOF File Size Ext Mod '
20 23 52 65 63 73 20 20		
20 45 4F 46 20 20 46 69		
6C 65 20 53 69 7A 65 20		
20 20 45 78 74 20 20 20		
20 20 4D 6F 64 20		
2CA1 44	21770 DB	'Date ',CR
61 74 65 20 0D		
	21780 ;	
2CA7 50	21790 PLEVEL DB	'Prot '
72 6F 74 20 20 20		
2CAE 20	21800 LRL DB	' '
20 20 20 20		
2CB3 20	21810 RECORDS DB	' '
20 20 20 20 20 20 20		
2CBB 20	21820 OFFSET DB	' '
20 20 20 20		
2CC0 20	21830 KSIZE DB	' . K '
20 20 20 20 2E 20 20 4B		
20 20 20 20		
2CCD 20	21840 EXTENTS DB	' '
20 20 20 20 20 20		
2CD4 64	21850 DATEFLD DB	'dd-mmm-yy',ETX
64 2D 6D 6D 6D 2D 79 79		
03		
	21860 ;	
2CDE 20	21870 FSTRING DB	' '
20 20 20 20 20 20 20		
20 20		
2CE9 20	21880 FDISP DB	' files out of '
20 20 20 66 69 6C 65 73		
20 6F 75 74 20 6F 66 20		
2CFA 20	21890 FUSED DB	' selected, Space = '

LBDIRC - Data

```

      20 20 20 73 65 6C 65 63
      74 65 64 2C 20 53 70 61
      63 65 20 3D 20
2D10 20      21900 SPUSED DB      '      . K',LF,CR
      20 20 20 20 2E 20 20 4B
      0A 0D
      21910 ;
2D1B 44      21920 NODISK DB      'Drive : '
      72 69 76 65 20 3A
2D22 6E      21930 NDRIVE DB      'n [No Disk]',LF,CR
      20 20 5B 4E 6F 20 20 44
      69 73 6B 5D 0A 0D
      21940 ;
2D31 6D      21950 TDATE DB      'mm/dd/yy" '
      6D 2F 64 64 2F 79 79 22
2D3A 20      21960 BLANKS DB      '
      20 20 20 20 20 20 20 20
      20 20 20
2D46 46      21970 PROTS$ DB      'FULLREMVNAMEWRITUPDTREADEXECNO '
      55 4C 4C 52 45 4D 56 4E
      41 4D 45 57 52 49 54 55
      50 44 54 52 45 41 44 45
      58 45 43 4E 4F 20 20
2D66 1F      21980 MAXDAYS DB      31,28,31,30,31,30,31,31,30,31,30,31
      1C 1F 1E 1F 1E 1F 1F 1E
      1F 1E 1F
2D72 30      21990 HUNDTAB DB      '00255075'
      30 32 35 35 30 37 35
2D7A 49      22000 NOMEM$ DB      'Insufficient memory for SORT buffer',CR
      6E 73 75 66 66 69 63 69
      65 6E 74 20 6D 65 6D 6F
      72 79 20 66 6F 72 20 53
      4F 52 54 20 62 75 66 66
      65 72 0D
2D9E 42      22010 BADFMT$ DB      'Bad date format',CR
      61 64 20 64 61 74 65 20
      66 6F 72 6D 61 74 0D
2DAE 4A      22020 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
2DD2 2A      22030 STARS DB      '*****'
      2A 2A 2A 2A
2DD7 00      22040 FTFLG DB      0
2DD8 00      22050 FILFLAG DB      0
2DD9      22060 DIRPTR EQU      $
2DDB      22070 MAXMEM EQU      DIRPTR+2
2DDD      22080 FMPAKD EQU      MAXMEM+2
2DDF      22090 TOPAKD EQU      FMPAKD+2
2DE1      22100 LILBUF$ EQU      TOPAKD+2
      22110 ;
      22120 ;
      22130 ;
2E00      22140 GAT EQU      LILBUF$+3<-8+1<+8
2E00      22150 HIT EQU      GAT
2F00      22160 BUF2 EQU      GAT+256
3000      22170 MEMORY EQU      GAT+512

```

LBDIRC - Data

```
22180 ;  
2DD9 22190 SUBTTL '<LBDIRC - Initialization Code>'
```

LBDIRC - Initialization Code

```

2DD9          22200          PAGE
              22210 ;
              22220 ;      DIR Entry Point - Initialization code
              22230 ;
              22240 DIR
2DD9          22250          @@CKBRKC          ;Check for break
2DD9 3E6A     00046          LD      A,106
2DD8 EF       00047          RST      40
2DDC 2804     22260          JR      Z,DIRA          ;If not go
2DDE 21FFFF   22270          LD      HL,-1          ; else abort
2DE1 C9       22280          RET
              22290 ;
              22300 DIRA
2DE2 ED73D24 22310          LD      (SAVE SP+1),SP ;Save SP address
2DE6 E5       22320          PUSH    HL          ;Save command ptr
              22330 ;
              22340 ;      Pick up Flag Table base Address
              22350 ;
2DE7          22360          @@FLAGS          ;IY => System Flag table
2DE7 3E65     00048          LD      A,101
2DE9 EF       00049          RST      40
2DEA FDE5     22370          PUSH    IY          ;Xfer to DE too
2DEC D1       22380          POP      DE
              22390 ;
              22400 ;      Calculate KFLAG$ address & stuff away
              22410 ;
2DED 210A00   22420          LD      HL,KFLAG$      ;KFLAG$ offset
2DF0 19       22430          ADD     HL,DE          ;HL => KFLAG$
2DF1 22942A   22440          LD      (KFLAG+1),HL   ;Save for later testing
2DF4 22A92A   22450          LD      (RESKFL+1),HL
2DF7 22AE2A   22460          LD      (KFLAG1+1),HL
              22470 ;
2DFA CDA82A   22480          CALL    RESKFL        ;Reset bits 0-2 of KFLAG$
2DFD E1       22490          POP      HL          ;Rvr command ptr
              22500 ;
              22510 ;      Pick up SFLAG
              22520 ;
2DFE FD7E12   22530          LD      A,(IY+'S'-'A') ;Get SFLAG
2E01 32782A   22540          LD      (SFLAG+1),A    ;Save for later testing
              22550 ;
              22560 ;      Find parameter entry if existent
              22570 ;
2E04 E5       22580          PUSH    HL          ;Save command ptr
2E05 7E       22590 FPLP     LD      A,(HL)          ;P/u character
2E06 FE28     22600          CP      '('          ;Parameter(s) ?
2E08 2807     22610          JR      Z,GETPRM        ;Yes - go get 'em
2E0A FE0D     22620          CP      CR          ;End of line ?
2E0C 2809     22630          JR      Z,RESTPTR        ;Yes - restore ptr
2E0E 23       22640          INC     HL          ;No - bump til end
2E0F 18F4     22650          JR      FPLP          ;Do til eol or "("
              22660 ;
              22670 ;      Process any parameters entered
              22680 ;
2E11 11C42B   22690 GETPRM  LD      DE,PRMTBL$      ;DE => Parameter table
2E14          22700          @@PARAM          ;@PARAM
2E14 3E11     00050          LD      A,17
2E16 EF       00051          RST      40
2E17 E1       22710 RESTPTR  POP      HL          ;Recover ptr
2E18 C21E24   22720          JP      NZ,IOERR        ;NZ - "Parameter Error"
              22730 ;

```


LBDIRC - Initialization Code

```

22740 ;      If first character is a "8" or "9" abort
22750 ;
2E1B 7E      22760 LD      A,(HL)      ;Is this a "8" or "9" ?
2E1C FE0D    22770 CP      CR        ;If CR, then global
2E1E 2834    22780 JR      Z,DIR2
2E20 FE38    22790 CP      '8'       ;If so - Illegal drive #
2E22 2804    22800 JR      Z,ILLDRV
2E24 FE39    22810 CP      '9'
2E26 2003    22820 JR      NZ,CKITOUT ;Must be a partspec
22830 ;
22840 ;      Illegal Drive Number
22850 ;
2E28 C31C24  22860 ILLDRV JP      ERR32 ;Go to I/O error handler
22870 ;
22880 ;      Pick up Drive # Range field if any
22890 ;
2E2B E5      22900 CKITOUT PUSH   HL      ;Save source ptr
2E2C CDE72E  22910 CALL   CKDSPEC ;Legal Drive range ?
2E2F D1      22920 POP     DE        ;Save source ptr in DE
2E30 2832    22930 JR      Z,DIR3     ;Legal - use HL
22940 ;
22950 ;      Point DE => Partspec match field, B=8 chars
22960 ;
2E32 EB      22970 EX      DE,HL      ;Illegal - use DE
2E33 7E      22980 LD      A,(HL)    ;P/u first char
2E34 23      22990 INC     HL        ; and bump to next
2E35 113A2D  23000 LD      DE,BLANKS ;DE => Partspec area
2E38 0608    23010 LD      B,8       ;B = 8 chars/filename
23020 ;
23030 ;      Was the NOT switch entered ?
23040 ;
2E3A FE2D    23050 CP      '-'       ;NOT ?
2E3C 2005    23060 JR      NZ,DIR1    ;No - continue
23070 ;
23080 ;      NOT "-" entered - set flag & bump cmd ptr
23090 ;
2E3E 32C426  23100 LD      (MFLG+1),A ;Stuff "-" in flag
2E41 7E      23110 LD      A,(HL)    ;P/u next char & bump
2E42 23      23120 INC     HL        ; command ptr
23130 ;
23140 ;      Transfer Filename to Filespec buffer
23150 ;
2E43 CD2A2F  23160 DIR1  CALL   PRSPC ;Parse 8 chars
2E46 FE2F    23170 CP      '/'       ;Extension ?
2E48 200A    23180 JR      NZ,DIR2    ;No - don't check
23190 ;
23200 ;      Transfer Extension to Filespec buffer
23210 ;
2E4A 11422D  23220 LD      DE,BLANKS+8 ;DE => Extension field
2E4D 0603    23230 LD      B,3       ;Max 3 chars
2E4F 7E      23240 LD      A,(HL)    ;P/u next character
2E50 23      23250 INC     HL        ;Bump
2E51 CD2A2F  23260 CALL   PRSPC ;Xfer extension
23270 ;
23280 ;      Was a drivespec entered ?
23290 ;
2E54 FE3A    23300 DIR2  CP      ':'     ;Drive entered?
2E56 010700  23310 LD      BC,7      ;St = 0, terminating = 7
2E59 2009    23320 JR      NZ,DIR3    ;No - use drive 0

```

LBDIRC - Initialization Code

```

23330 ;
23340 ;      Check if char following is a legal drive #
23350 ;
2E5B CDE72E 23360 CALL CKDSPEC      ;Legal Drive field ?
2E5E 20C8    23370 JR      NZ,ILLDRV ;Illegal - abort
2E60 FE08    23380 CP      8          ;Trap DIR :8
2E62 28C4    23390 JR      Z,ILLDRV
23400 ;
23410 ;      B = Start drv #, C = Term drv # - save 'em
23420 ;
2E64 78      23430 DIR3 LD      A,B          ;Save starting drive
2E65 32DE2E 23440 LD      (DIR3A+1),A
2E68 91      23450 SUB      C          ;Set Specific Drive flag
2E69 323626 23460 LD      (SPECIF+1),A
2E6C 79      23470 LD      A,C          ;Save term drive
2E6D 322B26 23480 LD      (TERMDRV+1),A
23490 ;
23500 ;      Command line parsed - check available mem
23510 ;
2E70 FDCB024E 23520 BIT      1,(IY+CFLAG$) ;Called from @CMNDR?
2E74 210000 23530 LD      HL,0          ;Set SORT (0) parm = 0
2E77 2803    23540 JR      Z,GETHI      ;No - fine
23550 ;
23560 ;      Executing from @CMNDR - Turn off SORT
23570 ;
2E79 22F926 23580 LD      (SORTPRM+1),HL
23590 ;
23600 ;      Pick up Current HIGH$, & set max mem to use
23610 ;
2E7C 45      23620 GETHI LD      B,L          ;B=0
2E7D          23630 @@HIGH$
2E7D 3E64    00052 LD      A,100
2E7F EF      00053 RST      40
2E80 11DFFF 23640 LD      DE,-33          ;Subtract 33 from it
2E83 19      23650 ADD      HL,DE
2E84 22DB2D 23660 LD      (MAXMEM),HL      ;Stuff in maximum memory
23670 ;
23680 ;      Turn on N parm if P parm specified
23690 ;
2E87 2AD329 23700 LD      HL,(PPARM+1) ;P/u P-parm
2E8A 7C      23710 LD      A,H          ;Specified ?
2E8B B5      23720 OR      L
2E8C 2803    23730 JR      Z,GTDATE      ;No - don't change N
2E8E 22722A 23740 LD      (NPARM+1),HL ;Turn on N-parm
23750 ;
23760 ;      Was the DATE parameter specified ?
23770 ;
2E91 3AE72B 23780 GTDATE LD      A,(DRESP) ;Check out response
2E94 B7      23790 OR      A          ;Any response ?
2E95 2846    23800 JR      Z,DIR3A      ;None entered - no date
23810 ;
23820 ;      Something was specified - Check type
23830 ;
2E97 210000 23840 DATPRM LD      HL,$-$      ;P/u date
2E9A CB77    23850 BIT      6,A          ;Flag input ?
2E9C 280C    23860 JR      Z,CHKSTR      ;No - must be string
23870 ;
23880 ;      Flag input - if YES, then use today's date
23890 ;

```

LBDIRC - Initialization Code

```

2E9E 7C      23900 LD      A,H          ;DATE = OFF ?
2E9F B5      23910 OR      L
2EA0 283B    23920 JR      Z,DIR3A      ;Yes - ignore it
          23930 ;
          23940 ;      DATE parameter entered - get today's date
          23950 ;
2EA2 21312D  23960 LD      HL,TDAT      ;HL => Todays Date
2EA5 E5      23970 PUSH    HL          ;Save position
2EA6         23980 @@DATE      ;Get today's date
2EA6 3E12    00054 LD      A,18
2EA8 EF      00055 RST      40
2EA9 E1      23990 POP      HL          ;HL => Today's Date
          24000 ;
          24010 ;      Display dates before "-mm/dd/yy" ?
          24020 ;
2EAA 7E      24030 CHKSTR LD      A,(HL)    ;P/u first char
2EAB FE2D    24040 CP      '-'          ;"to-" ?
2EAD 2815    24050 JR      Z,CKTO      ;Yes - do it
          24060 ;
          24070 ;      Not before - set flag accordingly
          24080 ;
2EAF 3E80    24090 LD      A,80H        ;Set from bit
2EB1 32D72D  24100 LD      (FTFLG),A      ;Note from entered
          24110 ;
          24120 ;      Pack Date entry
          24130 ;
2EB4 CD492F  24140 CALL    PAKDAT      ;Pack the date entry
2EB7 ED43DD2D 24150 LD      (FMPAKD),BC  ;Stuff away date
          24160 ;
          24170 ;      End of first date ?
          24180 ;
2EBB 7E      24190 LD      A,(HL)        ;P/u terminator
2EBC FE22    24200 CP      '-'          ;End of date ?
2EBE 2811    24210 JR      Z,FRCTO      ;Yes - use spec'd date
          24220 ;
          24230 ;      Is there a to "-" symbol following date ?
          24240 ;
2EC0 FE2D    24250 CP      '-'          ;Check for "-to"
2EC2 2019    24260 JR      NZ,DIR3A      ;No - check if legal
          24270 ;
          24280 ;      Is there a date following ?
          24290 ;
2EC4 23      24300 CKTO INC      HL          ;Bypass the '-'
2EC5 7E      24310 LD      A,(HL)        ;P/u next char
2EC6 FE22    24320 CP      '-'          ;End of parm ?
2EC8 2813    24330 JR      Z,DIR3A      ;Yes - use that date
          24340 ;
2ECA FE0D    24350 CP      CR          ;End of parm ?
2ECC 280F    24360 JR      Z,DIR3A      ;Yes - use that date
          24370 ;
          24380 ;      Something following - parse date
          24390 ;
2ECE CD492F  24400 CALL    PAKDAT      ;Pack Date
          24410 ;
          24420 ;      Stuff in "TO" packed date & set TO flag
          24430 ;
2ED1 3AD72D  24440 FRCTO LD      A,(FTFLG) ;P/u From-To Flag
2ED4 F601    24450 OR      1          ;Set TO bit
2ED6 32D72D  24460 LD      (FTFLG),A      ;Stuff in flag

```

LBDIRC - Initialization Code

```

2ED9 ED43DF2D 24470 LD (TOPAKD),BC ;Stuff for later
                24480 ;
                24490 ; P/u starting drive #, & init page counter
                24500 ;
2EDD 0E00 24510 DIR3A LD C,$-$ ;P/u starting drive
2EDF 3E16 24520 LD A,22 ;Max lines to dsply
2EE1 32662A 24530 LD (CKPAGE+1),A ;Stuff in counter
2EE4 C33324 24540 JP DIR4 ;Directory Start
                24550 ;
                24560 ; CKDSPEC - Check if a drive spec field is legal
                24570 ; HL => Drive specification Field
                24580 ; Z - Set if Drive spec Field is Legal
                24590 ; B <= Starting Drive # (0-7)
                24600 ; C <= Terminating Drive # (0-7)
                24610 ;
2EE7 7E 24620 CKDSPEC LD A,(HL) ;P/u first character
2EE8 FE2D 24630 CP '-' ;"TO" or "NOT" ?
2EEA 200C 24640 JR NZ,NOTDASH ;No - check if drive #
                24650 ;
                24660 ; Char is a "-" ---- Could be "TO" or "NOT"
                24670 ;
2EEC CD222F 24680 CALL LEGDRV ;Legal Drive Number ?
2EEF D8 24690 RET C ;No - RETurn NZ
                24700 ;
                24710 ; Legal Drive # - Next char must be a term
                24720 ;
2EF0 4F 24730 LD C,A ;C = Terminating Drive
2EF1 23 24740 INC HL ;HL => Following char
2EF2 CD0B2F 24750 CALL TERM ;Does a term follow ?
2EF5 0600 24760 LD B,0 ;B default start 0
2EF7 C9 24770 RET ;RETurn Z or NZ
                24780 ;
                24790 ; Is the First character a legal drive # ?
                24800 ;
2EF8 CD232F 24810 NOTDASH CALL LEGDRV1 ;Legal drive (0-7) ?
2EFB D8 24820 RET C ;No - RETurn NZ (ex 8)
2EFC 47 24830 LD B,A ;Set B = Starting Drive
2EFD 4F 24840 LD C,A ;Set C = Terminator
                24850 ;
                24860 ; Legal Drive - a "-" or term MUST follow
                24870 ;
2EFE 23 24880 INC HL ;Bump to next char
2EFF 7E 24890 LD A,(HL) ;If next char is not a
2F00 FE2D 24900 CP '-' ; "-", RETurn Z or NZ
2F02 2811 24910 JR Z,CKTDRIV ; depending on next char.
2F04 CD0B2F 24920 CALL TERM ;Legal terminator ?
2F07 C2282E 24930 JP NZ,ILLDRV ;No - Illegal Drive #
2F0A C9 24940 RET ;Yes - Return
                24950 ;
                24960 ; Is the character a terminator ?
                24970 ;
2F0B 7E 24980 TERM LD A,(HL) ;P/u char
2F0C FE20 24990 CP ' ' ;Space is legal
2F0E C8 25000 RET Z ;RETurn Z if space
2F0F FE0D 25010 CP CR ;CR is legal
2F11 C8 25020 RET Z ;RETurn Z if CR
2F12 FE28 25030 CP '(' ;Paren is legal
2F14 C9 25040 RET ;RETurn w/ condition
                25050 ;

```

LBDIRC - Initialization Code

```

25060 ;      Next char must be a valid drive # or term
25070 ;
2F15 CD222F 25080 CKTDRIV CALL LEGDRV      ;Legal Drive # ?
2F18 0E07   25090 LD      C,7             ;C = Default term drive 7
2F1A 38EF   25100 JR      C,TERM          ;Not drv # - ck for term
25110 ;
25120 ;      Make sure Term Drive # > or = Start Drive #
25130 ;
2F1C 4F     25140 LD      C,A             ;Set C = Term drive #
2F1D B8     25150 CP      B               ;> or = start drive # ?
2F1E D8     25160 RET      C              ;Less - Return
25170 ;
25180 ;      Drive span range good - make sure term legal
25190 ;
2F1F 23     25200 INC      HL              ;Bump ptr
2F20 18E9   25210 JR      TERM            ;RETurn Z or NZ
25220 ;
25230 ;      LEGDRV - Is a character a legal drive #
25240 ;      HL => One before Character to check
25250 ;      HL <= Character in question
25260 ;      A <= Drive Number (0-7)
25270 ;      CF <= Set if Character is not a legal drive #
25280 ;
2F22 23     25290 LEGDRV INC      HL        ;Bump to next
2F23 7E     25300 LEGDRV1 LD      A,(HL)    ;P/u char
2F24 D630   25310 SUB      '0'            ;Convert to binary
2F26 FE08   25320 CP      7+1             ;Greater than "7" ?
2F28 3F     25330 CCF                     ;C - Illegal
2F29 C9     25340 RET                     ;RETurn with condition
25350 ;
25360 ;      PRSPC - Parse a line and stuff in buffer
25370 ;      HL => Source Buffer
25380 ;      DE => Destination of converted field
25390 ;      B = # of characters to parse
25400 ;
2F2A FE24   25410 PRSPC CP      '$'        ;Wild character?
2F2C 2814   25420 JR      Z,PS2           ;Yes - stuff in buff
2F2E FE41   25430 CP      'A'            ;Alphabetic ?
2F30 3006   25440 JR      NC,PS1          ;Maybe - convert to U/C
25450 ;
25460 ;      Is the character a numeric value (0-9) ?
25470 ;
2F32 FE3A   25480 CP      '9'+1          ;Greater than "9" ?
2F34 D0     25490 RET      NC             ;Yes - return
2F35 FE30   25500 CP      '0'            ;Less than "0" ?
2F37 D8     25510 RET      C              ;Yes - return
25520 ;
25530 ;      Convert character to Upper Case
25540 ;
2F38 FE61   25550 PS1    CP      'a'        ;Lower case alpha ?
2F3A 3806   25560 JR      C,PS2           ;No - stuff in buffer
2F3C FE7B   25570 CP      'z'+1          ;
2F3E 3002   25580 JR      NC,PS2          ;
2F40 CBAF   25590 RES      5,A            ;Convert to U/C
25600 ;
25610 ;      Put char in buffer, & bump cmd & buffer ptrs
25620 ;
2F42 12     25630 PS2    LD      (DE),A    ;Stuff in buffer
2F43 13     25640 PS3    INC      DE        ;Bump

```

LBDIRC - Initialization Code

```

2F44 7E      25650      LD      A,(HL)      ;P/u command buff char
2F45 23      25660      INC      HL          ;Bump
2F46 10E2    25670      DJNZ     PRSPC      ; B times
2F48 C9      25680      RET
                25690 ;
                25700 ;
                25710 ;      PAKDAT - Pack Date & Stuff into buffer
                25720 ;      HL => Buffer containing Date string
                25730 ;      BC <= Packed Date in lsb,msb format
2F49 7E      25740 PAKDAT LD      A,(HL)      ;P/u character
2F4A 0E2F    25750      LD      C,'/'      ;Init separator
                25760 ;
                25770 ;      Is the date a valid entry ?
                25780 ;
2F4C CD8E2F  25790      CALL     PARSDAT      ;Parse entry
2F4F C21124  25800      JP      NZ,BADFMT      ;Abort on format error
                25810 ;
                25820 ;      If year = 1980 or 84 then set FEB = 29 days
                25830 ;
2F52 EB      25840      EX      DE,HL        ;Save command ptr
2F53 3AE12D  25850      LD      A,(LILBUF$)    ;P/u year (80-87)
2F56 E603    25860      AND      3          ;Mask off bits 7-2
2F58 21672D  25870      LD      HL,MAXDAYS+1  ;Set Feb to have 29 days
2F5B 2001    25880      JR      NZ,NOTLEAP    ;No - don't inc it
2F5D 34      25890      INC      (HL)        ;Leap year - inc max days
                25900 ;
                25910 ;      Check Range of month - must be 1-12
                25920 ;
2F5E 3AE32D  25930 NOTLEAP LD      A,(LILBUF$+2) ;P/u month
2F61 3D      25940      DEC      A          ;Set month = 1-11
2F62 FE0C    25950      CP      12         ;Valid month ?
2F64 D21124  25960      JP      NC,BADFMT      ;Abort if 0 or >12
                25970 ;
                25980 ;      Valid month - point HL to max days/month
                25990 ;
2F67 2B      26000      DEC      HL          ;Point before JAN entry
2F68 85      26010      ADD      A,L        ;Add the month
2F69 6F      26020      LD      L,A        ;HL => max days for month
2F6A 3001    26030      JR      NC,NOINC     ;Bump H if C set
2F6C 24      26040      INC      H
                26050 ;
                26060 ;      Check if day entry is valid
                26070 ;
2F6D 3AE22D  26080 NOINC  LD      A,(LILBUF$+1) ;P/u day entry
2F70 3D      26090      DEC      A          ;Reduce for test (0->FF)
2F71 BE      26100      CP      (HL)        ;More than max days ?
2F72 D21124  26110      JP      NC,BADFMT      ;Go if too large (or 0)
                26120 ;
                26130 ;      Pick up month from buffer
                26140 ;
2F75 21E32D  26150      LD      HL,LILBUF$+2  ;HL => Month
2F78 7E      26160      LD      A,(HL)      ;P/u month
2F79 47      26170      LD      B,A        ;Save month
                26180 ;
                26190 ;      Transfer Day to Bit positions 3-7
                26200 ;
2F7A 2B      26210      DEC      HL          ;HL => Day
2F7B 7E      26220      LD      A,(HL)      ;P/u day
2F7C 2B      26230      DEC      HL          ;HL => Year

```

LBDIRC - Initialization Code

```

2F7D 07      26240      RLCA                      ;Shift day to 3-7
2F7E 07      26250      RLCA
2F7F 07      26260      RLCA
2F80 4F      26270      LD      C,A                ;Save in C
                26280 ;
                26290 ;      Pick up year & convert to binary (0-7)
                26300 ;
2F81 7E      26310      LD      A,(HL)            ;P/u year
2F82 D650    26320      SUB      80              ;Adjust for offset
2F84 3001    26330      JR      NC,GDATE          ;If entry < 1980,
2F86 AF      26340      XOR      A                ; then use 1980
                26350 ;
                26360 ;      Shift year into positions 7-5
                26370 ;
2F87 0F      26380      GDATE RRCA                ;Shift into bits 7-5
2F88 0F      26390      RRCA
2F89 0F      26400      RRCA
                26410 ;
                26420 ;      Merge with month & Return
                26430 ;
2F8A B0      26440      OR      B                ; & merge with month
2F8B 47      26450      LD      B,A                ;Stuff in B
2F8C EB      26460      EX      DE,HL             ;HL => Buffer
2F8D C9      26470      RET                      ;RETurn
                26480 ;
                26490 ;      PARSDAT - Parse TIME/DATE string entry
                26500 ;      HL => Buffer containing string to parse
                26510 ;      C => Delimiter ("/" = DATE, ":" = TIME)
                26520 ;      LILBUF$-LILBUF$+2 <= Data in compressed format
                26530 ;      Z - Set if successful
                26540 ;
2F8E 11E32D  26550      PARSDAT LD      DE,LILBUF$+2 ;Point to buf end
2F91 0603    26560      LD      B,3                ;Process 3 fields
                26570 ;
                26580 ;      Parse a field - Return NZ if bad
                26590 ;
2F93 D5      26600      PRS1  PUSH     DE                ;Save pointer
2F94 CDA32F  26610      CALL     PRS2                ;Get a digit pair
2F97 D1      26620      POP      DE                ;Recover pointer
2F98 C0      26630      RET      NZ                ;Ret if bad digit pair
                26640 ;
                26650 ;      Good field - Stuff in buff, dec ptr, & count
                26660 ;
2F99 12      26670      LD      (DE),A            ; else stuff the value
2F9A 1B      26680      DEC      DE                ;Backup the pointer
2F9B 05      26690      DEC      B                ;Loop countdown
2F9C C8      26700      RET      Z                ;Do for 3 fields
                26710 ;
                26720 ;      Parsed a field - is the separator valid ?
                26730 ;
2F9D 7E      26740      LD      A,(HL)            ;P/u separator
2F9E 23      26750      INC      HL                ;Bump pointer
2F9F B9      26760      CP      C                ;Correct ?
2FA0 28F1    26770      JR      Z,PRS1            ;Yes - continue
2FA2 C9      26780      RET                      ;No - RET NZ
                26790 ;
                26800 ;      PRS2 - Parse a digit pair at HL
                26810 ;
2FA3 CDB92F  26820      PRS2  CALL     PRS4                ;Get a digit

```

LBDIRC - Initialization Code

```

2FA6 300F      26830      JR      NC,PRS3      ;Illegal - clr stc & RET
                26840 ;
                26850 ;      Legal Digit - Multiply by 10
                26860 ;
2FA8 5F        26870      LD      E,A          ;Multiply by ten
2FA9 07        26880      RLCA                ;X 2
2FAA 07        26890      RLCA                ;X 4
2FAB 83        26900      ADD     A,E          ;X 5
2FAC 07        26910      RLCA                ;X 10
2FAD 5F        26920      LD      E,A          ;Stuff in E
                26930 ;
                26940 ;      Get another digit
                26950 ;
2FAE CDB92F    26960      CALL    PRS4          ;Get ones digit
2FB1 3004      26970      JR      NC,PRS3      ;Bad - return NZ
                26980 ;
                26990 ;      Legal digit - Add to tens digit & set Z flag
                27000 ;
2FB3 83        27010      ADD     A,E          ;Accumulate new digit
2FB4 5F        27020      LD      E,A          ;Save 2-digit value
2FB5 BF        27030      CP      A            ;Clear flags
2FB6 C9        27040      RET                ;Return Z
                27050 ;
                27060 ;      Force NZ & Return
                27070 ;
2FB7 B7        27080 PRS3   OR      A            ;Set NZ
2FB8 C9        27090      RET                ;RETurn
                27100 ;
                27110 ;      Pick up a digit and convert to binary
                27120 ;
2FB9 7E        27130 PRS4   LD      A,(HL)      ;P/u a digit &
2FBA 23        27140      INC     HL           ; bump ptr
2FBB D630      27150      SUB     '0'         ;Convert to binary
2FBD FE0A      27160      CP      10          ;Legal ?
2FBF C9        27170      RET                ;C - legal, NC - illegal
                27180 ;
2FC0           27190 ENDMEM EQU      $
                00310 ;
                00320 ;      Bytes Free =
                00330 ;
0040           00340 FREE$ EQU      3000H-ENDMEM
                00350 ;
                00360      IFGT    $,2FFFFH
                00370      ERR     'LIB memory region overflow
                00380      ENDIF
                00390 ;
2FC0           00400      SUBTTL  <>
2400           00410      END     ENTRY

```


\$JPD	25A3 \$JP1	26F5 @@1	0000
@@2	0000 @@3	0000 @@4	0000
@MOD2	0000 @MOD4	FFFF ABB	0010
ABORT	2417 ABORT3	242F AGAIN	2B54
ALL01	2825 ALL02	2826 ALL03	282F
ALL04	2869 ALL06	28EA ALL08	2924
ALL09	2945 ALL09A	295B ALL10	2960
ALL11	2961 ALL14	2964 ALL15	2977
ALL16	2989 ALRPRT	2741 AP	0027
APARM	2805 BACK	2B84 BADFMT	2411
BADFMT\$	2D9E BLANKS	2D3A BLKHASH	4296
BREAK	0080 BS	0008 BUF2	2F00
BYTOUT	29CB BYTOUT2	29CA CALCK	29E8
CALCK1	29EC CALCK2	29FC CATBGN	2403
CFLAG\$	0002 CHKSTR	2EAA CK2HIT	26C6
CKDIR1	2665 CKDIR3	2688 CKDSPEC	2EE7
CKHIT	25A6 CKHIT1	25A7 CKHIT2	25AB
CKHIT4	2634 CKHIT5	2645 CKHIT6	2651
CKITOUT	2E2B CKMOD	2694 CKNAM	26A2
CKNAM1	26AC CKNAM2	26B8 CKNAM2A	26C9
CKNAM2B	26E4 CKPAGE	2A65 CKPAW1	2A9C
CKPAW2	2A9F CKPAWS	2A93 CKTDRIV	2F15
CKITL	2AB1 CKTO	2EC4 CLRLP	2537
COUNT	25D5 COUNTM1	2B33 CPHLDE	2A13
CR	000D CVD0	2A4A CVD1	2A4F
CVD10	2A3E CVD100	2A38 CVD2	2A63
CVDDEC	2A2A CYCLE	2B38 CYLCNT	2C1B
D79EQ	25DE D79L	2ACF DATBUF	2C4D
DATEFLD	2CD4 DATPRM	2E97 DEN	2BFF
DENSITY	2C24 DFLAG\$	0003 DIR	2DD9
DIR0	2E35 DIR1	2E43 DIR2	2E54
DIR3	2E64 DIR3A	2EDD DIR4	2433
DIRA	2DE2 DIRPTR	2DD9 DIR_0	28B0
DIVIDE	2930 DODSP	272D DOLDIR	24AA
DONAM1	274C DONAM2	2759 DONAM3	275D
DONAM4	2769 DONAM5	2774 DONAM5A	278D
DONAM6	27DA DONAM7	27DC DONAM8	27FD
DONAM9	280D DONTDEC	2874 DONTINC	2884
DRESP	2BE7 DRIVE	2C0E DSTRING	2C07
DUBDEN	249D ENDLINE	2819 ENDMEM	2FC0
ENTRY	2400 EOFBYTE	2895 ERR32	241C
ETX	0003 EXIT	2429 EXITSRT	2BB1
EXTENTS	2CCD FDISP	2CE9 FILES	25CF
FILFLAG	2DD8 FINIS	2BAF FLAG	0040
FMPAKD	2DDD FPLP	2E05 FRCTO	2ED1
FREE\$	0040 FS1	24CB FS2	24CC
FS3	24D1 FSTRING	2CDE FTFLG	2DD7
FUSED	2CFA GAT	2E00 GDATE	2F87
GDCKDRV	2475 GETHI	2E7C GETPRM	2E11
GOODCYL	2507 GPCLOOP	24F0 GTDATE	2E91
GTUSED	250E HARD	2C03 HEADING	2C57
HIT	2E00 HUNDTAB	2D72 ILLDATE	2579
ILLDRV	2E28 IOER1	29C6 IOERR	241E
IOERR5	2A7F IPARM	268C KEEPEOF	289E
KFLAG	2A93 KFLAG\$	000A KFLAG1	2AAD
KFREE	2C30 KPOSS	2C3C KSIZE	2CC0
LEGDRV	2F22 LEGDRV1	2F23 LF	000A
LILBUF\$	2DE1 LINOUT	29B9 LKLOOP	2501

LRL	2CAE	MATCH	2733	MAXDAYS	2D66
MAXMEM	2DDB	MEMORY	3000	MFLG	26C3
MONTBL	2DAE	NAME	2C11	NDRIVE	2D22
NEXTDRV	2472	NODISK	2D1B	NOINC	2F6D
NOMEM	240D	NOMEM\$	2D7A	NOPRT	29DE
NOTAP	2619	NOTDASH	2EF8	NOTDUB	24E9
NOTITLE	2A87	NOTLEAP	2F5E	NO_DISK	245F
NPARM	2A71	NUM	0080	OFFSET	2CBB
OUTCHR	2940	OUTMOD	2937	OUTSPC	29E0
PAKDAT	2F49	PARSDAT	2F8E	PAR_ERR	002C
PLEVEL	2CA7	POP	2B90	PPARM	29D2
PRMTBL\$	2BC4	PROTS\$	2D46	PRS1	2F93
PRS2	2FA3	PRS3	2FB7	PRS4	2FB9
PRSPC	2F2A	PS1	2F38	PS2	2F42
PS3	2F43	RECORDS	2CB3	REPEAT	2B5A
RESKFL	2AA8	RESTPTR	2E17	SAVESP	242C
SBUFFER	2985	SFLAG	2A77	SFLAG\$	0012
SHELL	2B32	SKIPLOC	2517	SORT1	2AEF
SORT2	2B0F	SORT2A	2B1F	SORT3	2B23
SORTIT	2ADF	SORTPRM	26F8	SPARM	267E
SPECIF	2635	SPUSED	2D10	STARS	2DD2
STORI	2B5B	STORJ	2B55	STORK	2BBA
STORM	2B39	STR	0020	SWAP	2B94
TAB	0009	TDATE	2D31	TERM	2F0B
TERMDRV	262A	TFILES	25F6	TOPAKD	2DDF
TOTGRNS	2601	TSTSIZ	2886	UNPACK	2A19
VFLAG\$	0015	@@ABORT	6E3D	@@ADTSK	6ED0
@@BANK	73E8	@@BKSP	70C8	@@BREAK	73FE
@@CHNIO	6E28	@@CKBRKC	744C	@@CKDRV	6F24
@@CKEOF	70DD	@@CKTSK	6EBB	@@CLOSE	70B3
@@CLS	7436	@@CMNDI	6E67	@@CMNDR	6E7C
@@CTL	6C8C	@@DATE	6DFE	@@DCSTAT	6F63
@@DEBUG	6EA6	@@DECHEX	7368	@@DIRRD	72D5
@@DIRWR	72EA	@@DIV16	7353	@@DIV8	733E
@@DODIR	6F39	@@DSP	6C50	@@DSPLY	6CF0
@@ERROR	6E91	@@EXIT	6E52	@@FEXT	7242
@@FLAGS	73D2	@@FNAME	7257	@@FSPEC	722D
@@GATRD	72C0	@@GATWR	72FF	@@GET	6C64
@@GTDCB	7281	@@GTDCI	726C	@@GTMOD	7296
@@HDFMT	700B	@@HEX16	73A7	@@HEX8	7392
@@HEXDEC	737D	@@HIGH\$	73BC	@@INIT	7089
@@KBD	6CC8	@@KEY	6C3C	@@KEYIN	6CDC
@@KLTSK	6F0F	@@LOAD	7203	@@LOC	70F2
@@LOF	7107	@@LOGGER	6D27	@@LOGOT	6D3C
@@MSG	6D73	@@MUL16	7329	@@MUL8	7314
@@OPEN	709E	@@PARAM	6DE9	@@PAUSE	6DD4
@@PEOF	711C	@@POSN	7131	@@PRINT	6D88
@@PRT	6CA0	@@PUT	6C78	@@RAMDIR	6F4E
@@RDSEC	6FE1	@@RDSSC	72AB	@@READ	7146
@@REMOV	7074	@@RENAM	705F	@@REW	715B
@@RMTSK	6EE5	@@RPTSK	6EFA	@@RREAD	7170
@@RSLCT	6FCC	@@RSTOR	6F8D	@@RUN	7218
@@RWRIT	7185	@@SEEK	6FB7	@@SEEKSC	719A
@@SKIP	71AF	@@SLCT	6F78	@@STEPI	6FA2
@@TIME	6E13	@@VDCTL	6DBF	@@VER	71C4
@@VRSEC	6FF6	@@WEOF	71D9	@@WHERE	6CB4
@@WRITE	71EE	@@WRSEC	7020	@@WRSSC	7035
@@WRTRK	704A				

2400 is the transfer address

00000 Total errors

NOTES:

NOTES:

Command: D0

Library: SYS6/SYS

ISAM # : 91H

```

00100 ;LBDO/ASM - Library 'DO' command
00110 TITLE <DO - LS-DOS 6.2>
00120 ;
00C0 00130 JFCB$ EQU 0C0H ;Low core EQU*
00140 ;
00150 ;
0000 00160 SMALL EQU 0
000D 00170 CR EQU 13
0000 00180 *GET SVCMAC:3 ;SVC Macro equivalents
00010 ;SVC MAC/ASM - LS-DOS Version VI
00020 *LIST OFF
03900 *LIST ON
00190 ;
2400 00200 ORG 2400H
00210 ;
2400 00220 DO EQU $
00230 ;
00240 ; Note: The first 80 bytes (until PARSINP) are
00250 ; Used as a line buffer during processing.
00260 ;
2400 00270 JCLBUF2 EQU $
2400 ED737929 00280 LD (SPSAV+1),SP ;Save stack pointer
00290 ;
00300 IF SMALL
00310 JR NOCPLS ;No compile if Small
00320 ENDIF
2404 226D24 00330 LD (INBUF+1),HL ;Save start of command
00340 ;
00350 *LIST OFF
00370 *LIST ON
2407 00380 @@FLAGS ;Get flag table pointer
2407 3E65 00001 LD A,101
2409 EF 00002 RST 40
240A 7E 00390 LD A,(HL)
240B FE2A 00400 CP '*' ;Execute last DO file?
240D CAF024 00410 JP Z,NOCPL2
2410 FE3D 00420 CP '=' ;Execute without compile?
2412 CAD924 00430 JP Z,NOCPL
2415 FE24 00440 CP '$' ;Compile only?
2417 200A 00450 JR NZ,GETSPEC
2419 32B424 00460 LD (NOEXEC?+1),A
241C 23 00470 INC HL
241D 7E 00480 LD A,(HL)
241E FE20 00490 CP ' ' ;Bypass space separator
2420 2001 00500 JR NZ,GETSPEC ; if present
2422 23 00510 INC HL
2423 117C29 00520 GETSPEC LD DE,DOFCB ;Get DO filespec
2426 00530 @@FSPEC
2426 3E4E 00003 LD A,78
2428 EF 00004 RST 40
2429 C24829 00540 JP NZ,SPCREQ ;Go if bad/missing filespec
242C E5 00550 PUSH HL ;Save INBUF$ pointer
242D 21D729 00560 LD HL,SYSJCL+7 ;Default ext to "/JCL"
2430 00570 @@FEXT
2430 3E4F 00005 LD A,79
2432 EF 00006 RST 40
2433 21002C 00580 LD HL,INPBUF ;Open DO file
2436 45 00590 LD B,L ;LRL=256
2437 FDCB12C6 00600 SET 0,(IY+'S'-'A') ;Inhibit file open bit
243B 00610 @@OPEN
243B 3E3B 00007 LD A,59

```

```

243D EF      00008      RST      40
243E C23D29  00620      JP       NZ,IOERR      ;Jump on open error
2441 CDF524  00630      CALL      MOVFCB      ;Move SYSTEM/JCL into FCB
2444 11C000  00640      LD        DE,JFCB$      ;Init FCB pointer
2447 21002D  00650      LD        HL,OUTBUF
244A         00660      @@INIT
244A 3E3A     00009      LD        A,58
244C EF      00010      RST      40
244D C25C29  00670      JP       NZ,DSKFUL      ;Jump on error
2450 E1       00680      POP       HL          ;Rcvr pointer to INBUF$
          00690 ;
          00700 ;
          00710 ;      Routine to parse a command line
          00720 ;
2451 7E       00720 PARSINP LD      A,(HL)      ;P/u line char
2452 FE0D     00730      CP       CR          ;End of line?
2454 CA4825  00740      JP       Z,TSTLBL
2457 23       00750      INC       HL          ;Bump pointer
2458 CD2D29  00760      CALL      CKSPCOM      ;Ignore spaces & commas
245B 28F4     00770      JR       Z,PARSINP
245D FE28     00780      CP       '('        ;Beginning of parms?
245F CA0125  00790      JP       Z,PARAM
2462 FE3B     00800      CP       ';'        ;Line continuation?
2464 C26029  00810      JP       NZ,PRMERR
2467 0E3F     00820      LD        C,'?'      ;Prompt for line continue
2469         00830      @@DSP
2469 3E02     00011      LD        A,2
246B EF      00012      RST      40
246C 210000  00840 INBUF  LD      HL,$-$      ;Input continuation line
246F 2D       00850      DEC       L          ;Backup to start
2470 2D       00860      DEC       L
2471 01004F  00870      LD        BC,79<8      ;Max 79 chars input
2474         00880      @@KEYIN
2474 3E09     00013      LD        A,9
2476 EF      00014      RST      40
2477 DA6029  00890      JP       C,PRMERR      ;Jump if break
247A         00900      @@LOGGER      ;Log the line
247A 3E0B     00015      LD        A,11
247C EF      00016      RST      40
247D 18D2     00910      JR       PARSINP      ;Go parse it
          00920 ;
          00930 ;
          00940 ;      Routine to move to higher level nest
          00950 ;
247F 2ABA2A  00950 UNNEST LD      HL,(NESTPTR)    ;Shift the last nest's
2482 2B       00960      DEC       HL          ; FCB into FCB area
2483 119B29  00970      LD        DE,DOFCB+31
2486 012000  00980      LD        BC,32
2489 EDB8     00990      LDDR
248B 23       01000      INC       HL
248C 22BA2A  01010      LD        (NESTPTR),HL      ;Reset current FCB ptr
248F 117C29  01020      LD        DE,DOFCB      ;Reread last sector of
2492         01030      @@RREAD      ; nested FCB
2492 3E45     00017      LD        A,69
2494 EF      00018      RST      40
2495 C23D29  01040      JP       NZ,IOERR
2498 C9       01050      RET
          01060 ;
          01070 ;
2499 2ABA2A  01070 CKNEST LD      HL,(NESTPTR)    ;P/u current FCB pointer
249C 11BC2A  01080      LD        DE,NESTFCB      ;Is it the first nest?
249F AF       01090      XOR       A
24A0 ED52     01100      SBC       HL,DE
24A2 2806     01110      JR       Z,CPLFIN      ;Jump if so & exit

```



```

24A4 CD7F24 01120 CALL UNNEST ; processing
24A7 C37E25 01130 JP CPLJCL
01140 ;
01150 ; Finished compilation - Close 'er up
01160 ;
24AA 11C000 01170 CPLFIN LD DE,JFCB$ ;Close SYSTEM/JCL file
24AD 01180 @@CLOSE
24AD 3E3C 00019 LD A,60
24AF EF 00020 RST 40
24B0 C23D29 01190 JP NZ,IOERR
24B3 3E00 01200 NOEXEC? LD A,0 ;Set to non-zero on
24B5 B7 01210 OR A ; compile only
24B6 210000 01220 LD HL,0
24B9 C0 01230 RET NZ ;Exit on compile only
01240 ENDIF
01250 *LIST ON
01260 ;
24BA 11C000 01270 CPLFIN1 LD DE,JFCB$ ;Point to SYSTEM/JCL FCB
24BD 210000 01280 LD HL,0 ;Correct bufptr later
24C0 45 01290 LD B,L ;LRL=256
24C1 FDCB12C6 01300 SET 0,(IY+'S'-'A') ;Inhibit file open bit
24C5 01310 @@OPEN ;Open it up
24C5 3E3B 00021 LD A,59
24C7 EF 00022 RST 40
24C8 C23D29 01320 JP NZ,IOERR ;Jump on error
24CB ED4BC600 01330 LD BC,(JFCB$+6) ;Get SBUFF$
24CF 01340 @@DIRRD
24CF 3E57 00023 LD A,87
24D1 EF 00024 RST 40
24D2 7C 01350 LD A,H ;Stuff high order to
24D3 32C400 01360 LD (JFCB$+4),A ; use for JFCB$ buffer
24D6 3E9D 01370 LD A,9DH ;Call SYS11, entry 1
24D8 EF 01380 RST 28H
01390 ;
01400 ; Process execution without compilation
01410 ;
24D9 23 01420 NOCPL INC HL
24DA 7E 01430 NOCPLS LD A,(HL) ;Bypass space separator
24DB FE20 01440 CP ' ' ; if present
24DD 28FA 01450 JR Z,NOCPL
24DF 11C000 01460 NOCPL1 LD DE,JFCB$ ;Fetch DO filespec
24E2 01470 @@FSPEC
24E2 3E4E 00025 LD A,78
24E4 EF 00026 RST 40
24E5 C24829 01480 JP NZ,SPCREQ ;Jump on error
24E8 21D729 01490 LD HL,SYSJCL+7 ;Default to /JCL
24EB 01500 @@FEXT
24EB 3E4F 00027 LD A,79
24ED EF 00028 RST 40
24EE 18CA 01510 JR CPLFIN1 ;Go execute file
01520 ;
01530 *LIST OFF
01550 *LIST ON
24F0 CDF524 01560 NOCPL2 CALL MOVFCB ;Execute SYSTEM/JCL
24F3 18C5 01570 JR CPLFIN1 ; file
01580 ;
24F5 21D029 01590 MOVFCB LD HL,SYSJCL ;Move SYSTEM/JCL into
24F8 11C000 01600 LD DE,JFCB$ ; FCB area
24FB 012000 01610 LD BC,32
24FE EDB0 01620 LDIR
2500 C9 01630 RET

```

```

01640 ;
01650 ; Found a parm entered
01660 ;
2501 CDE727 01670 PARAM CALL PARSNAM ;Parse symbol -> current
2504 2014 01680 JR NZ,PARAM1 ;Jump if bad symbol
2506 F5 01690 PUSH AF ;Save separator char
2507 3E00 01700 FNDLBL LD A,0 ;Test if a label
2509 B7 01710 OR A ; was found
250A 2029 01720 JR NZ,MOVLBL
250C CDB128 01730 CALL FINDSYM ;Search symbol table
250F CA6429 01740 JP Z,MULDEF ;Multiply defined if in
2512 CD9428 01750 CALL MOVNAME ;Add symbol to table
2515 F1 01760 POP AF ;Recover separator
2516 FE3D 01770 CP '=' ;Assignment?
2518 2811 01780 JR Z,PARAM2
251A CD2D29 01790 PARAM1 CALL CKSPCOM ;Ck space or comma
251D 28E2 01800 JR Z,PARAM
251F FE29 01810 CP ')' ;Exit parm scan on
2521 CA5124 01820 JP Z,PARSINP ; closing paren
2524 FE0D 01830 CP CR ;Also accept closing CR
2526 2820 01840 JR Z,TSTLBL
2528 C36029 01850 JP PRMERR ;Else parm error
01860 ;
252B CDF227 01870 PARAM2 CALL PARSVAL ;Parse value into buf
252E F5 01880 PUSH AF ;Save separator char
252F CDA628 01890 CALL MOVALUE ;Symbol value into table
2532 F1 01900 GETSEP POP AF ;Recover separator
2533 18E5 01910 JR PARAM1 ;Loop
01920 ;
2535 E5 01930 MOVLBL PUSH HL
2536 219C29 01940 LD HL,CURSYM ;Pt to current sym buf
2539 11C529 01950 LD DE,LBLSAV ; & save label for
253C 010800 01960 LD BC,8 ; later testing
253F EDB0 01970 LDIR
2541 AF 01980 XOR A ;Turn off "found label"
2542 320825 01990 LD (FNDLBL+1),A
2545 E1 02000 POP HL ;Rcvr line ptr
2546 18EA 02010 JR GETSEP ;Back for more
02020 ;
02030 ; Got to end of JCL command line
02040 ;
2548 3A5528 02050 TSTLBL LD A,(GOTLBL+1) ;Was @LABEL a parm?
254B B7 02060 OR A
254C 2830 02070 JR Z,CPLJCL ;If not, don't look
02080 ;
02090 ; Find the procedure block named @LABEL
02100 ;
254E CD6126 02110 FNDLBL CALL RDJCL ;Read JCL line
2551 2811 02120 JR Z,GOTLIN ;Go if line read
2553 2ABA2A 02130 LD HL,(NESTPTR) ;See if nested
2556 11BC2A 02140 LD DE,NESTFCB ; in an Include file
2559 AF 02150 XOR A
255A ED52 02160 SBC HL,DE
255C CA5429 02170 JP Z,NOFIND ;If not, lable not found
255F CD7F24 02180 CALL UNNEST ; else continue search
2562 18EA 02190 JR FNDLBL
02200 ;
2564 218C2B 02210 GOTLIN LD HL,JCLBUF1 ;Pt to start
2567 7E 02220 LD A,(HL) ;Is 1st char a label
2568 FE40 02230 CP '@' ; indicator?
256A 20E2 02240 JR NZ,FNDLBL ;Back for more if not

```

```

02250 ;
02260 ;      Found a label - is it the one needed?
02270 ;
256C 23      02280      INC      HL      ;Pt to 1st char
256D EB      02290      EX       DE,HL   ;Ptr to DE
256E 21C529  02300      LD       HL,LBLSAV
2571 010808  02310      LD       BC,808H   ;Symbol & field len =8
2574 CDDD28  02320      CALL     FNDPRM    ;A match?
2577 20D5    02330      JR       NZ,FINDLBL ;No match? look for next
2579 1803    02340      JR       CPLJCL    ; else you're the one
02350 ;
257B CD9226  02360 CONDCPL CALL     TSTCOND
257E CD6126  02370 CPLJCL  CALL     RDJCL   ;Read line from JCL file
2581 C29924  02380      JP       NZ,CKNEST ;Exit on end of file
2584 218C2B  02390      LD       HL,JCLBUF1 ;Parse the line just read
2587 110024  02400      LD       DE,JCLBUF2
258A 7E      02410      LD       A,(HL)
258B 23      02420      INC      HL
258C FE40    02430      CP       '@'      ;End procedure if found
258E CA9924  02440      JP       Z,CKNEST  ; another label
2591 FE2F    02450      CP       '/'      ;Slash?
2593 2004    02460      JR       NZ,CPLJCL1
2595 BE      02470      CP       (HL)     ;Double slash?
2596 CA3326  02480      JP       Z,MACRO   ;Jump on double slash
02490 CPLJCL1
02500 ;
02510 ;      Modification for HEX parsing
02520 ;
2599 FE23    02530      CP       '#'      ;Substitution?
259B 2825    02540      JR       Z,CPLJCL4
259D FE25    02550      CP       '%'      ;Hex value?
259F 2017    02560      JR       NZ,CPLJCL2 ;Back to take char if not
25A1 CDA625  02570      CALL     CPLJCL7    ;Go test double %
25A4 1818    02580      JR       CPLJCL3
25A6 BE      02590 CPLJCL7 CP       (HL)   ;Double %?
25A7 2821    02600      JR       Z,CPLJCL6
25A9 CDCE25  02610      CALL     CVRTHX     ;Convert digit
25AC 23      02620      INC      HL       ;Bump to next char
25AD 07      02630      RLCA
25AE 07      02640      RLCA
25AF 07      02650      RLCA
25B0 07      02660      RLCA            ;Rotate into left nybble
25B1 4F      02670      LD       C,A      ;Save for now
25B2 CDCE25  02680      CALL     CVRTHX     ;Convert 2nd digit
25B5 B1      02690      OR       C        ;Merge left nybble
25B6 1812    02700      JR       CPLJCL6
25B8 12      02710 CPLJCL2 LD       (DE),A ;Nothing special, xfer
25B9 13      02720      INC      DE
25BA FE0D    02730      CP       CR
25BC 28BD    02740      JR       Z,CONDCPL ;Exit on end of line
25BE 7E      02750 CPLJCL3 LD       A,(HL) ;Grab next input char
25BF 23      02760      INC      HL
25C0 18D7    02770      JR       CPLJCL1    ; & loop
25C2 CDC725  02780 CPLJCL4 CALL     CPLJCL5 ;Ck on double '#'
25C5 18F7    02790      JR       CPLJCL3    ;Substitute if not ##
25C7 BE      02800 CPLJCL5 CP       (HL)   ;Double #?
25C8 2015    02810      JR       NZ,SUBSYM  ;Jump to substitute if
25CA 23      02820 CPLJCL6 INC      HL     ; only single #
25CB 12      02830      LD       (DE),A   ; else xfer the char
25CC 13      02840      INC      DE
25CD C9      02850      RET

```

```

02860 ;
25CE 7E 02870 CVRTHX LD A,(HL) ;P/u the digit
25CF D630 02880 SUB 30H ;Start conversion
25D1 380A 02890 JR C,CVRTHE1 ;Error if < 0
25D3 FE0A 02900 CP 10
25D5 D8 02910 RET C ;Go if 0-9
25D6 CBAF 02920 RES 5,A ;In case 1/c
25D8 D607 02930 SUB 7 ;Adjust A-F -> 10-15
25DA FE10 02940 CP 16
25DC D8 02950 RET C ;Go if 10-15
25DD 183F 02960 CVRTHE1 JR BADHDR
02970 ;
02980 ; Symbol substitution routine
02990 ;
25DF E5 03000 SUBSYM PUSH HL
25E0 D5 03010 PUSH DE
25E1 CDE727 03020 CALL PARSNAM ;Parse symbol
25E4 FE23 03030 CP '#' ;Must have closing #
25E6 2036 03040 JR NZ,BADHDR ;Bad JCL format if not
25E8 E3 03050 EX (SP),HL ;Put new posn on stack
25E9 E5 03060 PUSH HL ; and get HL=start posn
25EA CDB128 03070 CALL FINDSYM ;Get symbol value
25ED 200F 03080 JR NZ,SUBSYM1 ;Bypass if not in table
25EF 1A 03090 LD A,(DE) ;Get symbol length
25F0 B7 03100 OR A
25F1 280B 03110 JR Z,SUBSYM1 ;Bypass if zero length
25F3 0600 03120 LD B,0
25F5 4F 03130 LD C,A
25F6 13 03140 INC DE ;Point to 1st symbol char
25F7 E1 03150 POP HL ;Rcvr where we need to
25F8 EB 03160 EX DE,HL ; substitute then move
25F9 EDB0 03170 LDIR ; symbol value into pos
25FB E1 03180 POP HL
25FC F1 03190 POP AF
25FD C9 03200 RET
03210 ;
25FE D1 03220 SUBSYM1 POP DE ;Symbol not in table, so
25FF F1 03230 POP AF ; leave as is in the DO
2600 E1 03240 POP HL ; file.
2601 3E23 03250 LD A,'#' ;Starting #
2603 12 03260 SUBSYM2 LD (DE),A
2604 13 03270 INC DE ;Inc buffer
2605 7E 03280 LD A,(HL) ;Get a char from line
2606 23 03290 INC HL
2607 FE0D 03300 CP CR ;If a CR before closing #
2609 2813 03310 JR Z,BADHDR ; abort
260B FE23 03320 CP '#' ;End of substitution?
260D 20F4 03330 JR NZ,SUBSYM2 ;Get more if not
260F 12 03340 LD (DE),A
2610 13 03350 INC DE
2611 C9 03360 RET
03370 ;
03380 ; Check if conditional is at top level
03390 ;
2612 ED5B5C2B 03400 CKCOND LD DE,(CONDPTR) ;P/u conditional pointer
2616 215E2B 03410 LD HL,CONDFLG ;Test if still on 1st one
2619 AF 03420 XOR A
261A ED52 03430 SBC HL,DE
261C EB 03440 EX DE,HL ;Pointer back to HL
261D C0 03450 RET NZ ;Ok if nested else error
03460 ;

```

```

03470 ;      Output invalid JCL format message
03480 ;
261E 11832B 03490 BADHDR LD      DE,BADHDR$+5    ;Show bad JCL line found
2621 2ACE29 03500      LD      HL,(LINENO)      ;Put decimal line #
2624      03510      @@HEXDEC                  ; into message
2624 3E61    00029      LD      A,97
2626 EF      00030      RST      40
2627 217E2B 03520      LD      HL,BADHDR$        ;Display bad line #
262A      03530      @@LOGOT
      00031      IFEQ     00H,1
      00032      LD      HL,
      00033      ENDIF
262A 3E0C    00034      LD      A,12
262C EF      00035      RST      40
262D 217A2A 03540 BADH1 LD      HL,BADJCL$      ; and abort message
2630 C36729 03550      JP      EXTERR
      03560 ;
      03570 ;      Compile "/" line
      03580 ;
      03590 ;
2633 23      03590 MACRO INC      HL
2634 CDE727 03600      CALL     PARSNAM          ;Get symbol name
2637 2015    03610      JR      NZ,MACRO2        ;Go if not JCL macro
2639 CDC228 03620      CALL     CK4COND          ;Ck for IF, ELSE, END
263C D5      03630      PUSH     DE              ;Stack the routine entry
263D C8      03640      RET      Z              ; & branch if found
263E D1      03650      POP      DE              ; else remove RET &...
      03660 ;
      03670 ;      Test the conditional logic state
      03680 ;
      03690 ;
263F ED5B5C2B 03690      LD      DE,(CONDPTR)    ;P/u conditional pointer
2643 1A      03700      LD      A,(DE)          ; & conditional state
2644 B7      03710      OR      A
2645 C27E25 03720      JP      NZ,CPLJCL        ;Jump if logic FALSE
2648 CDCB28 03730      CALL     CK4ASSN          ;Test for SET, RESET,
      03740      ; ASSIGN, INCLUDE, QUIT
264B D5      03750      PUSH     DE              ;Stack the routine entry
264C C8      03760      RET      Z              ; & branch if found
264D D1      03770      POP      DE
264E 118C2B 03780 MACRO2 LD      DE,JCLBUF1      ;Point to where we left
2651 AF      03790      XOR      A              ; off and continue to
2652 ED52    03800      SBC      HL,DE          ; parse the input line
2654 44      03810      LD      B,H            ; from the JCL file
2655 4D      03820      LD      C,L
2656 218C2B 03830      LD      HL,JCLBUF1
2659 110024 03840      LD      DE,JCLBUF2
265C EDB0    03850      LDIR
265E C3BE25 03860      JP      CPLJCL3
      03870 ;
      03880 ;      Read a line from the JCL file
      03890 ;
      03900 ;
2661 2ACE29 03900 RDJCL LD      HL,(LINENO)      ;Bump line counter
2664 23      03910      INC      HL
2665 22CE29 03920      LD      (LINENO),HL
2668 218C2B 03930      LD      HL,JCLBUF1        ;Point to line buffer
266B 117C29 03940      LD      DE,DOFCB          ;Point to FCB
266E 0650    03950      LD      B,80            ;Permit only 80 chars
2670      03960 RDJCL1 @@GET                    ;Get a char
2670 3E03    00036      LD      A,3
2672 EF      00037      RST      40
2673 2014    03970      JR      NZ,RDJCL2        ;Jump on error
2675 B7      03980      OR      A

```

The Source	LIBRARY Files	DO - LS-DOS 6.2	Page 00008
2676 2816	03990	JR Z,RDJCL3	;Bypass on null byte
2678 77	04000	LD (HL),A	;Xfer byte to line buf
2679 23	04010	INC HL	
267A FE0D	04020	CP CR	;End of line?
267C C8	04030	RET Z	
267D 10F1	04040	DJNZ RDJCL1	;Loop if not
	04050 ;		
	04060 ;		
	04070 ;		
			If falls through, line too long
267F 360D	04080	LD (HL),CR	;Stuff CR & provide
2681 21EE29	04090	LD HL,LINLNG\$; error log message
2684 222E26	04100	LD (BADH1+1),HL	
2687 1895	04110	JR BADHDR	
	04120 ;		
2689 FE1C	04130	RDJCL2 CP 1CH	;EOF?
268B C23D29	04140	JP NZ,IOERR	;Jump on any other error
268E 3E1C	04150	RDJCL3 LD A,1CH	
2690 B7	04160	OR A	
2691 C9	04170	RET	
	04180 ;		
	04190 ;		
	04200 ;		Act on JCL line if conditional state = TRUE
2692 2A5C2B	04210	TSTCOND LD HL,(CONDPTR)	;Grab conditional pointer
2695 7E	04220	LD A,(HL)	;Grab conditional state
2696 B7	04230	OR A	
2697 C0	04240	RET NZ	;Return if logic FALSE
2698 210024	04250	LD HL,JCLBUF2	;Point to processed line
269B 11C000	04260	LD DE,JFCB\$;SYSTEM/JCL FCB
269E 7E	04270	LD A,(HL)	;Ck on double /
269F FE2F	04280	CP '/'	
26A1 2010	04290	JR NZ,WRCPLD	;Done if not /
26A3 23	04300	INC HL	
26A4 BE	04310	CP (HL)	;Check for double /
26A5 2B	04320	DEC HL	
26A6 200B	04330	JR NZ,WRCPLD	;Jump if not //
26A8 3A0224	04340	LD A,(JCLBUF2+2)	;Ck on comment
26AB FE2E	04350	CP '.'	;//. ?
26AD 2004	04360	JR NZ,WRCPLD	;Bypass if not comment
26AF	04370	@@DSPLY	;Else display the comment
	00038	IFEQ 00H,1	
	00039	LD HL,	
	00040	ENDIF	
26AF 3E0A	00041	LD A,10	
26B1 EF	00042	RST 40	
26B2 C9	04380	RET	
	04390 ;		
	04400 ;		
	04410 ;		Write compiled line to SYSTEM/JCL
26B3 4E	04420	WRCPLD LD C,(HL)	;P/u a char
26B4	04430	@@PUT	;Put it out
26B4 3E04	00043	LD A,4	
26B6 EF	00044	RST 40	
26B7 C23D29	04440	JP NZ,IOERR	;Jump on error
26BA 7E	04450	LD A,(HL)	;Grab again to test
26BB 23	04460	INC HL	;Bump pointer
26BC FE0D	04470	CP CR	;End of line?
26BE 20F3	04480	JR NZ,WRCPLD	;Loop if not
26C0 C9	04490	RET	
	04500 ;		
	04510 ;		Parameter tables
	04520 ;		

```

26C1 49      04530 CONDTBL DB      'IF      '
      46 20 20 20
26C6 0A27    04540      DW      IF01
26C8 45      04550      DB      'ELSE      '
      4C 53 45 20
26CD 3327    04560      DW      ELSE1
26CF 45      04570      DB      'END      '
      4E 44 20 20
26D4 3E27    04580      DW      END1
26D6 00      04590      NOP
26D7 53      04600 ASSNTBL DB      'SET      '
      45 54 20 20 20 20
26DF 6D27    04610      DW      SET1
26E1 52      04620      DB      'RESET      '
      45 53 45 54 20 20 20
26E9 7C27    04630      DW      RESET1
26EB 41      04640      DB      'ASSIGN      '
      53 53 49 47 4E 20 20
26F3 8E27    04650      DW      ASSIGN
26F5 49      04660      DB      'INCLUDE      '
      4E 43 4C 55 44 45 20
26FD A927    04670      DW      INCLUD
26FF 51      04680      DB      'QUIT      '
      55 49 54 20 20 20 20
2707 E127    04690      DW      QUIT
2709 00      04700      NOP
      04710 ;
      04720 ;      Process IF command
      04730 ;
270A CD4727  04740 IF01      CALL      IF05      ;Parse expression
270D 2814    04750      JR      Z,IF02      ;Z=true, NZ=false
270F FE0D    04760      CP      CR          ;False & end of line?
2711 2813    04770      JR      Z,IF03
2713 FE2B    04780      CP      '+'        ;Logical OR?
2715 28F3    04790      JR      Z,IF01
      04800 ;
      04810 ;      Test for FALSE and logical AND (&)
      04820 ;
2717 FE26    04830      CP      '&'        ;Separator AND?
2719 2055    04840      JR      NZ,BADHDR0 ;Invalid format if not
271B 23      04850 IF01A    INC      HL          ;Ignore rest of line
271C 7E      04860      LD      A,(HL)
271D FE0D    04870      CP      CR
271F 20FA    04880      JR      NZ,IF01A
2721 1803    04890      JR      IF03
2723 AF      04900 IF02    XOR      A          ;Logic = true
2724 1802    04910      JR      IF04
2726 3EFF    04920 IF03    LD      A,0FFH      ;Logic = false
2728 2A5C2B  04930 IF04    LD      HL,(CONDPTR) ;Get conditional pointer
272B B6      04940      OR      (HL)        ;Set logic state
272C 23      04950      INC      HL          ;Bump pointer
272D 77      04960      LD      (HL),A      ;Stuff state result
272E 225C2B  04970      LD      (CONDPTR),HL ;Save pointer
2731 1846    04980      JR      GOJCL
      04990 ;
      05000 ;      Process ELSE command
      05010 ;
2733 CD1226  05020 ELSE1    CALL      CKCOND      ;Ck nest of conditional
2736 7E      05030      LD      A,(HL)        ;Flip state of flag based
2737 2F      05040      CPL          ; on previous test
2738 2B      05050      DEC      HL

```

The Source	LIBRARY Files	DO - LS-DOS 6.2	Page 00010
2739 B6	05060	OR (HL)	;OR in previous state
273A 23	05070	INC HL	
273B 77	05080	LD (HL),A	;Store new value
273C 183B	05090	JR GOJCL	
	05100 ;		
	05110 ;	Process END command	
	05120 ;		
273E CD1226	05130 END1	CALL CKCOND	;Ck nest level
2741 2B	05140	DEC HL	;Backup conditional one
2742 225C2B	05150	LD (CONDPTR),HL	; level & reset pointer
2745 1832	05160	JR GOJCL	
	05170 ;		
	05180 ;	Parse conditional expression logic	
	05190 ;		
2747 CD5127	05200 IF 05	CALL IF06	;Get if symbol is true
274A C0	05210	RET NZ	; or false & ret if false
274B FE26	05220	CP '&'	;Logical AND separator?
274D 28F8	05230	JR Z,IF05	;If TRUE AND -> ck next
274F AF	05240	XOR A	;True and not AND,
2750 C9	05250	RET	; ret true
2751 7E	05260 IF 06	LD A,(HL)	
2752 FE2D	05270	CP '-'	;Logical NOT?
2754 200A	05280	JR NZ,IF08	
2756 23	05290	INC HL	;Bypass '-'
2757 CD6027	05300	CALL IF08	;Grab symbol logic state
275A 2001	05310	JR NZ,IF07	;Z=true, NZ=false
275C F6	05320	DB 0F6H	;Was true, not => false
275D AF	05330 IF 07	XOR A	;Was false, not => true
275E 78	05340	LD A,B	;Rcvr separator
275F C9	05350	RET	
2760 CDE727	05360 IF 08	CALL PARSNAM	;Get symbol name into buf
2763 C0	05370	RET NZ	;Ret if bad symbol
2764 F5	05380	PUSH AF	
2765 E5	05390	PUSH HL	
2766 CDB128	05400	CALL FINDSYM	;Find symbol in table
2769 E1	05410	POP HL	
276A C1	05420	POP BC	
276B 78	05430	LD A,B	;Put zero in A & use flag
276C C9	05440	RET	;From search
	05450 ;		
	05460 ;	Process SET command	
	05470 ;		
276D CDE727	05480 SET1	CALL PARSNAM	;Parse symbol name
2770 C21E26	05490 BADHDR0	JP NZ,BADHDR	;Jump if bad symbol
2773 CDB128	05500	CALL FINDSYM	;Find in table
2776 C49428	05510	CALL NZ,MOVNAME	;Move name into table
2779 C37E25	05520 GOJCL	JP CPLJCL	
	05530 ;		
	05540 ;	Process RESET command	
	05550 ;		
277C CDE727	05560 RESET1	CALL PARSNAM	;Parse symbol name
277F 20EF	05570	JR NZ,BADHDR0	
2781 CDB128	05580	CALL FINDSYM	;Find symbol in table
2784 20F3	05590	JR NZ,GOJCL	;No problem if not there
2786 21F8FF	05600	LD HL,-8	;Point to start of name
2789 19	05610	ADD HL,DE	; & put in a blank
278A 3620	05620	LD (HL),' '	; to remove symbol
278C 18EB	05630	JR GOJCL	
	05640 ;		
	05650 ;	Process ASSIGN command	
	05660 ;		


```

278E CDE727 05670 ASSIGN CALL PARSNAM ;Parse symbol name
2791 20DD 05680 JR NZ,BADHDR0 ;Jump on bad name
2793 F5 05690 PUSH AF ;Save separator char
2794 CDB128 05700 CALL FINDSYM ;Find in table
2797 C49428 05710 CALL NZ,MOVNAME ;Add to table if not in
279A F1 05720 POP AF ;Recover separator
279B FE3D 05730 CP '=' ;Error if not =
279D 20D1 05740 JR NZ,BADHDR0
279F CDF227 05750 CALL PARSVAl ;Parse value of symbol
27A2 20CC 05760 JR NZ,BADHDR0
27A4 CDA628 05770 CALL MOVALUE ;Place value into table
27A7 18D0 05780 JR GOJCL
05790 ;
05800 ; Process INCLUDE command
05810 ;
27A9 E5 05820 INCLUD PUSH HL
27AA ED5BBA2A 05830 LD DE,(NESTPTR) ;Point to next FCB save
27AE 215C2B 05840 LD HL,NESTEND ; area & check if room
27B1 AF 05850 XOR A ; to store another FCB
27B2 ED52 05860 SBC HL,DE
27B4 CA4C29 05870 JP Z,NESTS ;Error if 5 nests already
27B7 217C29 05880 LD HL,DOFCB ;Shift current FCB into
27BA 012000 05890 LD BC,32 ; INCLUDE FCB save area
27BD EDB0 05900 LDIR
27BF ED53BA2A 05910 LD (NESTPTR),DE ;Update new nest pointer
27C3 E1 05920 POP HL
27C4 117C29 05930 LD DE,DOFCB ;Point to FCB
27C7 05940 @@FSPEC ;Fetch included file
27C7 3E4E 00045 LD A,78
27C9 EF 00046 RST 40
27CA 20A4 05950 JR NZ,BADHDR0 ;Jump on error
27CC 21D729 05960 LD HL,SYSJCL+7 ;Default to /JCL
27CF 05970 @@FEXT
27CF 3E4F 00047 LD A,79
27D1 EF 00048 RST 40
27D2 21002C 05980 LD HL,INPBUF ;Open the included file
27D5 45 05990 LD B,L
27D6 FDCB12C6 06000 SET 0,(IY+'S'-'A') ;Inhibit file open bit
27DA 06010 @@OPEN
27DA 3E3B 00049 LD A,59
27DC EF 00050 RST 40
27DD 2091 06020 JR NZ,BADHDR0
27DF 1898 06030 JR GOJCL
06040 ;
06050 ; Process QUIT command
06060 ;
27E1 218C2B 06070 QUIT LD HL,JCLBUF1 ;Log the //QUIT command
27E4 C36729 06080 JP EXTERR
06090 ;
06100 ; Parse symbol name
06110 ; A <= separator char
06120 ; Z = ok, NZ = bad symbol char
06130 ;
27E7 D5 06140 PARSNAM PUSH DE
27E8 0608 06150 LD B,8 ;8 chars max
27EA 119C29 06160 LD DE,CURSYM ;Symbol buffer area
27ED CD2E28 06170 CALL PARSEr ;Parse it
27F0 D1 06180 POP DE
27F1 C9 06190 RET
06200 ;
06210 ; Parse a symbol value

```

```

06220 ;
27F2 D5 06230 PARSVAL PUSH DE
27F3 0620 06240 LD B,32 ;32 chars max
27F5 11A529 06250 LD DE,VALBUF ;Value buffer
27F8 CD1128 06260 CALL XFRSTR ;Transfer from input
27FB F5 06270 PUSH AF
27FC E5 06280 PUSH HL
27FD EB 06290 EX DE,HL ;Calculate length of
27FE 11A529 06300 LD DE,VALBUF ; the string
2801 AF 06310 XOR A
2802 ED52 06320 SBC HL,DE
2804 7D 06330 LD A,L
2805 FE21 06340 CP 33
2807 D25029 06350 JP NC,TOOLNG ;Jump if > 32 chars
280A 32A429 06360 LD (STRLEN),A ;Stuff string length
280D E1 06370 POP HL
280E F1 06380 POP AF
280F D1 06390 POP DE
2810 C9 06400 RET
06410 ;
06420 ; Transfer a string field
06430 ;
2811 CD2E28 06440 XFRSTR CALL PARSE ;Xfer max of 32 chars
2814 CD2D29 06450 XFRSTR1 CALL CKSPCOM ;Return on space
2817 C8 06460 RET Z ; or comma
2818 FE0D 06470 CP CR
281A C8 06480 RET Z ;Ret on end of line
281B FE3D 06490 CP '='
281D C8 06500 RET Z ;Ret on =
281E FE28 06510 CP '('
2820 C8 06520 RET Z ;Ret on left paren
2821 FE29 06530 CP ')'
2823 C8 06540 RET Z ;Ret on right paren
2824 FE23 06550 CP '#'
2826 20E9 06560 JR NZ,XFRSTR ;Loop if not #
2828 CDC725 06570 CALL CPLJCL5 ;Ck on substitution
282B 7E 06580 LD A,(HL)
282C 18E6 06590 JR XFRSTR1 ;Then loop
06600 ;
06610 ; Parse a field
06620 ;
282E 78 06630 PARSE LD A,B ;Set max length of field
282F 329028 06640 LD (PAR6+1),A
2832 04 06650 INC B
2833 7E 06660 PAR2 LD A,(HL) ;P/u entry char
2834 FE03 06670 CP 3 ;ETX?
2836 284C 06680 JR Z,PAR5
2838 FE0D 06690 CP CR
283A 2848 06700 JR Z,PAR5
283C 23 06710 INC HL ;Not ending char, bump
283D FE22 06720 CP '"' ;Ck on string quote
283F 2007 06730 JR NZ,NOTQT
2841 EE22 06740 XOR A ;Ck if opening or closing
2842 06750 STUFQT EQU $-1
2843 324228 06760 LD (STUFQT),A
2846 18EB 06770 JR PAR2 ;Loop until terminator
2848 4F 06780 NOTQT LD C,A ;Save char & test if
2849 3A4228 06790 LD A,(STUFQT) ; within quoted string
284C B7 06800 OR A
284D 79 06810 LD A,C ;Get back the char
284E 2826 06820 JR Z,PAR3 ;Allow all within "..."

```

```

2850 FE40 06830 CP '0' ;Start of label?
2852 200D 06840 JR NZ,NOLBL
2854 D600 06850 GOTLBL SUB 0 ;Make sure only one
2856 CA5829 06860 JP Z,LBLERR
2859 325528 06870 LD (GOTLBL+1),A ;Stuff '&' into test
285C 320825 06880 LD (FNDLBL+1),A ; & also for check
285F 18D2 06890 JR PAR2 ;Loop through start
2861 FE2E 06900 NOLBL CP '.' ;Accept (., /, 0-9, :)
2863 381F 06910 JR C,PAR5
2865 FE3B 06920 CP ':' +1
2867 380D 06930 JR C,PAR3
2869 FE41 06940 CP 'A' ;Test for A-Z
286B 3817 06950 JR C,PAR5
286D FE5B 06960 CP 'Z' +1
286F 3805 06970 JR C,PAR3
2871 CD3329 06980 CALL CKLCA2Z ;Test for a-z
2874 380E 06990 JR C,PAR5
2876 05 07000 PAR3 DEC B ;Char count down
2877 2808 07010 JR Z,PAR4
2879 12 07020 LD (DE),A ;Save the char
287A AF 07030 XOR A ;Show we found at
287B 329028 07040 LD (PAR6+1),A ; least one valid char
287E 13 07050 INC DE ;Bump receiving buffer
287F 18B2 07060 JR PAR2 ;Loop
2881 04 07070 PAR4 INC B ;Ignore trailing chars
2882 18AF 07080 JR PAR2 ; past max length
2884 4F 07090 PAR5 LD C,A ;Found char out of range
2885 D5 07100 PUSH DE ;Save current end of buf
2886 1804 07110 JR PAR5B
2888 3E20 07120 PAR5A LD A,' ' ;Fill out remaining field
288A 12 07130 LD (DE),A ; with blanks
288B 13 07140 INC DE
288C 10FA 07150 PAR5B DJNZ PAR5A
288E D1 07160 POP DE ;Recover pointer to last
288F 3E00 07170 PAR6 LD A,0 ;Char xfered, get max len
2891 B7 07180 OR A ;Note if we found a char
2892 79 07190 LD A,C ;Xfer separator char
2893 C9 07200 RET
07210 ;
07220 ; Xfer symbol name to table & init value
07230 ;
2894 E5 07240 MOVNAME PUSH HL
2895 219C29 07250 LD HL,CURSYM ;Current symbol buffer
2898 010800 07260 LD BC,8 ;8 chars to move
289B EDB0 07270 LDIR
289D AF 07280 XOR A ;Zero accumulator
289E 12 07290 LD (DE),A ;Show symbol length=0
289F 212100 07300 LD HL,33 ;Point to 1st byte
28A2 19 07310 ADD HL,DE ; of next symbol pos and
28A3 77 07320 LD (HL),A ; show it spare
28A4 E1 07330 POP HL
28A5 C9 07340 RET
07350 ;
07360 ; Place symbol value into table
07370 ;
28A6 E5 07380 MOVALUE PUSH HL
28A7 21A429 07390 LD HL,STRLEN ;Current value buffer
28AA 012100 07400 LD BC,33 ;Length & value
28AD EDB0 07410 LDIR
28AF E1 07420 POP HL
28B0 C9 07430 RET

```

```

07440 ;
07450 ; Find symbol in table
07460 ;
28B1 E5 07470 FINDSYM PUSH HL
28B2 119C29 07480 LD DE,CURSYM ;Symbol buffer
28B5 21002E 07490 LD HL,SYMTAB ;Start of table
28B8 012908 07500 LD BC,8<8!41 ;CP8, field (8,1,32)
28BB CDDD28 07510 CALL FNDPRM ;Search in progress
28BE 54 07520 LD D,H ;Xfer pointer of symbol
28BF 5D 07530 LD E,L ; or to spare slot
28C0 E1 07540 POP HL
28C1 C9 07550 RET
07560 ;
07570 ; Routine to check for IF, ELSE, END
07580 ;
28C2 E5 07590 CK4COND PUSH HL
28C3 21C126 07600 LD HL,COND_TBL ;Parm table
28C6 010705 07610 LD BC,5<8!7 ;5 chars, 7-char field
28C9 1807 07620 JR CK4AS1
07630 ;
07640 ; Check on SET, RESET, ASSIGN, INCLUDE, QUIT
07650 ;
28CB E5 07660 CK4ASSN PUSH HL
28CC 21D726 07670 LD HL,ASSNTBL ;Parm table
28CF 010A08 07680 LD BC,8<8!10 ;Parm length, field len
28D2 119C29 07690 CK4AS1 LD DE,CURSYM ;Buffer area
28D5 CDDD28 07700 CALL FNDPRM ;Ck for match
28D8 5E 07710 LD E,(HL) ;Xfer vector address
28D9 23 07720 INC HL
28DA 56 07730 LD D,(HL)
28DB E1 07740 POP HL
28DC C9 07750 RET
07760 ;
07770 ; Scan parm table for match
07780 ;
28DD 7E 07790 FNDPRM LD A,(HL) ;End of parm table?
28DE B7 07800 OR A
28DF 2002 07810 JR NZ,FND1 ;Jump if not
28E1 3C 07820 INC A ; else show not found
28E2 C9 07830 RET
28E3 1A 07840 FND1 LD A,(DE) ;Char match?
28E4 CD3329 07850 CALL CKLCA2Z ;Convert a-z to A-Z
28E7 BE 07860 CP (HL)
28E8 2807 07870 JR Z,FND3 ;Jump if 1st matches
28EA C5 07880 FND2 PUSH BC ; else bypass complete
28EB 0600 07890 LD B,0 ; field & go to next one
28ED 09 07900 ADD HL,BC
28EE C1 07910 POP BC
28EF 18EC 07920 JR FNDPRM
28F1 E5 07930 FND3 PUSH HL ;1st matches, ck rest
28F2 D5 07940 PUSH DE
28F3 C5 07950 PUSH BC
28F4 05 07960 DEC B ;Adj for 1st match
28F5 13 07970 FND4 INC DE
28F6 23 07980 INC HL
28F7 1A 07990 LD A,(DE)
28F8 FE20 08000 CP ' '
28FA 2827 08010 JR Z,FND7 ;Stop checking on space
28FC FE0D 08020 CP CR
28FE 2823 08030 JR Z,FND7 ;Or end of line
2900 CD3329 08040 CALL CKLCA2Z ;Ck & convert a-z to A-Z

```

```

2903 BE      08050      CP      (HL)      ;Compare remaining chars
2904 200D    08060      JR      NZ,FND6    ;Jump on mismatch
2906 10ED    08070      DJNZ    FND4       ;Loop to count
2908 C1      08080 FND5  POP      BC       ;Must have matched
2909 D1      08090      POP      DE       ;Bypass remaining part
290A E1      08100      POP      HL       ; of field and point to
290B C5      08110      PUSH     BC       ; address vector of parm
290C 48      08120      LD       C,B      ; in parm table
290D 0600    08130      LD       B,0
290F 09      08140      ADD      HL,BC
2910 C1      08150      POP      BC
2911 AF      08160      XOR      A
2912 C9      08170      RET
2913 FE30    08180 FND6  CP      '0'      ;No match, is it ASCII?
2915 380C    08190      JR      C,FND7
2917 FE3A    08200      CP      '9'+1     ;0-9?
2919 380D    08210      JR      C,FND8
291B FE41    08220      CP      'A'      ;A-Z?
291D 3804    08230      JR      C,FND7
291F FE5B    08240      CP      'Z'+1
2921 3805    08250      JR      C,FND8
2923 7E      08260 FND7  LD      A,(HL)   ;If table entry also a
2924 FE20    08270      CP      ' '      ; space, we have a match
2926 28E0    08280      JR      Z,FND5
2928 C1      08290 FND8  POP      BC
2929 D1      08300      POP      DE
292A E1      08310      POP      HL
292B 18BD    08320      JR      FND2
                08330      ENDIF
                08340 ;
                08350 ;      Routine to ck on space or comma
                08360 ;
292D FE20    08370 CKSPCOM CP      ' '
292F C8      08380      RET      Z
2930 FE2C    08390      CP      ','
2932 C9      08400      RET
                08410 ;
                08420 ;      Routine to convert a-z to A-Z & set C-flag
                08430 ;
2933 FE61    08440 CKLCA2Z CP      'a'      ;Back with C-flag if
2935 D8      08450      RET      C       ; not a-z
2936 FE7B    08460      CP      'z'+1
2938 3F      08470      CCF
2939 D8      08480      RET      C
293A EE20    08490      XOR      20H     ;Make U/C & reset CF
293C C9      08500      RET
                08510 *LIST      ON
                08520 ;
                08530 ;
                08540 ;      Error processing
                08550 ;
293D 6F      08560 IOERR LD      L,A      ;Xfer errnum to HL
293E 2600    08570      LD      H,0
2940 F6C0    08580      OR      0C0H     ;Set brief, return
2942 4F      08590      LD      C,A
2943         08600      @@ERROR    ;Display error
2943 3E1A    00051      LD      A,26
2945 EF      00052      RST      40
2946 1825    08610      JR      ERREXIT
                08620 ;
2948 21DB29  08630 SPCREQ LD      HL,SPCREQ$ ;"filespec required"

```

```

08640 ;
08650 *LIST OFF
08670 *LIST ON
294B DD 08680 DB 0DDH
294C 21A12A 08690 NESTS LD HL,NESTS$
294F DD 08700 DB 0DDH
2950 21FC29 08710 TOOLNG LD HL,TOOLNG$ ;"symbol too long..
2953 DD 08720 DB 0DDH
2954 21132A 08730 NOFIND LD HL,NOFIND$ ;"proc not found..
2957 DD 08740 DB 0DDH
2958 21272A 08750 LBLERR LD HL,LBLERR$ ;"too many proc labels..
295B DD 08760 DB 0DDH
295C 213C2A 08770 DSKFUL LD HL,DSKFUL$ ;"can't create SYS/JCL"
295F DD 08780 DB 0DDH
2960 216A2A 08790 PRMERR LD HL,PRMERR$ ;"parameter error"
2963 DD 08800 DB 0DDH
2964 21592A 08810 MULDEF LD HL,MULDEF$ ;"multiply defined
08820 ENDIF
08830 *LIST ON
08840 ;
2967 08850 EXTERR @@LOGOT
00053 IFEQ 00H,1
00054 LD HL,
00055 ENDIF
2967 3E0C 00056 LD A,12
2969 EF 00057 RST 40
296A 21FFFF 08860 LD HL,-1 ;Set error exit
296D 08870 ERREXIT EQU $
296D 11C000 08880 LD DE,JFCB$ ;If the output JCL file
2970 1A 08890 LD A,(DE) ; is open, then we need
2971 CB7F 08900 BIT 7,A ; to close it
2973 2803 08910 JR Z,SPSAV
2975 08920 @@CLOSE
2975 3E3C 00058 LD A,60
2977 EF 00059 RST 40
2978 310000 08930 SPSAV LD SP,$-$
297B C9 08940 RET
08950 ;
08960 *LIST OFF
08980 *LIST ON
0020 08990 DOFCB DS 32
0008 09000 CURSYM DS 8
0001 09010 STRLEN DS 1
0020 09020 VALBUF DS 32
0008 09030 LBLSAV DS 8
29CD 00 09040 NOP ;Must be zero
09050 ENDIF
09060 ;
09070 *LIST ON
29CE 0000 09080 LINENO DW 0 ;JCL line #
29D0 53 09090 SYSJCL DB 'SYSTEM/JCL',3
59 53 54 45 4D 2F 4A 43
4C 03
29DB 46 09100 SPCREQ$ DB 'File spec required',CR
69 6C 65 20 73 70 65 63
20 72 65 71 75 69 72 65
64 0D
09110 *LIST OFF
09130 *LIST ON
29EE 4C 09140 LINLNG$ DB 'Line too long',CR
69 6E 65 20 74 6F 6F 20

```

29FC	53	6C 6F 6E 67 0D	09150 TOOLNG\$ DB	'Symbol string too long',CR
	79	6D 62 6F 6C 20 73 74		
	72	69 6E 67 20 74 6F 6F		
	20	6C 6F 6E 67 0D		
2A13	50		09160 NOFIND\$ DB	'Procedure not found',CR
	72	6F 63 65 64 75 72 65		
	20	6E 6F 74 20 66 6F 75		
	6E	64 0D		
2A27	54		09170 LBLERR\$ DB	'Too many Proc labels',CR
	6F	6F 20 6D 61 6E 79 20		
	50	72 6F 63 20 6C 61 62		
	65	6C 73 0D		
2A3C	43		09180 DSKFUL\$ DB	'Can''t create SYSTEM/JCL file',CR
	61	6E 27 74 20 63 72 65		
	61	74 65 20 53 59 53 54		
	45	4D 2F 4A 43 4C 20 66		
	69	6C 65 0D		
2A59	4D		09190 MULDEF\$ DB	'Multiply defined ' ;Follow with PRMERR\$
	75	6C 74 69 70 6C 79 20		
	64	65 66 69 6E 65 64 20		
2A6A	50		09200 PRMERR\$ DB	'Parameter error',CR
	61	72 61 6D 65 74 65 72		
	20	65 72 72 6F 72 0D		
2A7A	49		09210 BADJCL\$ DB	'Invalid JCL format, processing aborted',CR
	6E	76 61 6C 69 64 20 4A		
	43	4C 20 66 6F 72 6D 61		
	74	2C 20 70 72 6F 63 65		
	73	73 69 6E 67 20 61 62		
	6F	72 74 65 64 0D		
2AA1	54		09220 NESTS\$ DB	'Too many nested INCLUDEs',CR
	6F	6F 20 6D 61 6E 79 20		
	6E	65 73 74 65 64 20 49		
	4E	43 4C 55 44 45 73 0D		
2ABA	BC2A		09230 NESTPTR DW	NESTFCB ;Pointer to nest FCB
00A0			09240 NESTFCB DS	32*5 ;Space for 5 levels
2B5C			09250 NESTEND EQU	\$;Ck for too many includes
2B5C	5E2B		09260 CONDPTR DW	CONDFLG ;Conditional pointer
2B5E	00		09270 CONDFLG DB	0 ;Init 1st state to TRUE
001F			09280 DS	31 ;32 conditional levels
2B7E	4C		09290 BADHDR\$ DB	'Line xxxxx -->'
	69	6E 65 20 78 78 78 78		
	78	20 2D 2D 3E		
0050			09300 JCLBUF1 DS	80
2C00			09310 ORG	\$<-8+1<+8
0100			09320 INPBUF DS	256
0100			09330 OUTBUF DS	256
2E00	00		09340 SYMTAB DB	0
			09350 ENDIF	
			09360 *LIST ON	
2E01			09370 CORE\$ DEFL	\$
			09380 ;	
2400			09390 END	DO

@@1	0000 @@2	0000 @@3	0000
@@4	0000 @MOD2	0000 @MOD4	FFFF
ASSIGN	278E ASSNTBL	26D7 BADH1	262D
BADHDR	261E BADHDR\$	2B7E BADHDR0	2770
BADJCL\$	2A7A CK4AS1	28D2 CK4ASSN	28CB
CK4COND	28C2 CKCOND	2612 CKLCA2Z	2933
CKNEST	2499 CKSPCOM	292D CONDCPL	257B
CONDFLG	2B5E CONDPTR	2B5C CONDTBL	26C1
CORE\$	2E01 CPLFIN	24AA CPLFIN1	24BA
CPLJCL	257E CPLJCL1	2599 CPLJCL2	25B8
CPLJCL3	25BE CPLJCL4	25C2 CPLJCL5	25C7
CPLJCL6	25CA CPLJCL7	25A6 CR	000D
CURSYM	299C CVRTHE1	25DD CVRTHEX	25CE
DO	2400 DOFCB	297C DSKFUL	295C
DSKFUL\$	2A3C ELSE1	2733 END1	273E
ERREXIT	296D EXTERR	2967 FINDLBL	254E
FINDSYM	28B1 FND1	28E3 FND2	28EA
FND3	28F1 FND4	28F5 FND5	2908
FND6	2913 FND7	2923 FND8	2928
FNDLBL	2507 FNDPRM	28DD GETSEP	2532
GETSPEC	2423 GOJCL	2779 GOTLBL	2854
GOTLIN	2564 IF01	270A IF01A	271B
IF02	2723 IF03	2726 IF04	2728
IF05	2747 IF06	2751 IF07	275D
IF08	2760 INBUF	246C INCLUD	27A9
INPBUF	2C00 IOERR	293D JCLBUF1	2B8C
JCLBUF2	2400 JFCB\$	00C0 LBLERR	2958
LBLERR\$	2A27 LBLSAV	29C5 LINENO	29CE
LINLNG\$	29EE MACRO	2633 MACRO2	264E
MOVVALUE	28A6 MOVFCB	24F5 MOVLBL	2535
MOVNAME	2894 MULDEF	2964 MULDEF\$	2A59
NESTEND	2B5C NESTFCB	2ABC NESTPTR	2ABA
NESTS	294C NESTS\$	2AA1 NOCPL	24D9
NOCPL1	24DF NOCPL2	24F0 NOCPLS	24DA
NOEXEC?	24B3 NOFIND	2954 NOFIND\$	2A13
NOLBL	2861 NOTQT	2848 OUTBUF	2D00
PAR2	2833 PAR3	2876 PAR4	2881
PAR5	2884 PAR5A	2888 PAR5B	288C
PAR6	288F PARAM	2501 PARAM1	251A
PARAM2	252B PARSER	282E PARSINP	2451
PARSNAM	27E7 PARSVAL	27F2 PRMERR	2960
PRMERR\$	2A6A QUIT	27E1 RDJCL	2661
RDJCL1	2670 RDJCL2	2689 RDJCL3	268E
RESET1	277C SET1	276D SMALL	0000
SPCREQ	2948 SPCREQ\$	29DB SPSAV	2978
STRLEN	29A4 STUFQT	2842 SUBSYM	25DF
SUBSYM1	25FE SUBSYM2	2603 SYMTAB	2E00
SYSJCL	29D0 TOOLNG	2950 TOOLNG\$	29FC
TSTCOND	2692 TSTLBL	2548 UNNEST	247F
VALBUF	29A5 WRCPLD	26B3 XFRSTR	2811
XFRSTR1	2814 @@ABORT	BF67 @@ADTSK	BFFA
@@BANK	C512 @@BKSP	C1F2 @@BREAK	C528
@@CHNIO	BF52 @@CKBRKC	C576 @@CKDRV	C04E
@@CKEOF	C207 @@CKTSK	BFE5 @@CLOSE	C1DD
@@CLS	C560 @@CMNDI	BF91 @@CMNDR	BFA6
@@CTL	BDB6 @@DATE	BF28 @@DCSTAT	C08D
@@DEBUG	BFD0 @@DECHEX	C492 @@DIRRD	C3FF
@@DIRWR	C414 @@DIV16	C47D @@DIV8	C468
@@DODIR	C063 @@DSP	BD7A @@DSPLY	BE1A
@@ERROR	BFBB @@EXIT	BF7C @@FEXT	C36C

@@FLAGS	C4FC @@FNAME	C381 @@FSPEC	C357
@@GATRD	C3EA @@GATWR	C429 @@GET	BD8E
@@GTDCB	C3AB @@GTDCB	C396 @@GTMOD	C3C0
@@HDFMT	C135 @@HEX16	C4D1 @@HEX8	C4BC
@@HEXDEC	C4A7 @@HIGH\$	C4E6 @@INIT	C1B3
@@KBD	BDF2 @@KEY	BD66 @@KEYIN	BE06
@@KLTSK	C039 @@LOAD	C32D @@LOC	C21C
@@LOF	C231 @@LOGGER	BE51 @@LOGOT	BE66
@@MSG	BE9D @@MUL16	C453 @@MUL8	C43E
@@OPEN	C1C8 @@PARAM	BF13 @@PAUSE	BEFE
@@PEOF	C246 @@POSN	C25B @@PRINT	BEB2
@@PRT	BDCA @@PUT	BDA2 @@RAMDIR	C078
@@RDSEC	C10B @@RDSSC	C3D5 @@READ	C270
@@REMOV	C19E @@RENAME	C189 @@REW	C285
@@RMTSK	C00F @@RPTSK	C024 @@RREAD	C29A
@@RSLCT	C0F6 @@RSTOR	C0B7 @@RUN	C342
@@RWIT	C2AF @@SEEK	C0E1 @@SEEKSC	C2C4
@@SKIP	C2D9 @@SLCT	C0A2 @@STEPI	C0CC
@@TIME	BF3D @@VDCTL	BEE9 @@VER	C2EE
@@VRSEC	C120 @@WEOF	C303 @@WHERE	BDDE
@@WRITE	C318 @@WRSEC	C14A @@WRSSC	C15F
@@WRTRK	C174		

2400 is the transfer address

00000 Total errors

NOTES:

NOTES:

Command: DUMP

Library: SYS7/SYS

ISAM # : 71H

```

00100 ;LBDUMP/ASM - DUMP Command
0000 00110 TITLE <DUMP - LS-DOS 6.2>
00120 ;
0000 00130 *GET SVCMAC:3 ;SVC Macro equivalents
00010 ;SVC MAC/ASM - LS-DOS Version VI
00020 *LIST OFF
03900 *LIST ON
00140 ;
000D 00150 CR EQU 13
002C 00160 PAR_ERR EQU 44 ;"Parameter Error" #
3000 00170 SINIT EQU 3000H
3000 00180 EINIT EQU SINIT
00190 ;
2400 00200 ORG 2400H
00210 ;
2400 00220 DUMP @@CKBRKC ;Break key down?
2400 3E6A 00001 LD A,106
2402 EF 00002 RST 40
2403 2804 00230 JR Z,BEGINA ;Ok if not
2405 21FFFF 00240 LD HL,-1 ; else abort
2408 C9 00250 RET
00260 ;
2409 111C26 00270 BEGINA LD DE,FCB1 ;Fetch the filespec
240C 00280 @@FSPEC
240C 3E4E 00003 LD A,78
240E EF 00004 RST 40
240F C26E25 00290 JP NZ,SPCREQ ;Jump on error
2412 1A 00300 LD A,(DE) ;Cannot be a device
2413 FE2A 00310 CP '*'
2415 CA6E25 00320 JP Z,SPCREQ ;Quit if device
2418 00330 @@FLAGS ;Get system flag table
2418 3E65 00005 LD A,101
241A EF 00006 RST 40
241B E5 00340 PUSH HL ;Save cmdline ptr
241C FD661A 00350 LD H,(IY+26) ;P/u SVC table MSB
241F 2E2C 00360 LD L,22*2 ; & point to @EXIT entry
2421 7E 00370 LD A,(HL) ;Get @EXIT LSB
2422 2C 00380 INC L
2423 66 00390 LD H,(HL) ;Get @EXIT MSB
2424 6F 00400 LD L,A
2425 224025 00410 LD (TPARM+1),HL ;Init transfer to @EXIT
2428 E1 00420 POP HL
00430 ;
00440 ; Search for parameters
00450 ;
2429 11BD25 00460 LD DE,PRMTBL$ ;Get the parms
242C 00470 @@PARAM
242C 3E11 00007 LD A,17
242E EF 00008 RST 40
242F C25A25 00480 JP NZ,PRMERR ;Jump on parm error
2432 2AE024 00490 LD HL,(EPARM+1) ;Ck on end > start
2435 ED4BDA24 00500 LD BC,(SPARM+1)
2439 AF 00510 XOR A
243A ED42 00520 SBC HL,BC
243C DA6A25 00530 JP C,ENLTST ;Jump on start < end
243F 210030 00540 LD HL,SINIT ;Pt to lowest possible
2442 2B 00550 DEC HL ;Reduce for compare
2443 ED42 00560 SBC HL,BC ;Ck on start > minimum
2445 D26625 00570 JP NC,STLT30 ;Jump if start < minimum
2448 010000 00580 APARM LD BC,0 ;ASCII txt or code cim
244B 78 00590 LD A,B

```

```

244C B1      00600      OR      C
244D 202B    00610      JR      NZ,DUMPTXT      ;Go if ASCII
244F 111626  00620      LD      DE,NAMFLD      ;Get up to a 6-character
2452 211C26  00630      LD      HL,FCB1      ;Filename to stuff
2455 0606    00640      LD      B,6          ;As file header
2457 7E      00650 $?1    LD      A,(HL)
2458 FE30    00660      CP      '0'          ;Stop on non-alpha
245A 3813    00670      JR      C,$?3
245C FE3A    00680      CP      '9'+1        ;Use if 0-9
245E 3808    00690      JR      C,$?2
2460 FE41    00700      CP      'A'          ;Ck on A-Z
2462 380B    00710      JR      C,$?3
2464 FE5B    00720      CP      'Z'+1
2466 3007    00730      JR      NC,$?3        ;Exit if not A-Z
2468 12      00740 $?2    LD      (DE),A      ;Xfer this char
2469 23      00750      INC     HL          ;Bump input ptr
246A 13      00760      INC     DE          ; & output pointer
246B 10EA    00770      DJNZ   $?1          ;Loop 6-chars max
246D 1806    00780      JR      $?4
246F 3E20    00790 $?3    LD      A,' '          ;Place blanks to
2471 12      00800      LD      (DE),A      ;Fill out to 6 chars
2472 13      00810      INC     DE
2473 10FA    00820      DJNZ   $?3
2475 21B725  00830 $?4    LD      HL,LMFEXT      ;Use /LMF extension
2478 1803    00840      JR      DUMPCIM
247A 21BA25  00850 DUMPTXT LD      HL,TXTEXT      ;Use /TXT extension
247D 111C26  00860 DUMPCIM LD      DE,FCB1      ;Default the EXT
2480         00870      @@FEXT
2480 3E4F    00009      LD      A,79
2482 EF      00010      RST     40
2483 010000  00880 LPARM    LD      BC,0          ;P/u LRL
2486 78      00890      LD      A,B          ;Test for > 256
2487 B7      00900      OR      A          ;If hi-order = 0,
2488 2808    00910      JR      Z,LP1        ;Just use lo-order
248A 3D      00920      DEC     A          ;Test hi-order = 1
248B C25A25  00930      JP      NZ,PRMERR
248E B1      00940      OR      C          ;P/u lo-order
248F C25A25  00950      JP      NZ,PRMERR      ;Lo-order must be 0
2492 B1      00960 LP1    OR      C          ;Merge lo-order
2493 47      00970      LD      B,A
2494 210027  00980      LD      HL,BUFFER      ;Pt to buffer
2497         00990      @@INIT      ;Init the file
2497 3E3A    00011      LD      A,58
2499 EF      00012      RST     40
249A C25C25  01000      JP      NZ,IOERR      ;Quit on init error
                01010 ;
                01020 ;      Display the filespec being dumped
                01030 ;
249D ED4B2226 01040      LD      BC,(FCB1+6)    ;P/u DEC & drive
24A1 11F625  01050      LD      DE,FCB2      ;Point to FCB area
24A4 D5      01060      PUSH   DE
24A5         01070      @@FNAME      ;Fetch the name
24A5 3E50    00013      LD      A,80
24A7 EF      00014      RST     40
24A8 E1      01080      POP     HL
24A9 C25C25  01090      JP      NZ,IOERR      ;Quit on error
24AC 3E20    01100      LD      A,20H          ;Scan until ETX char
24AE 23      01110 FNLP   INC     HL
24AF BE      01120      CP      (HL)
24B0 38FC    01130      JR      C,FNLP
24B2 360D    01140      LD      (HL),CR      ;Replace with CR

```

```

24B4      01150      @LOGOT DUMP$      ;Display "Dumping..."
          00015      IFEQ      01H,1
24B4 21ED25 00016      LD        HL,DUMP$
          00017      ENDIF
24B7 3E0C    00018      LD        A,12
24B9 EF      00019      RST      40
          01160      ;

24BA 111C26 01170      LD        DE,FCB1      ;Get dump FCB
24BD 3A4924 01180      LD        A,(APARM+1)    ;Ck if ASCII parm used
24C0 B7      01190      OR        A
24C1 2016    01200      JR        NZ,SPARM      ; and go if so
24C3 3E05    01210      LD        A,5          ;Name header
24C5 CD5325 01220      CALL     PUTOUT
24C8 3E06    01230      LD        A,6          ;Name length
24CA CD5325 01240      CALL     PUTOUT
24CD 0606    01250      LD        B,6          ;Init loop
24CF 211626 01260      LD        HL,NAMFLD
24D2 7E      01270      LD        A,(HL)
24D3 23      01280      INC      HL
24D4 CD5325 01290      CALL     PUTOUT      ;Output the filename
24D7 10F9    01300      DJNZ     $?5
          01310      ;

24D9 210030 01320      SPARM    LD        HL,SINIT      ;P/u starting addr
24DC E5      01330      $?7    PUSH     HL          ;Ck on write of
24DD 44      01340      LD        B,H          ; last byte written
24DE 4D      01350      LD        C,L
24DF 21FF2F 01360      EPARM    LD        HL,EINIT-1    ;Where to end
24E2 23      01370      INC      HL
24E3 AF      01380      XOR      A
24E4 ED42    01390      SBC      HL,BC
24E6 282E    01400      JR        Z,$?10      ;Go if at end
24E8 06FE    01410      LD        B,254      ;254-byte blocks
24EA 7C      01420      LD        A,H          ;A full sector left
24EB B7      01430      OR        A          ;To write?
24EC 2006    01440      JR        NZ,$?8
24EE 7D      01450      LD        A,L
24EF FEFF    01460      CP        0FFH
24F1 3001    01470      JR        NC,$?8      ;If less than full,
24F3 45      01480      LD        B,L          ; reset len
24F4 E1      01490      $?8    POP      HL
24F5 3A4924 01500      LD        A,(APARM+1)
24F8 B7      01510      OR        A
24F9 2012    01520      JR        NZ,$?9      ;Bypass if TXT
24FB 3C      01530      INC      A          ;Init start of block
24FC CD5325 01540      CALL     PUTOUT
24FF 78      01550      LD        A,B          ;Get block length
2500 C602    01560      ADD      A,2          ;Add 2 for load address
2502 CD5325 01570      CALL     PUTOUT      ; & put it out
2505 7D      01580      LD        A,L
2506 CD5325 01590      CALL     PUTOUT      ;Lo-order load address
2509 7C      01600      LD        A,H
250A CD5325 01610      CALL     PUTOUT      ;Hi-order load address
250D 7E      01620      $?9    LD        A,(HL)      ;Write a load block
250E 23      01630      INC      HL
250F CD5325 01640      CALL     PUTOUT
2512 10F9    01650      DJNZ     $?9
2514 18C6    01660      JR        $?7      ;Loop for more
          01670      ;

2516 E1      01680      $?10   POP      HL          ;Stack integrity
2517 3A4924 01690      LD        A,(APARM+1)    ;No TRAADR if TXT
251A B7      01700      OR        A          ; or TRAADR if not TXT

```

The Source	LIBRARY Files	DUMP - LS-DOS 6.2	Page 00004
251B 2818	01710 JR	Z,TRAADR	
251D 010300	01720 ETXPARM LD	BC,3	;P/u etx character
2520 79	01730 LD	A,C	
2521 21E925	01740 LD	HL,ETXRESP	
2524 CB7E	01750 BIT	7,(HL)	;Value input means
2526 201F	01760 JR	NZ,PUTETX	; put the ETX given
2528 0A	01770 LD	A,(BC)	;In case string
2529 CB6E	01780 BIT	5,(HL)	;String input puts the
252B 201A	01790 JR	NZ,PUTETX	; entered char
252D CB76	01800 BIT	6,(HL)	;Flag input gives ETX=3
252F 2819	01810 JR	Z,CLSFIL	; if ETX=on
2531 3E03	01820 LD	A,3	
2533 1812	01830 JR	PUTETX	
2535 3E02	01840 TRAADR LD	A,2	;Put traadr header
2537 CD5325	01850 CALL	PUTOUT	
253A 3E02	01860 LD	A,2	
253C CD5325	01870 CALL	PUTOUT	
253F 210000	01880 TPARM LD	HL,\$-\$;P/u transfer address
2542 7D	01890 LD	A,L	
2543 CD5325	01900 CALL	PUTOUT	;Tra lo-order
2546 7C	01910 LD	A,H	
2547 CD5325	01920 PUTETX CALL	PUTOUT	;Tra hi-order or ETX
254A	01930 CLSFIL @@CLOSE		;Close 'er up
254A 3E3C	00020 LD	A,60	
254C EF	00021 RST	40	
254D 210000	01940 LD	HL,0	
2550 C8	01950 RET	Z	;Back on no error
2551 1809	01960 JR	IOERR	;Go on error
	01970 ;		
2553 4F	01980 PUTOUT LD	C,A	;Xfer the char
2554	01990 @@PUT		;Test each byte transfer
2554 3E04	00022 LD	A,4	
2556 EF	00023 RST	40	
2557 C8	02000 RET	Z	;Back if no error
2558 E1	02010 POP	HL	;Pop the RET addr
2559 21	02020 DB	21H	;Skip LD A,## instruction
255A 3E2C	02030 PRMERR LD	A,PAR_ERR	; "Parameter Error"
255C 6F	02040 IOERR LD	L,A	;Error code to HL
255D 2600	02050 LD	H,0	
255F F6C0	02060 OR	0C0H	;Abbrev & return
2561 4F	02070 LD	C,A	
2562	02080 @@ERROR		;Show the error
2562 3E1A	00024 LD	A,26	
2564 EF	00025 RST	40	
2565 C9	02090 RET		
	02100 ;		
	02110 ;	Internal error routine	
	02120 ;		
2566 218C25	02130 STLT30 LD	HL,STLT30\$	
2569 DD	02140 DB	0DDH	
256A 217825	02150 ENLTST LD	HL,ENLTST\$	
256D DD	02160 DB	0DDH	
256E 21A425	02170 SPCREQ LD	HL,SPCREQ\$	
2571	02180 @@LOGOT		
	00026 IFEQ	00H,1	
	00027 LD	HL,	
	00028 ENDIF		
2571 3E0C	00029 LD	A,12	
2573 EF	00030 RST	40	
2574 21FFFF	02190 LD	HL,-1	
2577 C9	02200 RET		


```

                02210 ;
2578 53          02220 ENLTST$ DB      'START or END error ',CR
      54 41 52 54 20 6F 72 20
      45 4E 44 20 65 72 72 6F
      72 20 0D
258C 53          02230 STLT30$ DB      'Start less than X''3000'',CR
      74 61 72 74 20 6C 65 73
      73 20 74 68 61 6E 20 58
      27 33 30 30 30 27 0D
25A4 46          02240 SPCREQ$ DB      'File spec required',CR
      69 6C 65 20 73 70 65 63
      20 72 65 71 75 69 72 65
      64 0D
25B7 4C          02250 LMFEXT DB      'LMF '
      4D 46
25BA 54          02260 TXTEXT DB      'TXT '
      58 54

                02270 ;
0080          02280 VAL EQU      80H
0040          02290 SW EQU      40H
0020          02300 STR EQU      20H
0010          02310 SGL EQU      10H
                02320 ;
25BD 80          02330 PRMTBL$ DB      80H
25BE 95          02340 DB      VAL!SGL!5,'START',0
      53 54 41 52 54 00
25C5 DA24        02350 DW      SPARM+1
25C7 93          02360 DB      VAL!SGL!3,'END',0
      45 4E 44 00
25CC E024        02370 DW      EPARM+1
25CE 93          02380 DB      VAL!SGL!3,'TRA',0
      54 52 41 00
25D3 4025        02390 DW      TPARM+1
25D5 55          02400 DB      SW!SGL!5,'ASCII',0
      41 53 43 49 49 00
25DC 4924        02410 DW      APARM+1
25DE 93          02420 DB      VAL!SGL!3,'LRL',0
      4C 52 4C 00
25E3 8424        02430 DW      LPARM+1
25E5 83          02440 DB      VAL!3,'ETX',0
      45 54 58 00
25E9            02450 ETXRESP EQU    $-1
25EA 1E25        02460 DW      ETXPARM+1
25EC 00          02470 NOP
                02480 ;
25ED 44          02490 DUMP$ DB      'Dumping: ' ;FCB2 must follow
      75 6D 70 69 6E 67 3A 20
0020          02500 FCB2 DS      32
0006          02510 NAMFLD DS      6
0020          02520 FCB1 DS      32
2700          02530 ORG      $<-8+1<+8
0100          02540 BUFFER DS      256
27FF          02550 LAST EQU    $-1
                02560 ;
2400          02570 END      DUMP

```

\$?1	2457	\$?10	2516	\$?2	2468
\$?3	246F	\$?4	2475	\$?5	24D2
\$?7	24DC	\$?8	24F4	\$?9	250D
@@1	0000	@@2	0000	@@3	0000
@@4	0000	@MOD2	0000	@MOD4	FFFF
APARM	2448	BEGINA	2409	BUFFER	2700
CLSF IL	254A	CR	000D	DUMP	2400
DUMPS	25ED	DUMPCIM	247D	DUMPTXT	247A
EINIT	3000	ENLTST	256A	ENLTST\$	2578
EPARM	24DF	ETXPARM	251D	ETXRESP	25E9
FCB1	261C	FCB2	25F6	FNLP	24AE
IOERR	255C	LAST	27FF	LMFEXT	25B7
LP1	2492	LPARM	2483	NAMFLD	2616
PAR ERR	002C	PRMERR	255A	PRMTBL\$	25BD
PUTETX	2547	PUTOUT	2553	SGL	0010
SINIT	3000	SPARM	24D9	SPCREQ	256E
SPCREQ\$	25A4	STLT30	2566	STLT30\$	258C
STR	0020	SW	0040	TPARM	253F
TRAADR	2535	TXTEXT	25BA	VAL	0080
@@ABORT	8160	@@ADTSK	81F3	@@BANK	870B
@@BKSP	83EB	@@BREAK	8721	@@CHNIO	814B
@@CKBRKC	876F	@@CKDRV	8247	@@CKEOF	8400
@@CKTSK	81DE	@@CLOSE	83D6	@@CLS	8759
@@CMNDI	818A	@@CMNDR	819F	@@CTL	7FAF
@@DATE	8121	@@DCSTAT	8286	@@DEBUG	81C9
@@DECHX	868B	@@DIRRD	85F8	@@DIRWR	860D
@@DIV16	8676	@@DIV8	8661	@@DODIR	825C
@@DSP	7F73	@@DSPLY	8013	@@ERROR	81B4
@@EXIT	8175	@@FEXT	8565	@@FLAGS	86F5
@@FNAME	857A	@@FSPEC	8550	@@GATRD	85E3
@@GATWR	8622	@@GET	7F87	@@GTDCB	85A4
@@GTDCT	858F	@@GTMOD	85B9	@@HDFMT	832E
@@HEX16	86CA	@@HEX8	86B5	@@HEXDEC	86A0
@@HIGH\$	86DF	@@INIT	83AC	@@KBD	7FEB
@@KEY	7F5F	@@KEYIN	7FFF	@@KLTSK	8232
@@LOAD	8526	@@LOC	8415	@@LOF	842A
@@LOGGER	804A	@@LOGOT	805F	@@MSG	8096
@@MUL16	864C	@@MUL8	8637	@@OPEN	83C1
@@PARAM	810C	@@PAUSE	80F7	@@PEOF	843F
@@POSN	8454	@@PRINT	80AB	@@PRT	7FC3
@@PUT	7F9B	@@RAMDIR	8271	@@RDSEC	8304
@@RDSSC	85CE	@@READ	8469	@@REMOV	8397
@@RENAM	8382	@@REW	847E	@@RMTSK	8208
@@RPTSK	821D	@@RREAD	8493	@@RSLCT	82EF
@@RSTOR	82B0	@@RUN	853B	@@RWRIT	84A8
@@SEEK	82DA	@@SEEKSC	84BD	@@SKIP	84D2
@@SLCT	829B	@@STEPI	82C5	@@TIME	8136
@@VDCTL	80E2	@@VER	84E7	@@VRSEC	8319
@@WEOF	84FC	@@WHERE	7FD7	@@WRITE	8511
@@WRSEC	8343	@@WRSSC	8358	@@WTRK	836D

2400 is the transfer address
 00000 Total errors

NOTES:

Command: FORMS

Library: SYS8/SYS

ISAM # : B1H

```

00100 ;LBFORMS/ASM - Set Line Printer Values
0000 00110 TITLE <FORMS - LS-DOS 6.2>
00120 ;
00130 ;
002C 00140 PAR_ERR EQU 44 ;Parameter Error Code
0007 00150 FLAGBT EQU 7 ;Flag byte offset
0007 00160 ADDLF EQU 7 ;Add Line Feed = Bit 0
0007 00170 FFHARD EQU 7 ;Form Feed Hard = Bit 1
0007 00180 TABV EQU 7 ;Tab Expansion = Bit 2
0008 00190 CHARS EQU 8 ;Characters per line
0006 00200 INDENT EQU 6 ;Indent after wrap-around
0002 00210 LINES EQU 2 ;Maximum Lines to Print
0009 00220 MARGIN EQU 9 ;Left hand margin value
0000 00230 PAGE EQU 0 ;Maximum Lines per page
0004 00240 XLATEF EQU 4 ;Xlate From
0005 00250 XLATET EQU 5 ;Xlate To
00260 ;
0042 00270 PDEF EQU 66 ;Page Default = 66
0042 00280 LDEF EQU 66 ;Line Default = 66
00290 ;
000E 00300 CURON EQU 0EH ;Cursor on
000F 00310 CUROFF EQU 0FH ;Cursor off
00DD 00320 SKIP EQU 0DDH ;Skip 3 byte instruction
00330 ;
0000 00340 *GET SVCMAC:3 ;SVC Macro equivalents
00010 ;SVCMAC/ASM - LS-DOS Version VI
00020 *LIST OFF
00300 *LIST ON
0000 00350 *GET VALUES:3 ;Misc. equates
00320 ;VALUES/ASM - Version 6
00330 *LIST OFF
00420 *LIST ON
00360 ;
2400 00370 ORG 2400H
00380 ;
00390 START
2400 ED732324 00400 LD (SAVEBP+1),SP ;Save SP loc
2404 CD2924 00410 CALL FORMS ;Execute Form Code
2407 210000 00420 EXIT LD HL,0 ;Set no error
240A 1816 00430 JR SAVEBP ;Exit
00440 ;
00450 ; I/O Error Handling
00460 ;
240C 3E2C 00470 PRMERR LD A,PAR_ERR ;Parameter Error
240E 6F 00480 IOERR LD L,A ;Xfer error # to HL
240F 2600 00490 LD H,0
2411 F6C0 00500 OR 0C0H ;Abbrev, return
2413 4F 00510 LD C,A ;Xfer to C
2414 00520 @@ERROR ;Display error
2414 3E1A 00001 LD A,26
2416 EF 00002 RST 40
2417 1809 00530 JR SAVEBP ;Go to exit routine
00540 ;
00550 ; Internal Error Message Handling
00560 ;
2419 217928 00570 NOPF LD HL,NOPF$ ;No filter present
241C 00580 @@LOGOT ;Log Message
00003 IFEQ 00H,1
00004 LD HL,
00005 ENDF
241C 3E0C 00006 LD A,12

```

```

241E EF      00007      RST      40
241F 21FFFF  00590 ABORT  LD      HL,-1      ;Set abort code
2422 310000  00600 SAVESP LD      SP,$-$      ;P/u original SP
2425         00610      @CKBRKC      ;Clear any <BREAK>
2425 3E6A    00008      LD      A,106
2427 EF      00009      RST      40
2428 C9      00620      RET              ;And RETurn to DOS
          00630 ;
          00640 ;      FORMS - Process the Forms Filter Parameters
          00650 ;
          00660 FORMS
2429 CD0027  00670      CALL     DOINIT      ;Do initialization
          00680 ;
          00690 ;      Ignore Leading Spaces
          00700 ;
242C 2B      00710      DEC      HL
242D 23      00720 IGSPCS INC      HL      ;Bump cmdline ptr
242E 7E      00730      LD      A,(HL)      ;Skip leading spaces
242F FE20    00740      CP      ' '
2431 28FA    00750      JR      Z,IGSPCS
          00760 ;
          00770 ;      Any Parameters Entered ?
          00780 ;
2433 FE0E    00790      CP      CR+1      ;End of line ?
2435 3007    00800      JR      NC,GETPRM    ;Go if not
          00810 ;
          00820 ;      Display current parameter settings
          00830 ;
2437 CD8924  00840 DISPFM CALL     DSFORMS      ;Create default string
243A CDF726  00850      CALL     DSPLY      ;Display defaults
243D C9      00860      RET              ; and RETurn
          00870 ;
          00880 ;      Display "Paramter Error" if Illegal input
          00890 ;
243E 116227  00900 GETPRM LD      DE,PRMTBL$      ;Any Paramters ?
2441         00910      @PARAM
2441 3E11    00010      LD      A,17
2443 EF      00011      RST      40
2444 C20C24  00920      JP      NZ,PRMERR      ;NZ - "Parameter Error"
          00930 ;
          00940 ;      Create Xlate From Data Area
          00950 ;
2447 3ABA27  00960      LD      A,(XTRESP)      ;P/u xlate TO response
244A 32DF27  00970      LD      (XFRESP),A      ;Xfer to FROM response
244D 21D627  00980      LD      HL,XTPARAM+1      ;HL => XLATE To
2450 7E      00990      LD      A,(HL)      ;P/u value
2451 3600    01000      LD      (HL),0      ;Set Xlate To msb = 0
2453 32E227  01010      LD      (XFARM),A      ;Xfer to From parm
          01020 ;
          01030 ;      Over-ride all other parms if <D>efault
          01040 ;
2456 3AC527  01050      LD      A,(DRESP)      ;<D>efault Parm entered ?
2459 B7      01060      OR      A
245A 2817    01070      JR      Z,CHECKQ      ;No - check for <Q>uery
          01080 ;
          01090 ;      Overwrite $FF data area with default values
          01100 ;
245C ED5BD125 01110      LD      DE,(DATAAREA+2) ;DE => Data area start
2460 010A00  01120      LD      BC,10      ;BC = 10 bytes in table
2463 215827  01130      LD      HL,DEFTAB      ;HL => Default Table
2466 E5      01140      PUSH     HL      ;Save regs

```

```

2467 C5      01150      PUSH      BC
2468 EDB0    01160      LDIR                      ;Xfer to $FF data area
246A C1      01170      POP        BC
246B E1      01180      POP        HL
246C 11E329  01190      LD          DE,DUPDA
246F EDB0    01200      LDIR                      ;Xfer to duplicate DA
2471 18C4    01210      JR          DISPFRM      ;Display forms & exit
          01220 ;
          01230 ;      Prompt for any parms not entered & stuff
          01240 ;
2473          01250 CHECKQ EQU      $
2473 CD1926  01260      CALL      INITVAL      ;Init parm values = def's
2476 CD8924  01270      CALL      DSFORMS      ;Create string
2479 3AAA27  01280      LD          A,(QRESP)      ;<Q>uery parm used?
247C B7      01290      OR          A
247D CCFC24  01300      CALL      Z,CKCOMM      ;Check cmdline values if not
2480 C42A25  01310      CALL      NZ,PROMPT      ;Prompt if "Q"
2483 CDCC25  01320 STUFFIN CALL      STFPRMS      ;Stuff parms in $FF data
2486 C30724  01330      JP          EXIT          ;Good exit
          01340 ;
          01350 ;      Display Current FORMS value settings
          01360 ;
2489 DDCB0746 01370 DSFORMS BIT      0,(IX+ADDLF) ;Add line feed ?
248D 11E128  01380      LD          DE,SADDLF      ;DE => addlf dsply msg
2490 C4E826  01390      CALL      NZ,XFERON      ;Put "ON" in message
          01400 ;
          01410 ;      Display "OFF" if zero, or value if <> zero
          01420 ;
2493 DD7E08  01430      LD          A,(IX+CHARS)      ;CHARS value if it wasn't
2496 B7      01440      OR          A              ;OFF ?
2497 11BA28  01450      LD          DE,SCHARS      ;DE => Chars msg
249A C42127  01460      CALL      NZ,HEXDEC      ;Convert value to dec ASCII
249D 2006    01470      JR          NZ,DOFFHRD      ;Go if Char parm used
          01480 ;
249F 217628  01490 CHAROFF LD          HL,OFFSTR      ; else xfer "OFF" into
24A2 CDEB26  01500      CALL      XFER          ; Chars message
          01510 ;
          01520 ;      FFHARD specified ?
          01530 ;
24A5 DDCB074E 01540 DOFFHRD BIT      1,(IX+FFHARD) ;FFHARD parm used?
24A9 11EE28  01550      LD          DE,SFFHARD      ;DE => Ffhard msg
24AC C4E826  01560      CALL      NZ,XFERON      ;Xfer "ON" if set
          01570 ;
          01580 ;      Xfer INDENT value into string
          01590 ;
24AF DD7E06  01600      LD          A,(IX+INDENT)      ;Default value
24B2 11D428  01610      LD          DE,SINDENT      ;DE=> Indent msg
24B5 CD2127  01620      CALL      HEXDEC      ;Convert to decimal ASCII
          01630 ;
          01640 ;      Xfer LINES value into string
          01650 ;
24B8 DD7E02  01660      LD          A,(IX+LINES)      ;P/u LINES value
24BB 11AD28  01670      LD          DE,SLINES      ;Pt to Lines msg
24BE CD2127  01680      CALL      HEXDEC      ;Convert to decimal ASCII
          01690 ;
          01700 ;      Xfer MARGIN value into string
          01710 ;
24C1 DD7E09  01720      LD          A,(IX+MARGIN)      ;P/u MARGIN value
24C4 11C728  01730      LD          DE,SMARGIN      ;DE => Margin msg
24C7 CD2127  01740      CALL      HEXDEC      ;Convert to decimal ASCII
          01750 ;

```

```

01760 ;      Xfer PAGE value into string
01770 ;
24CA DD7E00 01780 LD      A,(IX+PAGE)      ;P/u page value
24CD 11A028 01790 LD      DE,SPAGE      ;DE => Page msg
24D0 CD2127 01800 CALL     HEXDEC      ;Convert to decimal ASCII
01810 ;
01820 ;      Xfer "ON" into string if Tab set
01830 ;
24D3 DDCB0756 01840 BIT      2,(IX+TABV)      ;Check Tab bit
24D7 11FB28 01850 LD      DE,STAB      ;DE => Tab msg
24DA C4E826 01860 CALL     NZ,XFERON      ;Xfer "ON" if set
01870 ;
01880 ;      Is Xlate FROM = Xlate TO ?
01890 ;
24DD DD7E04 01900 LD      A,(IX+XLATEF)      ;P/u FROM byte
24E0 DD4605 01910 LD      B,(IX+XLATET)      ;P/u TO byte
24E3 B8      01920 CP      B      ;Same ?
24E4 2812    01930 JR      Z,NOSHOW      ;Yes - no show
01940 ;
01950 ;      Two distinct values - convert to Hex
01960 ;
24E6 21FF28 01970 LD      HL,DOXLATE      ;Change CR to LF
24E9 360A    01980 LD      (HL),LF      ; so msg will dsply
01990 ;
24EB 210B29 02000 LD      HL,SXLFROM      ;"From" message
24EE CD3C27 02010 CALL     HEX8      ;Convert A to Hex @ HL
24F1 78      02020 LD      A,B      ;P/u TO
24F2 211429 02030 LD      HL,SXLTO      ;"To" message
24F5 CD3C27 02040 CALL     HEX8      ;Convert A to Hex @ HL
02050 ;
02060 ;      Point HL to string & RETurn
02070 ;
24F8 219728 02080 NOSHOW LD      HL,VALUES      ;HL => Default val string
24FB C9      02090 RET      ;RETurn
02100 ;
02110 ;
02120 ;      CKCOMM - Check command line parameter values
02130 ;
02140 ;
24FC 1E0A    02150 CKCOMM LD      E,10      ;10 values to check
24FE FD210E28 02160 LD      IY,STRTAB      ;IY => Response table
02170 ;
2502 FD6E01 02180 CKCOMML LD      L,(IY+1)      ;P/u address of response
2505 FD6602 02190 LD      H,(IY+2)
02200 ;
02210 ;      Set BC = Parameter Response
02220 ;
2508 7E      02230 LD      A,(HL)      ;Was anything entered ?
2509 B7      02240 OR      A
250A 23      02250 INC     HL      ;Parm addr follows resp
250B 4E      02260 LD      C,(HL)      ;Set HL = (HL)
250C 23      02270 INC     HL
250D 66      02280 LD      H,(HL)
250E 69      02290 LD      L,C
250F 4E      02300 LD      C,(HL)      ;P/u response value
2510 23      02310 INC     HL      ; into BC
2511 46      02320 LD      B,(HL)
02330 ;
02340 ;      Call routine to Range check parm entry
02350 ;
2512 FD6E05 02360 LD      L,(IY+5)      ;P/u address of routine

```


The Source	LIBRARY Files	FORMS - LS-DOS 6.2	Page 00005
2515 FD6606	02370	LD H,(IY+6) ; to check value validity.	
2518 221C25	02380	LD (CALLINS+1),HL ;Stuff addr to CALL instr	
251B C40000	02390 CALLINS	CALL NZ,\$-\$;BC = response, A = type	
251E C20C24	02400	JP NZ,PMERR ;NZ - "Parameter Error"	
	02410 ;		
	02420 ;	Position to next table entry	
	02430 ;		
2521 010900	02440	LD BC,9 ;Pos to next STRTAB entry	
2524 FD09	02450	ADD IY,BC	
2526 1D	02460	DEC E ;Done ?	
2527 20D9	02470	JR NZ,CKCOMML	
2529 C9	02480	RET ;Yes - RETurn	
	02490 ;		
	02500 ;		
	02510 ;	PROMPT - for any vals not entered in parm line	
	02520 ;		
252A 060A	02530 PROMPT	LD B,10 ;Eight normal + 2 Xlates	
252C FD210E28	02540	LD IY,STRTAB ;Prompt, response table	
	02550 ;		
	02560 ;	P/u type byte from table & set length = 1	
	02570 ;		
2530 FD7E00	02580 PROMPTL	LD A,(IY) ;P/u type byte	
2533 3C	02590	INC A ;Merge length = 1	
2534 326928	02600	LD (FAKETAB+1),A ;Store new type byte	
	02610 ;		
	02620 ;	P/u address of response byte	
	02630 ;		
2537 FD5E01	02640 REINPUT	LD E,(IY+1) ;P/u address	
253A FD5602	02650	LD D,(IY+2) ; in DE	
	02660 ;		
	02670 ;	Pick up Prompt string address & display it	
	02680 ;		
253D FD6E03	02690 DOPRMPT	LD L,(IY+3) ;P/u address in HL	
2540 FD6604	02700	LD H,(IY+4)	
2543 CD5825	02710	CALL DISPROM ;P/u default & display	
	02720 ;		
	02730 ;	Input response & stuff into Parm table	
	02740 ;		
2546 CD4327	02750	CALL INPUT ;Input value	
2549 C5	02760	PUSH BC ;Save count	
254A C4A325	02770	CALL NZ,STUFVAL ;Stuff in valid input	
254D C1	02780	POP BC ;Restore count	
254E 20E7	02790	JR NZ,REINPUT ;Re-input if bad value	
	02800 ;		
	02810 ;	Position to next table entry	
	02820 ;		
2550 110900	02830 NEXTPR	LD DE,9 ;9 bytes per entry	
2553 FD19	02840	ADD IY,DE	
2555 10D9	02850	DJNZ PROMPTL ;B prompts	
2557 C9	02860	RET ;Done	
	02870 ;		
	02880 ;		
	02890 ;	DISPROM - Display Prompt	
	02900 ;		
2558 D5	02910 DISPROM	PUSH DE ;Save regs	
2559 C5	02920	PUSH BC	
255A 0E0F	02930	LD C,CUROFF ;Turn off cursor	
255C CDF126	02940	CALL DSP	
255F 0620	02950	LD B,32 ;Space padding base	
	02960 ;		
2561 4E	02970 PRLP	LD C,(HL) ;P/u character	

```

2562 23      02980      INC      HL      ;Pos to next
2563 05      02990      DEC      B       ;Dec count
2564 CDF126  03000      CALL     DSP      ;Output byte
2567 79      03010      LD       A,C     ;P/u char
2568 FE7B    03020      CP        '{'    ;Bracket ?
256A 20F5    03030      JR       NZ,PRLP ;No - go til bracket
256C CD8225  03040      CALL     STUFDEF  ;Display default
256F 78      03050      LD       A,B     ;P/u base #
2570 81      03060      ADD      A,C     ; & calculate # of
2571 47      03070      LD       B,A     ;Spaces to print
2572 0E20    03080      LD       C,' '
          03090 ;
2574 CDF126  03100 SPLP   CALL     DSP      ;Output spaces
2577 10FB    03110      DJNZ     SPLP
          03120 ;
2579 219328  03130      LD       HL,ENDPROM ;End of prompt
257C CDF726  03140      CALL     DSPLY
257F C1      03150      POP      BC      ;Recover regs
2580 D1      03160      POP      DE
2581 C9      03170      RET           ; and RETURN
          03180 ;
          03190 ;
          03200 ;      STUFDEF - Stuff default value in prompt
          03210 ;
2582 FD6E07  03220 STUFDEF LD      L,(IY+7) ;P/u default string
2585 FD6608  03230      LD      H,(IY+8) ; address
2588 0E05    03240      LD      C,5     ;5 chars max
258A 7E      03250 PNLP   LD      A,(HL)
258B 23      03260      INC      HL     ;Bump source
258C FE0A    03270      CP        LF     ;Done ?
258E 280A    03280      JR       Z,DUNLP
2590 FE20    03290      CP        ' '    ;Leading space ?
2592 28F6    03300      JR       Z,PNLP  ;Yes - ignore it
2594 CD9C25  03310      CALL     DISPA   ;Output A
          03320 ;
2597 0D      03330 PNLP2  DEC      C       ;Dec count
2598 20F0    03340      JR       NZ,PNLP
          03350 ;
259A 3E7D    03360 DUNLP  LD      A,'}'    ;Output end bracket
259C C5      03370 DISPA  PUSH     BC      ;Save count in C
259D 4F      03380      LD      C,A     ;Xfer char to C
259E CDF126  03390      CALL     DSP      ;Output byte
25A1 C1      03400      POP      BC      ;Recover C
25A2 C9      03410      RET           ; and RETURN
          03420 ;
          03430 ;
          03440 ;      STUFVAL - Stuff values into Parm Table
          03450 ;
25A3 D5      03460 STUFVAL PUSH     DE      ;DE => Response Byte
25A4 21FF27  03470      LD      HL,FAKEPRM ;HL => Fake Parm Entry
25A7 116828  03480      LD      DE,FAKETAB ;DE => Fake Parm Table
25AA         03490      @PARAM      ;Parse entry
25AA 3E11    00012      LD      A,17
25AC EF      00013      RST      40
25AD E1      03500      POP      HL     ;HL => Response
25AE C0      03510      RET      NZ     ;NZ - Re-input
          03520 ;
          03530 ;      Stuff response into Parameter Table
          03540 ;
25AF E5      03550      PUSH     HL     ;Save response dest
25B0 3A6B28  03560      LD      A,(FAKERES) ;P/u response

```

```

25B3 010000 03570 VALUE LD BC,$-$ ;P/u value
25B6 23 03580 INC HL ;HL => Parm Address
25B7 5E 03590 LD E,(HL) ;P/u parm address
25B8 23 03600 INC HL
25B9 56 03610 LD D,(HL)
25BA EB 03620 EX DE,HL ;HL => Parm lsb
25BB 71 03630 LD (HL),C ;Stuff response in table
03640 ;
03650 ; CALL range checking routine
03660 ;
25BC 21C725 03670 LD HL,RETADR ;Put RET addr on stack
25BF E5 03680 PUSH HL
25C0 FD6E05 03690 LD L,(IY+5) ;P/u addr of range
25C3 FD6606 03700 LD H,(IY+6) ;Checking in HL
25C6 E9 03710 JP (HL) ;Routine sets Z for stat
25C7 E1 03720 RETADR POP HL ;HL => Response byte
25C8 C0 03730 RET NZ ;Don't change if NZ
25C9 3680 03740 LD (HL),80H ; else stuff non-zero
25CB C9 03750 RET ; value for response
03760 ;
03770 ;
03780 ; STFPRMS - Stuff Numeric & Flag Parms into $FF
03790 ;
03800 ;
03810 ; Pt HL => Response byte addr & offset Table
03820 ;
25CC 21E427 03830 STFPRMS LD HL,RESPTAB ;HL => Response Table
25CF DD210000 03840 DATAREA LD IX,$-$ ;P/u Data Area pointer
25D3 0607 03850 LD B,7 ;7 numeric values
03860 ;
03870 ; P/u response byte & offset byte to $FF data
03880 ;
25D5 5E 03890 STUFLP LD E,(HL) ;P/u response address
25D6 23 03900 INC HL
25D7 56 03910 LD D,(HL)
25D8 23 03920 INC HL ;HL => $FF data offset
25D9 4E 03930 LD C,(HL) ;P/u offset in data area
25DA 23 03940 INC HL
03950 ;
25DB 1A 03960 LD A,(DE) ;P/u response
25DC B7 03970 OR A ;Parm entered ?
03980 ;
03990 ; Parm entered - calculate Parm's Location
04000 ;
25DD 13 04010 INC DE ;DE => Parameter Dest
25DE EB 04020 EX DE,HL ;Xfer to HL
25DF 7E 04030 LD A,(HL) ;Set HL = (HL)
25E0 23 04040 INC HL
25E1 66 04050 LD H,(HL)
25E2 6F 04060 LD L,A
04070 ;
04080 ; Stuff parm response into $FF data region
04090 ;
25E3 79 04100 LD A,C ;Xfer offset to A
25E4 4E 04110 LD C,(HL) ;P/u lsb of Parm response
25E5 EB 04120 EX DE,HL ;Recover HL (Table ptr)
25E6 2806 04130 NOPOUT JR Z,NOPARM ;No - don't stuff
25E8 32ED25 04140 LD (IXINST+2),A ;Modify offset in IX inst
25EB DD7100 04150 IXINST LD (IX+$-$),C ;Xfer parm resp to $FF
04160 ;
25EE 10E5 04170 NOPARM DJNZ STUFLP ;Next entry

```

```

04180 ;
04190 ;      Set Flag bits in $FF data area if parms set
04200 ;
25F0 0603 04210 GETFLAG LD      B,3          ;3 flag values
25F2 5E    04220 FLOOP  LD      E,(HL)       ;P/u response address
25F3 23    04230        INC      HL
25F4 56    04240        LD      D,(HL)
25F5 23    04250        INC      HL
25F6 1A    04260        LD      A,(DE)       ;Entered ?
25F7 B7    04270        OR      A
25F8 281C  04280        JR      Z,NEXTFLG    ;No - get next one
04290 ;
04300 ;      Response - If true (SET), False (RES)
04310 ;
25FA 13    04320        INC      DE          ;Pos to parm address
25FB EB    04330        EX       DE,HL       ;P/u Parm
25FC 7E    04340        LD      A,(HL)       ;Set HL = (HL)
25FD 23    04350        INC      HL
25FE 66    04360        LD      H,(HL)
25FF 6F    04370        LD      L,A
2600 EB    04380        EX       DE,HL       ;Put into DE
2601 0E86  04390        LD      C,10000110B  ;Default = Reset bit inst
2603 1A    04400        LD      A,(DE)       ;P/u lsb of parm
2604 B7    04410        OR      A           ;Set ?
2605 2802  04420        JR      Z,SKIPSET    ;No - skip SET inst
2607 CBF1  04430        SET     6,C          ;Change to Set bit inst
04440 ;
04450 ;      Create Post opcode for IX instruction
04460 ;
2609 78    04470 SKIPSET LD      A,B          ;P/u bit # (0-2)
260A 3D    04480        DEC      A
260B 07    04490        RLCA
260C 07    04500        RLCA
260D 07    04510        RLCA
260E B1    04520        OR      C           ;Post op code
260F 321526 04530        LD      (IXINST2+3),A ;Change RES b,(IX+nn) ins
2612 DDCB0786 04540 IXINST2 RES  $-$(IX+FLAGBT) ;Set/Reset bit B in $FF
2616 10DA   04550 NEXTFLG DJNZ   FLOOP      ;Get next flag
2618 C9     04560        RET              ;Done - RETURN
04570 ;
04580 ;
04590 ;      INITVAL - Initial Parm values
04600 ;
04610 ;
2619 0605  04620 INITVAL LD      B,5          ;5 values to stuff
261B 21E427 04630        LD      HL,RESPTAB   ;HL => Response & offsets
04640 ;
261E 5E    04650 SDLP   LD      E,(HL)       ;P/u response byte addr
261F 23    04660        INC      HL
2620 56    04670        LD      D,(HL)
2621 23    04680        INC      HL
2622 1A    04690        LD      A,(DE)       ;P/u response byte
04700 ;
04710 ;      Get parm table address - DE = (DE)
04720 ;
2623 EB    04730        EX       DE,HL
2624 23    04740        INC      HL          ;Parm address after resp
2625 4E    04750        LD      C,(HL)       ;P/u lsb
2626 23    04760        INC      HL
2627 66    04770        LD      H,(HL)       ;P/u msb
2628 69    04780        LD      L,C          ;HL = (HL)

```

```

2629 EB      04790      EX      DE,HL      ;Get back to DE
            04800 ;
            04810 ;      P/u default value from $FF data area
            04820 ;

262A D5      04830      PUSH     DE      ;Save HL & DE
262B E5      04840      PUSH     HL
262C 5E      04850      LD       E,(HL)      ;P/u offset
262D 1600    04860      LD       D,0      ;DE = offset to default
262F 2AD125  04870      LD       HL,(DATAAREA+2) ;HL => Data Area
2632 19      04880      ADD      HL,DE
2633 4E      04890      LD       C,(HL)      ;P/u default value
2634 E1      04900      POP      HL      ;Restore regs
2635 D1      04910      POP      DE
            04920 ;
            04930 ;      If parm wasn't entered - stuff default value
            04940 ;

2636 23      04950      INC      HL      ;Posn to next entry
2637 B7      04960      OR       A      ;Parm entered ?
2638 2806    04970      JR       Z,STFDEF      ;No - stuff default
263A 3AAA27  04980      LD       A,(QRESP)      ;<Q>uery parm used?
263D B7      04990      OR       A
263E 2802    05000      JR       Z,PRMENT      ;No - don't stuff
2640 79      05010 STFDEF LD       A,C      ;Yes - stuff default
2641 12      05020      LD       (DE),A
2642 10DA    05030 PRMENT DJNZ     SDLP
2644 C9      05040      RET
            05050 ;
            05060 ;
            05070 ;      Range Checking Code of Values
            05080 ;
            05090 ;
            05100 ;      Is the Page length valid ?
            05110 ;

2645 CDD126  05120 RPAGE CALL     MORE0?      ;Number between 1 - 255 ?
2648 C0      05130      RET      NZ      ;No - NZ
2649 3ACF27  05140      LD       A,(LPARM)      ;P/u LINES value
264C 3D      05150      DEC      A
264D B9      05160      CP       C      ;LINES > PAGE ?
264E F5      05170      PUSH     AF      ;Save status
264F 3AAA27  05180      LD       A,(QRESP)      ;<Q>uery parm used?
2652 B7      05190      OR       A
2653 2003    05200      JR       NZ,PQUERY      ;Go if so
2655 F1      05210      POP      AF      ;No
2656 1872    05220      JR       VALID2?      ;Return NZ if L>P
2658 F1      05230 PQUERY POP      AF      ;L > P ?
2659 380D    05240      JR       C,SETZ      ;No - Set Z flag
265B 79      05250      LD       A,C      ;Yes - Set LINES = PAGE
265C 32E529  05260      LD       (DUPDA+LINES),A
265F 328F27  05270      LD       (LRESP),A      ;Pretend that LINES was
2662 32CF27  05280      LD       (LPARM),A      ; responded to
2665 CD8924  05290      CALL     DSFORMS      ;Reset defaults
2668 BF      05300 SETZ   CP       A      ;Set Z flag
2669 C9      05310      RET
            05320 ;
            05330 ;      Is the lines printed per page valid ?
            05340 ;

266A CDD126  05350 RLINES CALL     MORE0?      ;Number between 1 - 255 ?
266D C0      05360      RET      NZ      ;No - NZ
266E 3D      05370      DEC      A
266F 21D327  05380      LD       HL,PPARM      ;HL => Page length
2672 1855    05390      JR       VALID1?      ;Set status accordingly

```

```

05400 ;
05410 ;      Is the Characters printed per line valid ?
05420 ;
2674 CB77      05430 RCHARS BIT      6,A      ;Flag response ?
2676 20F0      05440 JR          NZ,SETZ      ;Yes - Set Z
2678 CDD126    05450 CALL     MORE0?      ;No - More than zero ?
267B C0        05460 RET          NZ          ;No - NZ
267C 3AAA27    05470 LD          A,(QRESP)    ;<Q>uery parm used?
267F B7        05480 OR          A          ;
2680 C8        05490 RET          Z          ;Return if not
05500 ;
05510 ;      <Q>uery - Make sure CHARS > INDENT+MARGIN
05520 ;
2681 21D127    05530 LD          HL,MPARM      ;HL => Margin value
2684 3ACD27    05540 LD          A,(IPARM)    ;A = Indent value
2687 86        05550 ADD          A,(HL)      ;A = Indent + Margin
2688 B9        05560 CP          C          ;Less than CHARS ?
2689 38DD      05570 JR          C,SETZ      ;Yes - Set Z
268B AF        05580 XOR          A          ;Reset INDENT & MARGIN=0
268C 32E929    05590 LD          (DUPDA+INDENT),A
268F 32EC29    05600 LD          (DUPDA+MARGIN),A
2692 32CD27    05610 LD          (IPARM),A
2695 32D127    05620 LD          (MPARM),A
2698 3C        05630 INC          A          ;Pretend that INDENT &
2699 329927    05640 LD          (MRESP),A    ; MARGIN were responded
269C 328627    05650 CHNGIND LD      (IRESP),A ; to
269F CD8924    05660 CALL     DSFORMS      ;Change defaults
26A2 AF        05670 XOR          A          ;Set Z & RETURN
26A3 C9        05680 RET
05690 ;
05700 ;      Is Margin less than Characters/Line ?
05710 ;
26A4 CDDA26    05720 RMARGIN CALL    NUMERIC      ;Number between 0 - 255 ?
26A7 C0        05730 RET          NZ          ;No - NZ
26A8 CDC626    05740 CALL     VALID?      ;Yes - less than CHARS ?
26AB C0        05750 RET          NZ          ;No - RETURN NZ
26AC 3ACD27    05760 LD          A,(IPARM)    ;P/u INDENT
26AF 81        05770 ADD          A,C          ;Add to MARGIN
26B0 CDC626    05780 CALL     VALID?      ;M + I < CHARS ?
26B3 C8        05790 RET          Z          ;Yes - RETURN Z
26B4 AF        05800 XOR          A          ;No - Set INDENT default
26B5 32CD27    05810 LD          (IPARM),A    ;Equal to Zero
26B8 32E929    05820 LD          (DUPDA+INDENT),A
26BB 3C        05830 INC          A          ;Pretend I was responded
26BC 18DE      05840 JR          CHNGIND      ; to
05850 ;
05860 ;      Is Margin + Indent less than chars/line ?
05870 ;
26BE CDDA26    05880 RINDENT CALL    NUMERIC      ;Number between 0 - 255 ?
26C1 C0        05890 RET          NZ          ;No - NZ
26C2 3AD127    05900 LD          A,(MPARM)    ;P/u MARGIN val
26C5 81        05910 ADD          A,C          ;A = MARGIN + INDENT
26C6 21CB27    05920 VALID? LD      HL,CPARM      ;HL => Characters/Line
26C9 BE        05930 VALID1? CP      (HL)        ;Response > (HL) ?
26CA 3002      05940 VALID2? JR      NC,SETNZ      ;Yes - Reset Z flag
26CC BF        05950 CP          A          ;No - Set Z flag
26CD C9        05960 RET
26CE AF        05970 SETNZ      XOR          A          ;Reset Z flag
26CF 3C        05980 INC          A
26D0 C9        05990 RET
06000 ;

```

```

06010 ;      Is the response a number between 1-255 ?
06020 ;
26D1 CDDA26 06030 MORE0? CALL    NUMERIC      ;Is the response a number
26D4 C0      06040      RET     NZ           ;Between 0 - 255 ?
26D5 B7      06050      OR      A           ;Is the response zero ?
26D6 28F6    06060      JR      Z,SETNZ      ;Yes - reset Z flag
26D8 BF      06070      CP      A           ;No - set Z flag
26D9 C9      06080      RET
06090 ;
06100 ;      Is the response a 1 byte number ?
06110 ;
26DA E680    06120 NUMERIC AND     80H       ;Bit 7 is set if the
26DC EE80    06130      XOR     80H       ;Response is numeric.
26DE C0      06140      RET     NZ         ;NZ <= if Bit is reset
26DF 04      06150      INC     B          ;Is the response only
26E0 05      06160      DEC     B          ;1 byte (msb = 0) ?
26E1 79      06170      LD      A,C        ;Set A = response
26E2 C9      06180      RET              ;Yes (Z), no (NZ)
06190 ;
06200 ;      Is the response a flag (ON/YES, OFF/NO) ?
06210 ;
26E3 E640    06220 FLAG?  AND     40H       ;Bit 6 is set if the
26E5 EE40    06230      XOR     40H       ;Response is a flag.
26E7 C9      06240      RET              ;Yes (Z), no (NZ)
06250 ;
06260 ;
06270 ;      XFER - Xfer string @ HL to DE
06280 ;      XFERON - Xfer "ON" string to DE
06290 ;
06300 ;
26E8 217328 06310 XFERON LD      HL,ONSTR    ;HL => "ON"
26EB 010300 06320 XFER  LD      BC,3        ;3 chars to xfer
26EE EDB0    06330      LDIR
26F0 C9      06340      RET
06350 ;
06360 ;
06370 ;      DSP - Display a byte
06380 ;
06390 ;
26F1 D5      06400 DSP    PUSH    DE          ;Save DE
26F2         06410      @@DSP      ;Output byte
26F2 3E02    06414      LD      A,2
26F4 EF      06415      RST     40
26F5 1804    06420      JR      EXDSP
06430 ;
06440 ;
06450 ;      DSPLY - Display a string
06460 ;
06470 ;
26F7 D5      06480 DSPLY  PUSH    DE          ;Save DE
26F8         06490      @@DSPLY  ;Display it
06490      06490      IFEQ     00H,1
06490      06490      LD      HL,
06490      06490      ENDIF
26F8 3E0A    06490      LD      A,10
26FA EF      06490      RST     40
26FB D1      06500 EXDSP  POP     DE
26FC C8      06510      RET     Z          ;Return if good
26FD C30E24 06520      JP      IOERR       ;NZ - I/O Error
06530 ;
06540 ;

```

```

06550 ; DOINIT - Sign on message & Get Data area
06560 ;
06570 ;
2700 E5 06580 DOINIT PUSH HL ;Save command ptr
2701 06590 @@FLAGS ;Get system flags
2701 3E65 00021 LD A,101
2703 EF 00022 RST 40
06600 ;
06610 ; Point IX to Filter Data area
06620 ;
2704 116F28 06630 LD DE,$FF ;DE => "$FF"
2707 06640 @@GTMOD ;Find start
2707 3E53 00023 LD A,83
2709 EF 00024 RST 40
270A C21924 06650 JP NZ,NOPF ;Abort if Forms/Flt missing
06660 ;
270D EB 06670 EX DE,HL ;HL => Data Area
270E 010400 06680 LD BC,4 ;Add 4 to ptr
2711 09 06690 ADD HL,BC
2712 22D125 06700 LD (DATAAREA+2),HL ;Save $FF data pointer
2715 11E329 06710 LD DE,DUPDA ;DE => Duplicate D area
2718 D5 06720 PUSH DE ;Save ptr
2719 0E0A 06730 LD C,10 ;BC = 10 bytes to xfer
271B EDB0 06740 LDIR
271D DDE1 06750 POP IX ;IX pts to data area
271F E1 06760 POP HL ;Recover cmdline ptr
2720 C9 06770 RET ; and RETURN
06780 ;
06790 ;
06800 ;
06810 ; HEXDEC - Convert Hex Number to Decimal ASCII
06820 ; A => 8-bit Hex Number to Convert
06830 ; DE => Destination of ASCII characters
06840 ;
06850 ;
2721 C5 06860 HEXDEC PUSH BC ;Save regs
2722 E5 06870 PUSH HL
2723 F5 06880 PUSH AF
06890 ;
06900 ; Transfer ASCII chars into temporary buffer
06910 ;
2724 D5 06920 PUSH DE ;Save real destination
2725 11DE29 06930 LD DE,TEMBUF ;DE => Temporary buffer
2728 2600 06940 LD H,0 ;Xfer # to HL
272A 6F 06950 LD L,A
272B 06960 @@HEXDEC ;Convert to ASCII
272B 3E61 00025 LD A,97
272D EF 00026 RST 40
272E 1B 06970 DEC DE ;Pos to 3-byte field
272F 1B 06980 DEC DE
2730 1B 06990 DEC DE
2731 E1 07000 POP HL ;Recover user buffer
2732 EB 07010 EX DE,HL ;HL to #, DE to user buff
2733 010300 07020 LD BC,3
2736 EDB0 07030 LDIR ;Move the ASCII number
07040 ;
2738 F1 07050 POP AF ;Recover #
2739 E1 07060 POP HL ; and other regs
273A C1 07070 POP BC
273B C9 07080 RET
07090 ;

```



```

07100 ;
07110 ;      HEX8 - Convert HEX Number in A to HEX @ HL
07120 ;
07130 ;
273C C5 07140 HEX8  PUSH    BC      ;Save regs
273D 4F 07150      LD      C,A      ;Xfer char to C
273E    07160      @@HEX8      ;Do it
273E 3E62 00027      LD      A,98
2740 EF 00028      RST      40
2741 C1 07170      POP     BC
2742 C9 07180      RET              ; and RETurn
07190 ;
07200 ;
07210 ;
07220 ;      INPUT - Input a string into INBUFF$
07230 ;
07240 ;
2743 E5 07250 INPUT  PUSH    HL      ;Save regs
2744 D5 07260      PUSH    DE
2745 C5 07270      PUSH    BC
07280 ;
2746 010003 07290      LD      BC,3<8      ;3 chars max
2749 210228 07300      LD      HL,INBUFF$    ;Key input buffer
274C    07310      @@KEYIN    ;Input line
274C 3E09 00029      LD      A,9
274E EF 00030      RST      40
274F DA1F24 07320      JP      C,ABORT      ;Abort if <BREAK>
07330 ;
2752 04 07340      INC     B      ;Set Z flag if
2753 05 07350      DEC     B      ; no input
07360 ;
2754 C1 07370      POP     BC      ;Restore regs
2755 D1 07380      POP     DE
2756 E1 07390      POP     HL
2757 C9 07400      RET              ; & RETurn with condition
07410 ;
07420 ;      Default Value Table
07430 ;
2758 42 07440 DEFTAB DB      PDEF,0,LDEF,0,0,0,0,00000100B,0,0
00 42 00 00 00 00 04 00
00
07450 ;
07460 ;      Parameter table
07470 ;
2762 80 07480 PRMTBL$ DB      80H      ;6.2 @PARAM
07490 ;
07500 ;      ADDLF (A) - Flag Input Only
07510 ;
2763 55 07520      DB      FLAG!ABB!5
2764 41 07530      DB      'ADDLF'
44 44 4C 46
2769 00 07540 ARESP  DB      0
276A D727 07550      DW      APARM
07560 ;
07570 ;      CHARS (C) - Accept Numeric or Flag input
07580 ;
276C D5 07590      DB      FLAG!ABB!NUM!5
276D 43 07600      DB      'CHARS'
48 41 52 53
2772 00 07610 CRESP  DB      0
2773 CB27 07620      DW      CPARAM

```

```

07630 ;
07640 ;      FFHARD (F) - Accept Flag input only
07650 ;
2775 56      07660      DB      FLAG!ABB!6
2776 46      07670      DB      'FFHARD'
      46 48 41 52 44
277C 00      07680 FRESP      DB      0
277D D927    07690      DW      FPARM
      07700 ;
      07710 ;      INDENT (I) - Accept Numeric Input only
      07720 ;
277F 96      07730      DB      NUM!ABB!6
2780 49      07740      DB      'INDENT'
      4E 44 45 4E 54
2786 00      07750 IRESP      DB      0
2787 CD27    07760      DW      IPARM
      07770 ;
      07780 ;      LINES (L) - Accept Numeric Input only
      07790 ;
2789 95      07800      DB      NUM!ABB!5
278A 4C      07810      DB      'LINES'
      49 4E 45 53
278F 00      07820 LRESP      DB      0
2790 CF27    07830      DW      LPARM
      07840 ;
      07850 ;      MARGIN (M) - Accept Numeric Input only
      07860 ;
2792 96      07870      DB      NUM!ABB!6
2793 4D      07880      DB      'MARGIN'
      41 52 47 49 4E
2799 00      07890 MRESP      DB      0
279A D127    07900      DW      MPARM
      07910 ;
      07920 ;      PAGE (P) - Accept Numeric Input only
      07930 ;
279C 94      07940      DB      NUM!ABB!4
279D 50      07950      DB      'PAGE'
      41 47 45
27A1 00      07960 PRES      DB      0
27A2 D327    07970      DW      PPARM
      07980 ;
      07990 ;      QUERY (Q) - Accept Flag Input Only
      08000 ;
27A4 55      08010      DB      FLAG!ABB!5
27A5 51      08020      DB      'QUERY'
      55 45 52 59
27AA 00      08030 QRESP      DB      0
27AB C927    08040      DW      QPARM
      08050 ;
      08060 ;      TAB (T) - Accept Flag input only
      08070 ;
27AD 53      08080      DB      FLAG!ABB!3
27AE 54      08090      DB      'TAB'
      41 42
27B1 00      08100 TRESP      DB      0
27B2 DB27    08110      DW      TPARM
      08120 ;
      08130 ;      XLATE (X) - Accept Numeric input only
      08140 ;
27B4 95      08150      DB      NUM!ABB!5
27B5 58      08160      DB      'XLATE'

```

```

      4C 41 54 45
27BA 00      08170 XTRESP DB      0
27BB D527    08180      DW      XTPARM
           08190 ;
           08200 ;      DEFAULT (D) - Accept Flag input only
           08210 ;
27BD 57      08220      DB      FLAG!ABB!7
27BE 44      08230      DB      'DEFAULT'
      45 46 41 55 4C 54
27C5 00      08240 DRESP DB      0
27C6 DD27    08250      DW      DPARM
           08260 ;
27C8 00      08270      DB      0
           08280 ;
27C9 0000    08290 QPARM DW      0
27CB 0000    08300 CPARM DW      0
27CD 0000    08310 IPARM DW      0
27CF 0000    08320 LPARM DW      0
27D1 0000    08330 MPARM DW      0
27D3 0000    08340 PPARM DW      0
27D5 0000    08350 XTPARM DW      0
           08360 ;
27D7 0000    08370 APARM DW      0
27D9 0000    08380 FPARM DW      0
27DB 0000    08390 TPARM DW      0
27DD 0000    08400 DPARM DW      0
           08410 ;
27DF 00      08420 XFRESP DB      0
27E0 E227    08430      DW      XFPARM
27E2 0000    08440 XFPARM DW      0
           08450 ;
           08460 ;
           08470 ;      Response Table - Response Addr, $FF Offset
           08480 ;
           08490 ;
           08500 ;      8-bit Numeric Responses
           08510 ;
27E4 7227    08520 RESPTAB DW      CRESP
27E6 08      08530      DB      CHARS
           08540 ;
27E7 8627    08550      DW      IRESP
27E9 06      08560      DB      INDENT
           08570 ;
27EA 8F27    08580      DW      LRESP
27EC 02      08590      DB      LINES
           08600 ;
27ED 9927    08610      DW      MRESP
27EF 09      08620      DB      MARGIN
           08630 ;
27F0 A127    08640      DW      PRESP
27F2 00      08650      DB      PAGE
           08660 ;
27F3 BA27    08670      DW      XTRESP
27F5 05      08680      DB      XLATET
           08690 ;
27F6 DF27    08700      DW      XFRESP
27F8 04      08710      DB      XLATEF
           08720 ;
           08730 ;      Flag Response Table
           08740 ;
27F9 B127    08750      DW      TRESP

```

```

27FB 7C27      08760      DW      FRESP
27FD 6927      08770      DW      ARESP
                08780 ;
                08790 ;
27FF 28        08800 FAKEPRM DB      '(F='
   46 3D
000C           08810 INBUFF$ DS      12
                08820 ;
                08830 ;
                08840 ;
                08850 ;
                08860 ;
                08870 ;
                08880 ;
                08890 ;
                08900 ;
                08910 ;
                08920 ;
                08930 ;
280E           08940 STRTAB EQU      $
280E 80        08950      DB      NUM          ;PAGE
280F A127      08960      DW      PRES,PPROMPT,RPAGE,SPAGE
                A229 4526 A028
2817 80        08970      DB      NUM          ;LINES
2818 8F27      08980      DW      LRESP,LPROMPT,RLINES,SLINES
                7A29 6A26 AD28
2820 80        08990      DB      NUM          ;CHARS
2821 7227      09000      DW      CRESP,CPROMPT,RCHARS,SCHARS
                3229 7426 BA28
2829 80        09010      DB      NUM          ;MARGIN
282A 9927      09020      DW      MRESP,MPROMPT,RMARGIN,SMARGIN
                9229 A426 C728
2832 80        09030      DB      NUM          ;INDENT
2833 8627      09040      DW      IRESP,IPROMPT,RINDENT,SINDENT
                6029 BE26 D428
283B 40        09050      DB      FLAG          ;ADDLF
283C 6927      09060      DW      ARESP,APROMPT,FLAG?,SADDLF
                1929 E326 E128
2844 40        09070      DB      FLAG          ;FFHARD
2845 7C27      09080      DW      FRESP,FPROMPT,FLAG?,SFFHARD
                4F29 E326 EE28
284D 40        09090      DB      FLAG          ;TAB
284E B127      09100      DW      TRESP,TPROMPT,FLAG?,STAB
                B829 E326 FB28
2856 80        09110      DB      NUM          ;XLATE From
2857 DF27      09120      DW      XFRESP,XPROMF,NUMERIC,SXLFROM-2
                C729 DA26 0929
285F 80        09130      DB      NUM          ;XLATE To
2860 BA27      09140      DW      XTRESP,XPROMT,NUMERIC,SXLTO-2
                D329 DA26 1229
                09150 ;
                09160 ;
                09170 ;
                Fake Parameter Table for prompts (QUERY)
2868 80        09180 FAKETAB DB      80H          ;6.2 @ PARAM
2869 00        09190      DB      0              ;Type byte
286A 46        09200      DB      'F'
286B 00        09210 FAKERES DB      0
286C B425      09220      DW      VALUE+1          ;Destination
286E 00        09230      DB      0
                09240 ;
                09250 ;

```

```

286F 24      09260 $FF      DB      '$FF',ETX
      46 46 03
2873 20      09270 ONSTR    DB      ' ON'
      4F 4E
2876 4F      09280 OFFSTR   DB      'OFF'
      46 46
      09290 ;
2879 46      09300 NOPF$     DB      'Forms Filter not Resident',CR
      6F 72 6D 73 20 46 69 6C
      74 65 72 20 6E 6F 74 20
      52 65 73 69 64 65 6E 74
      0D
      09310 ;
2893 3F      09320 ENDPROM   DB      '? ',CURON,ETX
      20 0E 03
2897 50      09330 VALUES   DB      'PAGE = '
      41 47 45 20 20 20 3D 20
28A0 20      09340 SPAGE     DB      ' 66',LF,'LINES = '
      36 36 0A 4C 49 4E 45 53
      20 20 3D 20
28AD 20      09350 SLINES    DB      ' 66',LF,'CHARS = '
      36 36 0A 43 48 41 52 53
      20 20 3D 20
28BA 4F      09360 SCHARS    DB      'OFF',LF,'MARGIN = '
      46 46 0A 4D 41 52 47 49
      4E 20 3D 20
28C7 20      09370 SMARGIN   DB      '  ',LF,'INDENT = '
      20 30 0A 49 4E 44 45 4E
      54 20 3D 20
28D4 20      09380 SINDENT   DB      '  ',LF,'ADDLF = '
      20 30 0A 41 44 44 4C 46
      20 20 3D 20
28E1 4F      09390 SADDLF    DB      'OFF',LF,'FFHARD = '
      46 46 0A 46 46 48 41 52
      44 20 3D 20
28EE 4F      09400 SFFHARD   DB      'OFF',LF,'TAB = '
      46 46 0A 54 41 42 20 20
      20 20 3D 20
28FB 4F      09410 STAB      DB      'OFF',LF
      46 46 0A
28FF 0D      09420 DOXLATE   DB      CR,'XLATE = X',AP
      58 4C 41 54 45 20 20 3D
      20 58 27
290B 30      09430 SXLFROM   DB      '00',AP,' => X',AP
      30 27 20 3D 3E 20 58 27
2914 30      09440 SXLTO     DB      '00',AP,LF,CR
      30 27 0A 0D
      09450 ;
      09460 ;
2919 41      09470 APROMPT   DB      'Add Line Feed after C/R {'
      64 64 20 4C 69 6E 65 20
      46 65 65 64 20 61 66 74
      65 72 20 43 2F 52 20 7B
2932 4D      09480 CPROMPT   DB      'Maximum Characters per Line {'
      61 78 69 6D 75 6D 20 43
      68 61 72 61 63 74 65 72
      73 20 70 65 72 20 4C 69
      6E 65 20 7B
294F 52      09490 FPPROMPT  DB      'Real Form Feeds {'
      65 61 6C 20 46 6F 72 6D
      20 46 65 65 64 73 20 7B

```

```

2960 49      09500 IPROMPT DB      'Indent after Wrap-around {'
        6E 64 65 6E 74 20 61 66
        74 65 72 20 57 72 61 70
        2D 61 72 6F 75 6E 64 20
        7B
297A 4C      09510 LPROMPT DB      'Lines Printed per Page {'
        69 6E 65 73 20 50 72 69
        6E 74 65 64 20 70 65 72
        20 50 61 67 65 20 7B
2992 4D      09520 MPROMPT DB      'Margin Setting {'
        61 72 67 69 6E 20 53 65
        74 74 69 6E 67 20 7B
29A2 50      09530 PPROMPT DB      'Physical Page Length {'
        68 79 73 69 63 61 6C 20
        50 61 67 65 20 4C 65 6E
        67 74 68 20 7B
29B8 54      09540 TPROMPT DB      'Tab Expansion {'
        61 62 20 45 78 70 61 6E
        73 69 6F 6E 20 7B
29C7 58      09550 XPROMF DB      'Xlate From {'
        6C 61 74 65 20 46 72 6F
        6D 20 7B
29D3 58      09560 XPROMT DB      'Xlate To {'
        6C 61 74 65 20 54 6F 20
        7B
                09570 ;
                09580 ;
29DD 20      09590          DB      ' '
0005      09600 TEMBUF DS      5
000A      09610 DUPDA DS      10
                09620 ;
2400      09630          END      START

```

\$FF	286F	@@1	0000	@@2	0000
@@3	0000	@@4	0000	@MOD2	0000
@MOD4	FFFF	ABB	0010	ABORT	241F
ADDLF	0007	AP	0027	APARM	27D7
APROMPT	2919	ARESP	2769	BREAK	0080
BS	0008	CALLINS	251B	CFLAG\$	0002
CHAROFF	249F	CHARS	0008	CHECKQ	2473
CHNGIND	269C	CKCOMM	24FC	CKCOMML	2502
CPARM	27CB	CPROMPT	2932	CR	000D
CRESP	2772	CUROFF	000F	CURON	000E
DATAAREA	25CF	DEFTAB	2758	DFLAG\$	0003
DISPA	259C	DISPFRM	2437	DISPROM	2558
DOFFHRD	24A5	DOINIT	2700	DOPRMP	253D
DOXLATE	28FF	DPARM	27DD	DRESP	27C5
DSFORMS	2489	DSP	26F1	DSPLY	26F7
DUNLP	259A	DUPDA	29E3	ENDPROM	2893
ETX	0003	EXDSP	26FB	EXIT	2407
FAKEPRM	27FF	FAKERES	286B	FAKETAB	2868
FFHARD	0007	FLAG	0040	FLAG?	26E3
FLAGBT	0007	FLOOP	25F2	FORMS	2429
FARM	27D9	FPROMPT	294F	FRESP	277C
GETFLAG	25F0	GETPRM	243E	HEX8	273C
HEXDEC	2721	IGSPCS	242D	INBUFF\$	2802
INDENT	0006	INITVAL	2619	INPUT	2743
IOERR	240E	IPARM	27CD	IPROMPT	2960
IRESP	2786	IXINST	25EB	IXINST2	2612
KFLAG\$	000A	LDEF	0042	LF	000A
LINES	0002	LPARM	27CF	LPROMPT	297A
LRESP	278F	MARGIN	0009	MORE0?	26D1
MPARM	27D1	MPROMPT	2992	MRESP	2799
NEXTFLG	2616	NEXTPR	2550	NOPARM	25EE
NOPF	2419	NOPF\$	2879	NOPOUT	25E6
NOSHOW	24F8	NUM	0080	NUMERIC	26DA
OFFSTR	2876	ONSTR	2873	PAGE	0000
PAR_ERR	002C	PDEF	0042	PNLP	258A
PNLP2	2597	PPARM	27D3	PPROMPT	29A2
PQUERY	2658	PRESP	27A1	PRLP	2561
PRMENT	2642	PRMERR	240C	PRMTBL\$	2762
PROMPT	252A	PROMPTL	2530	QPARM	27C9
QRESP	27AA	RCHARS	2674	REINPUT	2537
RESPTAB	27E4	RETADR	25C7	RINDENT	26BE
RLINES	266A	RMARGIN	26A4	RPAGE	2645
SADDLF	28E1	SAVESP	2422	SCHARS	28BA
SDLP	261E	SETNZ	26CE	SETZ	2668
SFFHARD	28EE	SFLAG\$	0012	SINDENT	28D4
SKIP	00DD	SKIPSET	2609	SLINES	28AD
SMARGIN	28C7	SPAGE	28A0	SPLP	2574
STAB	28FB	START	2400	STFDEF	2640
STFPRMS	25CC	STR	0020	STRTAB	280E
STUFDEF	2582	STUFFIN	2483	STUFLP	25D5
STUFVAL	25A3	SXLFROM	290B	SXLTO	2914
TAB	0009	TABV	0007	TEMBUF	29DE
TPARM	27DB	TPROMPT	29B8	TRESP	27B1
VALID1?	26C9	VALID2?	26CA	VALID?	26C6
VALUE	25B3	VALUES	2897	VFLAG\$	0015
XFER	26EB	XFERON	26E8	XFPARM	27E2
XFRESP	27DF	XLATEF	0004	XLATET	0005
XPROMF	29C7	XPROMT	29D3	XTPARM	27D5
XTRESP	27BA	@@ABORT	BD3C	@@ADTSK	BDCF
@@BANK	C2E7	@@BKSP	BFC7	@@BREAK	C2FD

@@CHNIO	BD27 @@CKBRKC	C34B @@CKDRV	BE23
@@CKEOF	BFDC @@CKTSK	BDBA @@CLOSE	BF82
@@CLS	C335 @@CMNDI	BD66 @@CMNDR	BD7B
@@CTL	BB8B @@DATE	BCFD @@DCSTAT	BE62
@@DEBUG	BDA5 @@DECHEX	C267 @@DIRRD	C1D4
@@DIRWR	C1E9 @@DIV16	C252 @@DIV8	C23D
@@DODIR	BE38 @@DSP	BB4F @@DSPLY	BBEF
@@ERROR	BD90 @@EXIT	BD51 @@FEXT	C141
@@FLAGS	C2D1 @@FNAME	C156 @@FSPEC	C12C
@@GATRD	C1BF @@GATWR	C1FE @@GET	BB63
@@GTDCEB	C180 @@GTDCT	C16B @@GTMOD	C195
@@HDFMT	BF0A @@HEX16	C2A6 @@HEX8	C291
@@HEXDEC	C27C @@HIGH\$	C2BB @@INIT	BF88
@@KBD	BBC7 @@KEY	BB3B @@KEYIN	BBDB
@@KLTSK	BE0E @@LOAD	C102 @@LOC	BFF1
@@LOF	C006 @@LOGGER	BC26 @@LOGOT	BC3B
@@MSG	BC72 @@MUL16	C228 @@MUL8	C213
@@OPEN	BF9D @@PARAM	BCE8 @@PAUSE	BCD3
@@PEOF	C01B @@POSN	C030 @@PRINT	BC87
@@PRT	BB9F @@PUT	BB77 @@RAMDIR	BE4D
@@RDSEC	BEE0 @@RDSSC	C1AA @@READ	C045
@@REMOV	BF73 @@RENAM	BF5E @@REW	C05A
@@RMTSK	BDE4 @@RPTSK	BDF9 @@RREAD	C06F
@@RSLCT	BECB @@RSTOR	BE8C @@RUN	C117
@@RWRTT	C084 @@SEEK	BEB6 @@SEEKSC	C099
@@SKIP	C0AE @@SLCT	BE77 @@STEPI	BEA1
@@TIME	BD12 @@VDCTL	BCBE @@VER	C0C3
@@VRSEC	BEF5 @@WEOF	C0D8 @@WHERE	BBB3
@@WRITE	C0ED @@WRSEC	BF1F @@WRSSC	BF34
@@WRTRK	BF49		

2400 is the transfer address

00000 Total errors

NOTES:

Command: FREE

Library: SYS7/SYS

ISAM # : 22H

```

00100 ;LBFREE/ASM - FREE Command
0000 00110 TITLE <FREE - LS-DOS 6.2>
00120 ;
0008 00130 TPL EQU 8 ;Tracks per Line = 8
0002 00140 @DSP EQU 2 ;@DSP SVC #
0006 00150 @PRT EQU 6 ;@PRT SVC #
000A 00160 @DSPLY EQU 10 ;@DSPLY SVC #
000E 00170 @PRINT EQU 14 ;@PRINT SVC #
0001 00180 @KEY EQU 1 ;@KEY SVC #
00190 ;
0000 00200 *GET SVCMAC:3 ;SVC Macro equivalents
00010 ;SVCMAC/ASM - LS-DOS Version VI
00020 *LIST OFF
0000 00300 *LIST ON
00210 *GET VALUES:3 ;Misc. equates
00320 ;VALUES/ASM - Version 6
00330 *LIST OFF
00420 *LIST ON
00220 ;
2400 00230 ORG 2400H
00240 ;
2400 00250 FREEMAP EQU $
00260 ;
2400 00270 @@CKBRKC ;See if break down
2400 3E6A 00001 LD A,106
2402 EF 00002 RST 40
2403 2804 00280 JR Z,BEGINA ;Ok if not,
2405 21FFFF 00290 LD HL,-1 ; else abort
2408 C9 00300 RET
00310 ;
00320 ; <BREAK> not hit - execute module
00330 ;
00340 BEGINA
2409 ED731424 00350 LD (SAVE SP+1),SP ;Save SP address for exit
240D CD2524 00360 CALL FREE ;Show Free Space
00370 ;
00380 ; Finished - Clear out <BREAK> & return
00390 ;
2410 210000 00400 EXIT LD HL,0 ;HL = 0 (normal exit)
2413 310000 00410 SAVESP LD SP,$-$ ;P/u old SP address
2416 00420 @@CKBRKC ;Clear <BREAK>
2416 3E6A 00003 LD A,106
2418 EF 00004 RST 40
2419 C9 00430 RET
00440 ;
00450 ; Error Handler - Display message & Abort
00460 ;
241A 6F 00470 IOERR LD L,A ;Set HL = Error #
241B 2600 00480 LD H,0
241D F6C0 00490 OR 0C0H ;Short error - & RETURN
241F 4F 00500 LD C,A ;Stuff error # in C
2420 00510 @@ERROR ;Display error
2420 3E1A 00005 LD A,26
2422 EF 00006 RST 40
2423 18EE 00520 JR SAVESP ;Exit
00530 ;
00540 ; FREE - Display Free Disk Space
00550 ;
00560 FREE
2425 00570 @@FLAGS ;IY => System Flags
2425 3E65 00007 LD A,101

```

```

2427 EF      00008      RST      40
              00580 ;
              00590 ;      Stuff Address of SFLAG$ into routine
              00600 ;
2428 111200    00610      LD      DE,SFLAG$      ;DE => Offset to SFLAG$
242B FD19      00620      ADD     IY,DE          ;IY => SFLAG$
242D FD226D26  00630      LD      (SFLAG1+1),IY    ;Save for later test
              00640 ;
              00650 ;      Position to parameters or end of line
              00660 ;
2431 E5        00670      PUSH    HL              ;Save command ptr
2432 7E        00680 SKPLP      LD      A,(HL)      ;P/u char
2433 FE28      00690      CP      '('              ;Parameter(s) ?
2435 2807      00700      JR      Z,GETPRMS        ;Yes - get parameters
2437 FE0D      00710      CP      CR              ;End of line ?
2439 2809      00720      JR      Z,GETHL          ;Recover command ptr
243B 23        00730      INC     HL              ;Bump ptr
243C 18F4      00740      JR      SKPLP            ;No - go til terminator
              00750 ;
              00760 ;      Process any parameters if entered
              00770 ;
243E 116828    00780 GETPRMS LD      DE,PRMTBL$    ;DE => Parameter table
2441          00790      @PARAM      @PARAM      ;@PARAM
2441 3E11      00009      LD      A,17
2443 EF        00010      RST      40
2444 E1        00800 GETHL     POP     HL          ;Recover cmdline ptr
2445 C21A24    00810      JP      NZ,IOERR          ;NZ - parameter error
              00820 ;
              00830 ;      Anything after FREE command entered ?
              00840 ;
2448 7E        00850      LD      A,(HL)          ;P/u first character
2449 FE29      00860      CP      '('+1          ;End of line ?
244B 3813      00870      JR      C,FREE0          ;Display lines
              00880 ;
              00890 ;      P/u next character if character is a colon
              00900 ;
244D FE3A      00910      CP      ':'              ;Drivspec ?
244F 2002      00920      JR      NZ,CKIFDRV        ;No - check if numeric
2451 23        00930      INC     HL              ;Yes - p/u next char
2452 7E        00940      LD      A,(HL)
              00950 ;
              00960 ;      Convert drive # to binary (if legal) & save
              00970 ;
2453 D630      00980 CKIFDRV  SUB     '0'          ;Legal drive Number ?
2455 3804      00990      JR      C,ILDRNUM        ;No - illegal drive #
2457 FE08      01000      CP      7+1
2459 383B      01010      JR      C,MAP            ;Less than 8 - good
              01020 ;
              01030 ;      Illegal Drive Number - display & Abort
              01040 ;
245B 3E20      01050 ILDRNUM LD      A,32          ;"Illegal Drive Number"
245D C31A24    01060      JP      IOERR            ;Display & Abort
              01070 ;
              01080 ;      Output a C/R to *PR if output is to *PR
              01090 ;
2460 2A4C26    01100 FREE0    LD      HL,(PPARM+1) ;P/u P parm
2463 2C        01110      INC     L              ;Specified ?
2464 200A      01120      JR      NZ,FREE0A        ;No - don't PRINT
2466 0E20      01130      LD      C,' '          ;Output space
2468 CD8B27    01140      CALL    PRT              ; to printer
246B 0E0D      01150      LD      C,CR            ;Output C/R

```

```

246D CD8B27 01160 CALL PRT ; to printer
2470 0E00 01170 FREE0A LD C,0 ;Init drive # to 0
01180 ;
01190 ; Is there a disk in the drive ?
01200 ;
2472 C5 01210 FREE1 PUSH BC ;Save drive #
2473 01220 @@GTDCT ;IY => DCT
2473 3E51 00011 LD A,81
2475 EF 00012 RST 40
2476 FD7E00 01230 LD A,(IY) ;Drive on line ?
2479 FEC3 01240 CP 0C3H
247B 2012 01250 JR NZ,NXTDRV ;No - get next drive
247D 01260 @@CKDRV ;Disk in drive ?
247D 3E21 00013 LD A,33
247F EF 00014 RST 40
01270 ;
01280 ; Check if the <BREAK> was hit
01290 ;
2480 F5 01300 PUSH AF ;Save @CKDRV condition
2481 CD9226 01310 CALL CKBREAK ;Check <BREAK>
2484 F1 01320 POP AF ;Recover @CKDRV cond
01330 ;
01340 ; Display <No Disk> if @CKDRV fails
01350 ;
2485 2805 01360 JR Z,DOINF ;Disk in - use header
2487 CD5426 01370 CALL NO_DISK ;No - display <No Disk>
248A 1803 01380 JR NXTDRV ;Get next drive
01390 ;
01400 ; Create Header String & display if successful
01410 ;
248C CD1425 01420 DOINF CALL GETINFO ;Display Header string
01430 ;
01440 ; Get next drive number
01450 ;
248F C1 01460 NXTDRV POP BC ;C = Drive #
2490 0C 01470 INC C ;Inc it
2491 CB59 01480 BIT 3,C ;Finished ?
2493 28DD 01490 JR Z,FREE1 ;No - get next drive
2495 C9 01500 RET ;Finished - RETurn
01510 ;
01520 ; MAP - Display Free Space Map
01530 ;
01540 ; Log in diskette if possible
01550 ;
2496 4F 01560 MAP LD C,A ;Xfer drive # to C
2497 01570 @@GTDCT ;IY => DCT + 0
2497 3E51 00015 LD A,81
2499 EF 00016 RST 40
249A FD7E00 01580 LD A,(IY) ;P/u enable/disable
249D FEC3 01590 CP 0C3H ;Drive enabled ?
249F C25B24 01600 JP NZ,ILDRNUM ;No - Illegal Drive #
24A2 01610 @@CKDRV ;Disk in drive ?
24A2 3E21 00017 LD A,33
24A4 EF 00018 RST 40
24A5 2806 01620 JR Z,DISKIN ;Good - Disk in Drive
01630 ;
01640 ; No Disk in Drive - Display message & Abort
01650 ;
24A7 CD5426 01660 CALL NO_DISK ;Display <No Disk>
24AA C31024 01670 JP EXIT ;Go to exit routine
01680 ;

```

```

01690 ;      Create header/footer strings & output header
01700 ;
24AD CD9226 01710 DISKIN CALL   CKBREAK      ;Check for <BREAK>
24B0 CD1425 01720      CALL   GETINFO      ;Get GAT, create header
24B3 CD2527 01730      CALL   DISPUND     ;Display underline
24B6 CD8426 01740      CALL   CLRLN       ;Clear Line buffer
01750 ;
01760 ;      Transfer " 0- 7" string to line buffer
01770 ;
24B9 21A427 01780      LD     HL,MTRK      ;Initial track #s
24BC 11002B 01790      LD     DE,LINBUF
24BF 010700 01800      LD     BC,7        ;Len track # display
24C2 EDB0    01810      LDIR
01820 ;
01830 ;      Pt HL => GAT+0, C = cylinder -1 (gets INCed)
01840 ;
24C4 210029 01850      LD     HL,GAT      ;Pt to stored GAT
24C7 0D      01860      DEC     C        ;Init Cyl = -1
01870 ;
01880 ;      Loop to Display each line of Cylinders
01890 ;
24C8 0608    01900 NEXTLIN LD     B,TPL      ;Max track per line count
24CA DD21082B 01910      LD     IX,LINBUF+8    ;Pt to display buffer
01920 ;
01930 ;      Bump cylinder number & display Gran info
01940 ;
24CE 0C      01950 DSPSC  INC     C        ;Current cylinder
24CF CDE326 01960      CALL   DFRE      ;Display Free grans
24D2 23      01970      INC     HL       ;Pt to next track
01980 ;
01990 ;      Finished Displaying all the cylinders ?
02000 ;
24D3 FD7E06 02010      LD     A,(IY+6)    ;P/u max cylinder
24D6 B9      02020      CP     C        ;Finished ?
24D7 281D    02030      JR     Z,ENDRET   ;Yes - display footer
02040 ;
02050 ;      Calculate offset (9-Grans/cyl) to next track
02060 ;
24D9 3AE526 02070      LD     A,(GRANS+1)  ;P/u Grans/Cyl
24DC ED44    02080      NEG
24DE C609    02090      ADD     A,TPL+1    ;A = offset to next
02100 ;
02110 ;      Add offset to Line buffer pointer (IX)
02120 ;
24E0 1600    02130      LD     D,0        ;Stuff in DE
24E2 5F      02140      LD     E,A
24E3 DD19    02150      ADD     IX,DE      ;Where to dsp next track
24E5 10E7    02160      DJNZ    DSPSC     ;Loop current 6 trks
02170 ;
02180 ;      Finished 8 cylinders - display line
02190 ;
24E7 E5      02200      PUSH   HL        ;Save buffer loc'n
24E8 C5      02210      PUSH   BC        ;Save current cyl
24E9 CD4226 02220      CALL   DSPLINE   ;Display current line
02230 ;
02240 ;      Clear granule display line buffer
02250 ;
24EC CD8426 02260      CALL   CLRLN      ;Clear Line buffer
24EF C1      02270      POP     BC        ;Recover cylinder # in C
02280 ;
02290 ;      Change cylinder numbers in line buffer

```

```

02300 ;
24F0 CD2D27 02310 CALL DSPTRK ;Calc new track #'s
24F3 E1 02320 POP HL ;Restore GAT pointer
24F4 18D2 02330 JR NEXTLIN ;Get next line
02340 ;
02350 ; Finished with drive - Display current line
02360 ;
24F6 CD4226 02370 ENDRET CALL DSPLINE ;Display tracks in buffer
24F9 CD2527 02380 CALL DISPUND ;Display underline
02390 ;
02400 ; Display Footer Message
02410 ;
24FC 21FB27 02420 LD HL,FOOTER ;HL => Footer Message
24FF CD4526 02430 CALL DSPMSG ;Display footer string
02440 ;
02450 ; If footer will cause a scroll - wait for key
02460 ;
2502 3A6126 02470 LD A,(CKPAGE+1) ;P/u # of lines left
2505 FE02 02480 CP 2 ;At least 2 lines left?
2507 3003 02490 JR NC,FRET ;Lprint, free to return
2509 CD7F27 02500 CALL KEY ;Wait for char
250C 360D 02510 FRET LD (HL),CR ;Scroll
250E CD4526 02520 CALL DSPMSG ;Display line
2511 C31024 02530 JP EXIT ;Go to normal exit
02540 ;
02550 ; Stuff Drive Number into String Header
02560 ;
2514 79 02570 GETINFO LD A,C ;P/u drive #
2515 C630 02580 ADD A,'0' ;Convert to ASCII
2517 32B227 02590 LD (HDRIVE),A ;Stuff into header string
02600 ;
02610 ; Read in the diskette's GAT
02620 ;
251A 02630 @@GTDCT ;IY => DCT+0
251A 3E51 00019 LD A,81
251C EF 00020 RST 40
251D FD5609 02640 LD D,(IY+9) ;P/u Directory cylinder
2520 1E00 02650 LD E,0 ;Sector Zero
2522 210029 02660 LD HL,GAT ;HL => GAT I/O buffer
2525 02670 @@RDSSC ;Read System Sector
2525 3E55 00021 LD A,85
2527 EF 00022 RST 40
2528 3E14 02680 LD A,14H ;Init to "GAT Read Error"
252A C21A24 02690 DERR JP NZ,IOERR ;Jump on GAT read error
252D CD9226 02700 CALL CKBREAK ;Check for <BREAK>
02710 ;
02720 ; Read in the diskette's HIT
02730 ;
2530 1C 02740 INC E ;Sector 1
2531 24 02750 INC H ;HL => HIT I/O buffer
2532 02760 @@RDSSC ;Read System Sector
2532 3E55 00023 LD A,85
2534 EF 00024 RST 40
2535 3E16 02770 LD A,16H ;Init to "HIT Read Error"
2537 20F1 02780 JR NZ,DERR ;Go to Error handler
2539 CD9226 02790 CALL CKBREAK ;Check for <BREAK>
02800 ;
02810 ; Pick up quantity of Sectors/Granule
02820 ;
253C FD7E08 02830 LD A,(IY+8) ;Bits 4-0 contain #
253F E61F 02840 AND 1FH ; of Sectors/Granule.

```

The Source	LIBRARY Files	FREE - LS-DOS 6.2	Page 00006
2541 3C	02850	INC A	;Adjust for zero offset
	02860 ;		
	02870 ;		Convert Sectors/Gran to K & stuff in string
	02880 ;		
2542 F5	02890	PUSH AF	;Save Sectors/Granule
2543 210000	02900	LD HL,0	;Set HLA = # Sec/Gran
2546 114428	02910	LD DE,FGRAN	;DE => Destination
2549 019C26	02920	LD BC,CVT2D	;Only 2 digits possible
254C CD5C27	02930	CALL CALCK2	;Convert to K
254F F1	02940	POP AF	;A = Sectors/Granule
2550 5F	02950	LD E,A	;Xfer to E
	02960 ;		
	02970 ;		Pick up number of cylinders in HL
	02980 ;		
2551 FD6E06	02990	LD L,(IY+6)	;P/u # of cylinders
2554 2600	03000	LD H,0	;Msb = 0
2556 23	03010	INC HL	;Relative to zero
	03020 ;		
	03030 ;		Calculate quantity of Granules/Cylinder
	03040 ;		
2557 F5	03050	PUSH AF	;Save # of sectors/gran
2558 FD7E08	03060	LD A,(IY+8)	;Bits 7-5 contain
255B E6E0	03070	AND 0E0H	; # of Granules/cylinder.
255D 07	03080	RLCA	; & shift to bits 0-2
255E 07	03090	RLCA	
255F 07	03100	RLCA	
2560 3C	03110	INC A	;Adjust for zero offset
2561 FDCB046E	03120	BIT 5,(IY+4)	;Double sided?
2565 2801	03130	JR Z,FREE2	;Bypass if one-sided
2567 87	03140	ADD A,A	;Else double the count
2568 32E526	03150 FREE2	LD (GRANS+1),A	;Save # Grans/Cyl
256B 4F	03160	LD C,A	;Save in C for @MULT8
	03170 ;		
	03180 ;		Calculate quantity of Sectors/Cylinder
	03190 ;		
256C	03200	@@MUL8	;Mult E x C
256C 3E5A	00025	LD A,90	
256E EF	00026	RST 40	
	03210 ;		
	03220 ;		A = quantity of Sectors per cylinder
	03230 ;		
256F F5	03240	PUSH AF	;Save # Sectors/Cyl
2570 E5	03250	PUSH HL	;Save # Cylinders
	03260 ;		
	03270 ;		File slots avail = 256 if more than 32 secs
	03280 ;		
2571 210001	03290	LD HL,256	;256 files maximum
2574 D602	03300	SUB 2	;Set A = # secs in dir
2576 FE20	03310	CP 20H	; Greater than 32 ?
2578 3006	03320	JR NC,FREE3	;Yes - use default of 256
	03330 ;		
	03340 ;		Calculate number of directory entries avail
	03350 ;		
257A 87	03360	ADD A,A	;Multiply # of Sectors
257B 87	03370	ADD A,A	; in directory by 8
257C 87	03380	ADD A,A	; to get # of slots.
257D 6F	03390	LD L,A	;Stuff in HL
257E 2600	03400	LD H,0	
2580 221D26	03410 FREE3	LD (FREE7+1),HL	;File slots to test later
	03420 ;		
	03430 ;		Stuff # of entries (HL) into header string


```

03440 ;
2583 11F727 03450 LD DE,HPOSSF ;DE => Destination
2586 CDA026 03460 CALL CVT3D ;Cvt HL to ASCII @ DE
2589 E1 03470 POP HL ;Recover # of cylinders
258A F1 03480 POP AF ;Rcvr # of sectors/cyl
03490 ;
03500 ; Calculate total # of sectors HL x A
03510 ;
258B 4F 03520 LD C,A ;Set C = Sec/cyl
258C B7 03530 OR A
258D 2803 03540 JR Z,SKIPMUL ;Don't multiply if zero
258F 03550 @MUL16 ;Multiply HL x C
258F 3E5B 00027 LD A,91
2591 EF 00028 RST 40
03560 ;
03570 ; Convert # of sectors to K & stuff in string
03580 ;
2592 11E027 03590 SKIPMUL LD DE,HPOSSK ;DE => Destination
2595 CD5927 03600 CALL CALCK ;Stuff into string
03610 ;
03620 ; Transfer Diskette Name from GAT into string
03630 ;
2598 21D029 03640 LD HL,GAT+0D0H ;HL => Pack Name
259B 11B527 03650 LD DE,HNAME ;DE => String destination
259E 0E08 03660 LD C,8 ;8 chars to xfer
25A0 EDB0 03670 LDIR ;Xfer into string
03680 ;
03690 ; Transfer Diskette Date from GAT into string
03700 ;
25A2 11BF27 03710 LD DE,HDATE ;DE => Destination
25A5 0E08 03720 LD C,8 ;8 chars to xfer
25A7 EDB0 03730 LDIR ;Xfer to string
03740 ;
03750 ; Pt HL => GAT, DE = Free Gran cnt, B = cyls
03760 ;
25A9 210029 03770 LD HL,GAT ;Pt to start of GAT
25AC 110000 03780 LD DE,0 ;Init gran counter
25AF 3ACC29 03790 LD A,(GAT+0CCH) ;P/u cyl excess
25B2 C623 03800 ADD A,35 ;Add base
25B4 47 03810 LD B,A ;Set loop counter
03820 ;
03830 ; Calculate quantity of Free granules left
03840 ;
25B5 7E 03850 FREE4 LD A,(HL) ;P/u GAT byte & set
25B6 37 03860 FREE5 SCF ; carry so bit 7 stays 1
03870 ;
03880 ; Is the granule in use ?
03890 ;
25B7 1F 03900 RRA ;Slide gran bit to carry
25B8 3801 03910 JR C,FREE6 ;Ignore if in use
03920 ;
03930 ; Free Granule - Bump Free Granule count
03940 ;
25BA 13 03950 INC DE ;Free, bump gran counter
25BB FEFF 03960 FREE6 CP 0FFH ;End of byte?
25BD 20F7 03970 JR NZ,FREE5 ;Loop if not
03980 ;
03990 ; Finished with one cylinder, advance to next
04000 ;
25BF 2C 04010 INC L ;Bump GAT byte pointer
25C0 10F3 04020 DJNZ FREE4 ;Loop for # cyls

```

```

04030 ;
04040 ;      Multiply # Grans (DE) by Sectors/Gran
04050 ;
25C2 EB 04060 EX DE,HL ;Xfer # Grans to HL
25C3 F1 04070 POP AF ;Rcvr # of sectors/gran
25C4 4F 04080 LD C,A ;Put in C for @MUL16
25C5 04090 @MUL16 ;Multiply HL x C
25C5 3E5B 00029 LD A,91
25C7 EF 00030 RST 40
04100 ;
04110 ;      Cvt # of Free Grans to K & stuff in string
04120 ;
25C8 11D627 04130 LD DE,HFREEK ;Cvrt to decimal
25CB CD5927 04140 CALL CALCK ;Cvrt to ASCII & stuff
04150 ;
04160 ;      Build Footer String in case of map
04170 ;
25CE 3E35 04180 LD A,'5' ;Init 5"/8" media
25D0 FDCB036E 04190 BIT 5,(IY+3) ;Test DCT for size
25D4 2802 04200 JR Z,FIVEIN ;Go if 5"
25D6 3E38 04210 LD A,'8' ;Else reset to 8"
25D8 320428 04220 FIVEIN LD (FSIZE),A ;Stuff size into header
04230 ;
04240 ;      P/u # of heads from DCT & stuff into footer
04250 ;
25DB FD7E07 04260 LD A,(IY+7) ;Bits 7-5 = # heads
25DE 07 04270 RLCA ;Shift to 0-2
25DF 07 04280 RLCA
25E0 07 04290 RLCA
25E1 E607 04300 AND 7 ;Mask off other junk
25E3 3C 04310 INC A ;Relative to zero
25E4 F630 04320 OR '0' ;Make it ASCII
25E6 321928 04330 LD (FHEADS),A ;Stuff into header
04340 ;
04350 ;      If this is a hard drive - ignore sides check
04360 ;
25E9 FDCB035E 04370 BIT 3,(IY+3) ;Check if hard
25ED 2810 04380 JR Z,DOSIDES ;Not hard - check sides
04390 ;
04400 ;      Hard Drive - overwrite Floppy in footer
04410 ;
25EF 219E27 04420 LD HL,HARD ;HL => "Hard "
25F2 110728 04430 LD DE,FTYPE ;DE => Dest in footer
25F5 010600 04440 LD BC,6 ;BC = 6 chars to xfer
25F8 EDB0 04450 LDIR ;Transfer to footer
25FA 219827 04460 LD HL,RIGID ;HL => "RIGID"
25FD 1815 04470 JR D3 ;Xfer "RIGID" to footer
04480 ;
04490 ;      Floppy disk - Stuff # of sides into footer
04500 ;
25FF 3E31 04510 DOSIDES LD A,'1' ;Init # of sides
2601 FDCB046E 04520 BIT 5,(IY+4) ;Test DCT for sides
2605 2801 04530 JR Z,ONESIDE ;Go if 1-sides
2607 3C 04540 INC A ;Else bump to 2
2608 321928 04550 ONESIDE LD (FHEADS),A ;Stuff into header
04560 ;
04570 ;      If floppy is double density pt HL to string
04580 ;
260B FDCB0376 04590 BIT 6,(IY+3) ;Test SDEN/DDEN
260F 280B 04600 JR Z,FREE7 ;Single - that's default
2611 219227 04610 LD HL,MDDEN ;Density MSG - Double

```

```

04620 ;
04630 ; Xfer "Single, Double, or Rigid" to footer
04640 ;
2614 112728 04650 D3 LD DE,FDENS ;Density MSG dsp pos
2617 010600 04660 LD BC,6 ;6 chars to xfer
261A EDB0 04670 LDIR ;Move Double to cover
04680 ;
04690 ; Calculate # of Free HIT positions available
04700 ;
261C 110000 04710 FREE7 LD DE,$-$ ;P/u # of poss entries
261F 21002A 04720 LD HL,HIT ;HL => HIT + 0
2622 1B 04730 FREE8 DEC DE ;Dec count in case of SYS
04740 ;
04750 ; Check SYS slots if this is a data disk
04760 ;
2623 3ACD29 04770 LD A,(GAT+0CDH) ;Bit 7 set if Data disk
2626 07 04780 RLCA
2627 3805 04790 JR C,DATDISK ;Set - ignore SYS check
04800 ;
04810 ; Is this a SYS slot - 00-07 or 20-27 ?
04820 ;
2629 7D 04830 LD A,L ;P/u HIT offset
262A E6D8 04840 AND 0D8H ;Reserved slot ?
262C 2805 04850 JR Z,FREE9 ;Yes - can't use it
04860 ;
04870 ; Not reserved - is the HIT posn in use ?
04880 ;
262E 7E 04890 DATDISK LD A,(HL) ;File in use?
262F B7 04900 OR A
2630 2001 04910 JR NZ,FREE9 ;Yes - don't bump count
04920 ;
04930 ; Slot not in use - bump free slot count
04940 ;
2632 13 04950 INC DE ;Bump free count
2633 2C 04960 FREE9 INC L ;Bump HIT pointer
2634 20EC 04970 JR NZ,FREE8 ;Loop if not through
04980 ;
04990 ; Stuff available files into string
05000 ;
2636 EB 05010 EX DE,HL ;Available files to HL
2637 11F327 05020 LD DE,HFREEF ;Cvrt to ASCII into msg
263A CDA026 05030 CALL CVT3D ;Convert & RETURN
05040 ;
05050 ; Display Header String & RETURN
05060 ;
263D 21AB27 05070 LD HL,HEADER ;HL => Header string
2640 1803 05080 JR DSPMSG ;Display header & RETURN
05090 ;
2642 21002B 05100 DSPLINE LD HL,LINBUF ;Fall into Display & RET
05110 ;
05120 ; DSPMSG - Display a message pointed to by HL
05130 ;
2645 CD9226 05140 DSPMSG CALL CKBREAK ;Check for <BREAK>
2648 CD8527 05150 CALL DSPLY ;Display message to video
264B 110000 05160 PPARM LD DE,$-$ ;P/u P parm
264E 1C 05170 INC E ;Was it entered ?
264F 200F 05180 JR NZ,CKPAGE ;No - Check page pause
2651 C38827 05190 JP PRINT ;Output line to *PR
05200 ;
2654 79 05210 NO_DISK LD A,C ;P/u drive #
2655 C630 05220 ADD A,'0' ;Cvt to ASCII

```

```

2657 325228 05230 LD (NODISK),A ;Stuff in string
265A 214B28 05240 LD HL,NODISK ;HL => Message
265D C34526 05250 JP DSPMSG ;Display Mess & RETurn
05260 ;
05270 ; Decrement Lines printed count
05280 ;
2660 3E16 05290 CKPAGE LD A,22 ;Ck for display pause
2662 3D 05300 DEC A ;Count down
2663 326126 05310 LD (CKPAGE+1),A ;Update
2666 C0 05320 RET NZ ;Ret if not yet full
05330 ;
05340 ; Printed a full page - Reset to count to max
05350 ;
2667 3E17 05360 LD A,23 ;Max lines to print
2669 326126 05370 LD (CKPAGE+1),A ;Reset to max
05380 ;
05390 ; Do not stop if a <D0> is in effect
05400 ;
266C 3A0000 05410 SFLAG1 LD A,($-$) ;P/u SFLAG$
266F E620 05420 AND 20H ;Do in effect ?
2671 C0 05430 RET NZ ;Yes - RETurn
05440 ;
05450 ; Wait for key - then clear screen
05460 ;
2672 CD7F27 05470 CALL KEY ;Wait for key entry
2675 05480 DISPHDR @@CLS ;Clear Screen
2675 3E69 00031 LD A,105
2677 EF 00032 RST 40
2678 C21A24 05490 JP NZ,IOERR ;Abort if I/O Error
05500 ;
05510 ; Display Map header
05520 ;
267B 21AB27 05530 LD HL,HEADER ;Point to the header
267E CD4526 05540 CALL DSPMSG ; & display it
2681 CD2527 05550 CALL DISPUND ;Display underline
05560 ;
05570 ; CLRLN - Clear line buffer
05580 ;
2684 3E20 05590 CLRLN LD A,' ' ;Clear buffer
2686 21002B 05600 BUFSTUF LD HL,LINBUF ;Point to buffer
2689 064F 05610 LD B,79 ;Length of buffer
268B 77 05620 CLRLN1 LD (HL),A ;Stuff with char given
268C 23 05630 INC HL
268D 10FC 05640 DJNZ CLRLN1
268F 360D 05650 LD (HL),CR ;End line with C/R
2691 C9 05660 RET
05670 ;
05680 ; CKBREAK - Check if the <BREAK> was pressed
05690 ;
2692 05700 CKBREAK EQU $
2692 05710 @@CKBRKC ;<BREAK> hit ?
2692 3E6A 00033 LD A,106
2694 EF 00034 RST 40
2695 C8 05720 RET Z ;No - RETurn
2696 21FFFF 05730 ABORT LD HL,-1 ;<BREAK> hit - abort
2699 C31324 05740 JP SAVESP
05750 ;
05760 ;
05770 ; CVTDEC - Convert Hex Number to Decimal ASCII
05780 ; CVD2D - CVD4D - Convert to 2,3, or 4 digits
05790 ;

```

```

058000 ;      HL => Hex Number to Convert
058100 ;      DE => Buffer to receive characters
058200 ;
269C 3E20 05830 CVT2D LD      A,' '
269E 181C 05840      JR      CVT10
26A0 3E20 05850 CVT3D LD      A,' '
26A2 1812 05860      JR      CVT100
26A4 3E20 05870 CVD4D LD      A,' '
26A6 1808 05880      JR      CVT1000
26A8 3E20 05890 CVTDEC LD      A,' '
059000 ;
26AA 011027 05910      LD      BC,10000
26AD CDC826 05920      CALL    CVD1
26B0 01E803 05930 CVT1000 LD      BC,1000
26B3 CDC826 05940      CALL    CVD1
26B6 016400 05950 CVT100 LD      BC,100
26B9 CDC826 05960      CALL    CVD1
26BC 010A00 05970 CVT10 LD      BC,10
26BF CDC826 05980      CALL    CVD1
26C2 7D      05990      LD      A,L
26C3 C630    06000      ADD     A,'0'
26C5 12      06010      LD      (DE),A
26C6 13      06020      INC     DE
26C7 C9      06030      RET
060400 ;
26C8 D5      06050 CVD1  PUSH    DE
26C9 5F      06060      LD      E,A
26CA 16FF    06070      LD      D,0FFH
26CC AF      06080      XOR     A
26CD 14      06090 CVD2  INC     D
26CE ED42    06100      SBC     HL,BC
26D0 30FB    06110      JR      NC,CVD2
26D2 09      06120      ADD     HL,BC
26D3 7B      06130      LD      A,E
26D4 42      06140      LD      B,D
26D5 D1      06150      POP     DE
26D6 12      06160      LD      (DE),A
26D7 04      06170      INC     B
26D8 05      06180      DEC     B
26D9 2806    06190      JR      Z,CVD3
26DB 78      06200      LD      A,B
26DC C630    06210      ADD     A,'0'
26DE 12      06220      LD      (DE),A
26DF 3E30    06230      LD      A,'0'
26E1 13      06240 CVD3  INC     DE
26E2 C9      06250      RET
062600 ;
062700 ;      DFRE - Stuff a cylinder's Gran symbols in buffer
062800 ;
062900 ;      IX => Buffer to receive characters
063000 ;      HL => GAT cylinder byte to use
063100 ;
26E3 C5      06320 DFRE  PUSH    BC      ;Save C, cur cyl loc
26E4 0600    06330 GRANS LD      B,$-$      ;P/u Grans/Cylinder
063400 ;
063500 ;      Is this cylinder the directory ?
063600 ;
26E6 FD7E09 06370      LD      A,(IY+9)      ;P/u dir cyl from DCT
26E9 B9      06380      CP      C      ;Directory ?
26EA 282F    06390      JR      Z,DDIR      ;Yes - use "D"'s
064000 ;

```

```

06410 ;      Not the directory cyl - use "x", "." & "*"
06420 ;
26EC DDE5 06430 PUSH    IX      ;Save buffer pointer
26EE C5    06440 PUSH    BC      ;Save Grans/Cyl
06450 ;
06460 ;      Is the Granule in use ?
06470 ;
26EF CB0E 06480 DF1    RRC     (HL)      ;P/u next Granule
26F1 3E78 06490 LD      A,'x'      ;Init "in use"
26F3 3802 06500 JR      C,DF2      ;Set - use "x"
06510 ;
06520 ;      Granule isn't in use - stuff a "." in buffer
06530 ;
26F5 3E2E 06540 LD      A,'.'      ;Else free
26F7 DD7700 06550 DF2    LD      (IX+0),A      ;Stuff char
06560 ;
06570 ;      Bump buffer pointer & decrement G/C count
06580 ;
26FA DD23 06590 INC     IX      ;Next display loc
26FC 10F1 06600 DJNZ    DF1      ;Loop thru all grans
06610 ;
06620 ;      Recover Buff ptr, Grans/Cyl
06630 ;
26FE C1    06640 POP     BC      ;B = Grans per Cyl
26FF DDE1  06650 POP     IX      ;IX to start of track
06660 ;
06670 ;      Position HL to Lockout table
06680 ;
2701 E5    06690 PUSH    HL      ;Save Cyl ptr
2702 116000 06700 LD      DE,60H      ;Offset to lockout table
2705 19    06710 ADD     HL,DE      ;Point to lockout
06720 ;
06730 ;      Go through lockout & overwrite if locked out
06740 ;
2706 FDCB035E 06750 L01    BIT     3,(IY+3)      ;If hard drive, there's
270A 2008    06760 JR      NZ,L02      ; no lockout
06770 ;
06780 ;      Diskette is a floppy - Is gran locked out ?
06790 ;
270C CB0E 06800 RRC     (HL)      ;Gran locked out ?
270E 3004 06810 JR      NC,L02      ;No - bump buff ptr
2710 DD36002A 06820 LD      (IX+0),'*'      ;Asterisk = lockout
06830 ;
06840 ;      Bump buffer pointer & loop til done
06850 ;
2714 DD23 06860 L02    INC     IX      ;Next gran dsp loc
2716 10EE  06870 DJNZ    L01      ;B grans/cyl
06880 ;
06890 ;      Recover ptrs & RETurn
06900 ;
2718 E1    06910 POP     HL
2719 C1    06920 POP     BC
271A C9    06930 RET
06940 ;
06950 ;      DDIR - Use "D"'s for Directory instead of "x"
06960 ;
271B DD360044 06970 DDIR    LD      (IX+0),'D'      ;Stuff "D" char
271F DD23  06980 INC     IX      ;Loop thru all DIR grans
2721 10F8  06990 DJNZ    DDIR
2723 C1    07000 POP     BC
2724 C9    07010 RET

```

```

07020 ;
07030 ;      DISPUND - Display a line of "-"
07040 ;
2725 3E2D 07050 DISPUND LD      A,'-'      ;Character to underline
2727 CD8626 07060      CALL    BUFSTUF      ;Stuff line & display
272A C34226 07070      JP      DSPLINE      ;Display line
07080 ;
07090 ;      DSPTRK - Stuff cylinder numbers in line buffer
07100 ;
272D D5 07110 DSPTRK  PUSH    DE            ;Save registers used
272E C5 07120      PUSH    BC
07130 ;
07140 ;      Stuff starting cylinder # in line buffer
07150 ;
272F 11002B 07160      LD      DE,LINBUF      ;Display buffer
2732 0C 07170      INC      C            ;Bump to next cylinder
2733 CD5027 07180      CALL    DSPTK4        ;Display cylinder number
07190 ;
07200 ;      Is this the only cylinder in the line ?
07210 ;
2736 FD7E06 07220      LD      A,(IY+6)      ;P/u maximum # cyls
2739 B9 07230      CP      C            ;Are we at the top?
273A 2811 07240      JR      Z,DSPTK3        ;Go if yes
07250 ;
07260 ;      More than 1 cyl - stuff "-" in line buffer
07270 ;
273C F5 07280      PUSH    AF            ;Save max cylinder
273D 3E2D 07290      LD      A,'-'      ;Stuff dash in buffer
273F 12 07300      LD      (DE),A
2740 F1 07310      POP     AF            ;Recover max cylinder
07320 ;
07330 ;      Get ending cylinder # for this line in C
07340 ;
2741 13 07350      INC     DE            ;Position to next avail
2742 0607 07360      LD      B,TPL-1        ;Need 7 more on a line
07370 ;
2744 0C 07380 DSPTK1  INC     C            ;Bump until 7 or max
2745 B9 07390      CP      C
2746 2802 07400      JR      Z,DSPTK2
2748 10FA 07410      DJNZ   DSPTK1
07420 ;
07430 ;      Stuff ending cylinder # into line buffer
07440 ;
274A CD5027 07450 DSPTK2  CALL    DSPTK4        ;Stuff ending cyl on line
274D C1 07460 DSPTK3  POP     BC            ;Recover registers
274E D1 07470      POP     DE
274F C9 07480      RET                    ;RETurn
07490 ;
07500 ;      Convert cylinder # (C) to ASCII at DE
07510 ;
2750 69 07520 DSPTK4  LD      L,C            ;Xfer cylinder # to HL
2751 2600 07530      LD      H,0
2753 C5 07540      PUSH    BC            ;Save cylinder #
2754 CDA026 07550      CALL    CVT3D        ;Convert cyl# to ASCII
2757 C1 07560      POP     BC            ;C = cylinder #
2758 C9 07570      RET                    ;RETurn
07580 ;
07590 ;      CALCK - Calculate Number of K & stuff in string
07600 ;      HLA => Total # of Sectors
07610 ;      DE => Destination of String
07620 ;

```

The Source	LIBRARY Files	FREE - LS-DOS 6.2	Page 00014
2759 01A826	07630 CALCK	LD BC,CVTDEC ;4 digit default	
275C ED436C27	07640 CALCK2	LD (CONVERT+1),BC	
2760 65	07650 DIVHLA	LD H,L ;Per disk pack	
2761 6F	07660	LD L,A	
2762 CB3C	07670	SRL H ;Divide total sectors	
2764 CB1D	07680	RR L ; by 4 to calculate	
2766 CB3C	07690	SRL H ; space in K	
2768 CB1D	07700	RR L	
	07710 ;		
	07720 ;	Convert K free (HL) to ASCII & put in string	
	07730 ;		
276A F5	07740	PUSH AF ;Save offset	
276B CD0000	07750 CONVERT	CALL \$-\$;Cvt HL to ASCII @ DE	
276E F1	07760	POP AF ;Recover offset	
	07770 ;		
	07780 ;	Stuff hundredths value into string	
	07790 ;		
276F 13	07800	INC DE ;Go past decimal point	
2770 E603	07810	AND 3 ;Modulo 4	
2772 87	07820	ADD A,A ;Multiply by 2	
2773 0600	07830	LD B,0 ; to position to	
2775 4F	07840	LD C,A ; hundredths string	
2776 216028	07850	LD HL,HUNDTAB ;HL => Table base	
2779 09	07860	ADD HL,BC ;HL => Hundredths string	
277A 0E02	07870	LD C,2 ;2 chars to xfer	
277C EDB0	07880	LDIR ;Stuff in string	
277E C9	07890	RET ;RETurn	
	07900 ;		
	07910 ;	KEY/DSP/DSPLY/PRT/PRINT - SVC routines	
	07920 ;		
277F 3E01	07930 KEY	LD A,@KEY ;Wait for key	
2781 11	07940	DB 11H	
	07950 ;		
2782 3E02	07960 DSP	LD A,@DSP ;Display byte	
2784 11	07970	DB 11H ;LD DE,nnnn	
	07980 ;		
2785 3E0A	07990 DSPLY	LD A,@DSPLY ;Display line	
2787 11	08000	DB 11H	
	08010 ;		
2788 3E0E	08020 PRINT	LD A,@PRINT ;Print line	
278A 11	08030	DB 11H	
	08040 ;		
278B 3E06	08050 PRT	LD A,@PRT ;Print byte	
	08060 ;		
278D EF	08070 DO_OUT	RST 40 ;Do SVC & check error	
278E C8	08080	RET Z ;RETurn if good	
278F C31A24	08090	JP IOERR ;NZ - I/O Error	
	08100 ;		
2792 44	08110 MDDEN	DB 'DOUBLE'	
4F 55 42	4C 45		
2798 52	08120 RIGID	DB 'RIGID '	
49 47 49	44 20		
279E 48	08130 HARD	DB 'Hard '	
61 72 64	20 20		
	08140 ;		
27A4 20	08150 MTRK	DB ' 0- 7'	
20 30 2D	20 20 37		
	08160 ;		
27AB 44	08170 HEADER	DB 'Drive :'	
72 69 76	65 20 3A		
27B2 64	08180 HDRIVE	DB 'd '	


```

20 20
27B5 64      08190 HNAME  DB      'diskname '
69 73 6B 6E 61 6D 65 20
20
27BF 64      08200 HDATE  DB      'dd/mm/yy   Free Space ='
64 2F 6D 6D 2F 79 79 20
20 20 46 72 65 65 20 53
70 61 63 65 20 3D
27D6 6E      08210 HFREEK DB      'nnnnn.nnK/'
6E 6E 6E 6E 2E 6E 6E 4B
2F
27E0 6E      08220 HPOSSK DB      'nnnnn.nnK  Files = '
6E 6E 6E 6E 2E 6E 6E 4B
20 20 46 69 6C 65 73 20
3D 20
27F3 64      08230 HFREFE DB      'ddd/'
64 64 2F
27F7 64      08240 HPOSSF DB      'ddd',CR
64 64 0D
      08250 ;
27FB 54      08260 FOOTER DB      'Type => '
79 70 65 20 3D 3E 20 20
2804 73      08270 FSIZE  DB      's" '
22 20
2807 46      08280 FTYPE  DB      'Floppy   Heads = '
6C 6F 70 70 79 20 20 20
20 48 65 61 64 73 20 3D
20
2819 6E      08290 FHEADS DB      'n   Density = '
20 20 20 44 65 6E 73 69
74 79 20 3D 20
2827 53      08300 FDENS  DB      'SINGLE   Note - 1 Position = '
49 4E 47 4C 45 20 20 20
4E 6F 74 65 20 2D 20 31
20 50 6F 73 69 74 69 6F
6E 20 3D 20
2844 6E      08310 FGRAN  DB      'nn.nnK'
6E 2E 6E 6E 4B
284A 0D      08320 ENDFOOT DB      CR
      08330 ;
284B 44      08340 NODISK DB      'Drive : '
72 69 76 65 20 3A
2852 64      08350 NODISKN DB      'd [No Disk]',CR
20 20 5B 4E 6F 20 20 44
69 73 6B 5D 0D
      08360 ;
2860 30      08370 HUNDTAB DB      '00255075'
30 32 35 35 30 37 35
      08380 ;
      08390 ;      Parameter Table
      08400 ;
2868 80      08410 PRMTBL$ DB      80H
2869 41      08420          DB      FLAG!1
286A 50      08430          DB      'P'
286B 00      08440          DB      0
286C 4C26    08450          DW      PPARM+1
286E 00      08460          NOP
      08470 ;
2900      08480          ORG      $<-8+1<+8
0100      08490 GAT      DS      256
0100      08500 HIT      DS      256

```

2B00

08510 LINBUF EQU

\$

08520 ;

2400

08530

END

FREEMAP

@@1	0000 @@2	0000 @@3	0000
@@4	0000 @DSP	0002 @DSPLY	000A
@KEY	0001 @MOD2	0000 @MOD4	FFFF
@PRINT	000E @PRT	0006 ABB	0010
ABORT	2696 AP	0027 BEGINA	2409
BREAK	0080 BS	0008 BUFSTUF	2686
CALCK	2759 CALCK2	275C CFLAG\$	0002
CKBREAK	2692 CKIFDRV	2453 CKPAGE	2660
CLRLN	2684 CLRLN1	268B CONVERT	276B
CR	000D CVD1	26C8 CVD2	26CD
CVD3	26E1 CVD4D	26A4 CVT10	26BC
CVT100	26B6 CVT1000	26B0 CVT2D	269C
CVT3D	26A0 CVTDEC	26A8 D3	2614
DATDISK	262E DDIR	271B DERR	252A
DF1	26EF DF2	26F7 DFLAG\$	0003
DFRE	26E3 DISKIN	24AD DISPHDR	2675
DISPUND	2725 DIVHLA	2760 DOINF	248C
DOSIDES	25FF DO OUT	278D DSP	2782
DSPLINE	2642 DSPLY	2785 DSPMSG	2645
DSPSC	24CE DSPTK1	2744 DSPTK2	274A
DSPTK3	274D DSPTK4	2750 DSPTRK	272D
ENDFOOT	284A ENDRET	24F6 ETX	0003
EXIT	2410 FDENS	2827 FGRAN	2844
FHEADS	2819 FIVEIN	25D8 FLAG	0040
FOOTER	27FB FREE	2425 FREE0	2460
FREE0A	2470 FREE1	2472 FREE2	2568
FREE3	2580 FREE4	25B5 FREE5	25B6
FREE6	25BB FREE7	261C FREE8	2622
FREE9	2633 FREEMAP	2400 FRET	250C
FSIZE	2804 FTYPE	2807 GAT	2900
GETHL	2444 GETINFO	2514 GETPRMS	243E
GRANS	26E4 HARD	279E HDATE	27BF
HDRIVE	27B2 HEADER	27AB HFREF	27F3
HFREEK	27D6 HIT	2A00 HNAME	27B5
HPOSSF	27F7 HPOSSK	27E0 HUNDTAB	2860
ILDRNUM	245B IOERR	241A KEY	277F
KFLAG\$	000A LF	000A LINBUF	2B00
LO1	2706 LO2	2714 MAP	2496
MDDEN	2792 MTRK	27A4 NEXTLIN	24C8
NODISK	284B NODISKN	2852 NO_DISK	2654
NUM	0080 NXTDRV	248F ONESIDE	2608
PAR_ERR	002C PPARM	264B PRINT	2788
PRMTBL\$	2868 PRT	278B RIGID	2798
SAVESP	2413 SFLAG\$	0012 SFLAG1	266C
SKIPMUL	2592 SKPLP	2432 STR	0020
TAB	0009 TPL	0008 VFLAG\$	0015
@@ABORT	B8C9 @@ADTSK	B95C @@BANK	BE74
@@BKSP	BB54 @@BREAK	BE8A @@CHNIO	B8B4
@@CKBRKC	BED8 @@CKDRV	B9B0 @@CKEOF	BB69
@@CKTSK	B947 @@CLOSE	BB3F @@CLS	BEC2
@@CMNDI	B8F3 @@CMNDR	B908 @@CTL	B718
@@DATE	B88A @@DCSTAT	B9EF @@DEBUG	B932
@@DECHEX	BDF4 @@DIRRD	BD61 @@DIRWR	BD76
@@DIV16	BDDF @@DIV8	BDCA @@DODIR	B9C5
@@DSP	B6DC @@DSPLY	B77C @@ERROR	B91D
@@EXIT	B8DE @@FEXT	BCCE @@FLAGS	BE5E
@@FNAME	BCE3 @@FSPEC	BCB9 @@GATRD	BD4C
@@GATWR	BD8B @@GET	B6F0 @@GTDCB	BD0D
@@GTDCT	BCF8 @@GTMOD	BD22 @@HDFMT	BA97
@@HEX16	BE33 @@HEX8	BE1E @@HEXDEC	BE09

@@HIGH\$	BE48 @@INIT	BB15 @@KBD	B754
@@KEY	B6C8 @@KEYIN	B768 @@KLTSK	B99B
@@LOAD	BC8F @@LOC	BB7E @@LOF	BB93
@@LOGGER	B7B3 @@LOGOT	B7C8 @@MSG	B7FF
@@MUL16	BDB5 @@MUL8	BDA0 @@OPEN	BB2A
@@PARAM	B875 @@PAUSE	B860 @@PEOF	BBA8
@@POSN	BBBD @@PRINT	B814 @@PRT	B72C
@@PUT	B704 @@RAMDIR	B9DA @@RDSEC	BA6D
@@RDSSC	BD37 @@READ	BBD2 @@REMOV	BB00
@@RENAM	BAEB @@REW	BBE7 @@RMTSK	B971
@@RPTSK	B986 @@RREAD	BBFC @@RSLCT	BA58
@@RSTOR	BA19 @@RUN	BCA4 @@RWIT	BC11
@@SEEK	BA43 @@SEEKSC	BC26 @@SKIP	BC3B
@@SLCT	BA04 @@STPI	BA2E @@TIME	B89F
@@VDCTL	B84B @@VER	BC50 @@VRSEC	BA82
@@WEOF	BC65 @@WHERE	B740 @@WRITE	BC7A
@@WRSEC	BAAC @@WRSSC	BAC1 @@WRTRK	BAD6

2400 is the transfer address

00000 Total errors

NOTES:

NOTES:

NOTES:

Command: LIB, CLS, TOF

Library: SYS6/SYS

ISAM # : 19H, 24H, 25H


```

00100 ;LBLIB/ASM - LIB/CLS/TOF Commands
0000 00110 TITLE <LIB - LS-DOS 6.2>
0000 00120 *GET SVCMAC:3 ;SVC Macro equivalents
00010 ;SVCMAC/ASM - LS-DOS Version VI
00020 *LIST OFF
03900 *LIST ON
00130 ;
000D 00140 CR EQU 13
000A 00150 LF EQU 10
1E00 00160 OVLAY$ EQU 1E00H
00170 ;
2400 00180 ORG 2400H
00190 ;
00200 ;
00210 ; LIB command entry point
00220 ;
2400 C32024 00230 LIB JP JPLIB ;Skip around to 'LIB'
00240 ;
00250 ; CLS command entry point
00260 ;
2403 C31924 00270 JP CLS ;Skip around to 'CLS'
00280 ;
00290 ; TOF command entry point
00300 ;
00310 TOF
2406 0E0C 00320 LD C,12 ;Load TOF character
2408 00330 @PRT ;Do @PRT SVC
2408 3E06 00001 LD A,6
240A EF 00002 RST 40
240B 210000 00340 LD HL,0 ;Init to no error
240E C8 00350 RET Z ; and back if ok
240F 6F 00360 TOFERR LD L,A ;Error code to HL
2410 2600 00370 LD H,0
2412 F6C0 00380 OR C,0H ;Abbrev, return
2414 4F 00390 LD C,A
2415 00400 @@ERROR
2415 3E1A 00003 LD A,26
2417 EF 00004 RST 40
2418 C9 00410 RET
00420 ;
2419 3E69 00430 CLS LD A,105 ;@CLS svc
241B EF 00440 RST 40 ; and do it
241C AF 00450 XOR A ;Ret without error
241D 6F 00460 LD L,A
241E 67 00470 LD H,A
241F C9 00480 RET
00490 ;
2420 0603 00500 JPLIB LD B,3 ;Init for 3 libraries
2422 C5 00510 LIB0 PUSH BC
2423 00520 @DSPLY LIBMSG
00005 IFEQ 01H,1
2423 219024 00006 LD HL,LIBMSG
00007 ENDIF
2426 3E0A 00008 LD A,10
2428 EF 00009 RST 40
2429 360A 00530 LD (HL),LF ;Put LF at start of
00540 ; string for next time
242B 2A021E 00550 LD HL,(OVLAY$+2) ;PU table pointer
242E 0602 00560 LIB1 LD B,2 ;Dsply 1st command
2430 CD8824 00570 CALL TAB ; tabbed in 2 spaces
2433 0E07 00580 LD C,7 ;Init for 7-across

```

```

2435 7E      00590 LIB2 LD      A,(HL)
2436 B7      00600      OR      A
2437 2837    00610      JR      Z,LIB4      ;Jump on end
2439 E5      00620      PUSH    HL
243A 110700  00630      LD      DE,7      ;Index to lib #
243D 19      00640      ADD     HL,DE
243E 3E80    00650 LIBX LD      A,80H      ;Init for LIB-A
2440 BE      00660      CP      (HL)      ;Is this command in
2441 2804    00670      JR      Z,LIB2A      ; the current library?
2443 F1      00680      POP     AF          ;Is not, skip past it
2444 23      00690      INC     HL
2445 18EE    00700      JR      LIB2
          00710 ;
2447 E1      00720 LIB2A POP     HL          ;Get 1st char of command
2448 C5      00730      PUSH    BC          ;Save reg C
2449 4E      00740      LD      C,(HL)      ; and display in upper
244A        00750      @DASP
244A 3E02    00010      LD      A,2
244C EF      00011      RST      40
244D 0605    00760      LD      B,5          ;Write 6-char LIB word
244F 23      00770 LIB3 INC     HL          ;Point to next char
2450 7E      00780      LD      A,(HL)
2451 FE20    00790      CP      ' '          ;If space, don't lower
2453 2802    00800      JR      Z,$+4      ; case the char
2455 EE20    00810      XOR     20H
2457 4F      00820      LD      C,A          ;Xfer to C & display it
2458        00830      @DASP
2458 3E02    00012      LD      A,2
245A EF      00013      RST      40
245B 10F2    00840      DJNZ    LIB3
          00850 ;
245D 0605    00860      LD      B,5          ;Move over 5 spaces
245F CD8824  00870      CALL    TAB          ; for start of next command
2462 23      00880      INC     HL          ;Bypass LIB parm vector
2463 23      00890      INC     HL
2464 23      00900      INC     HL
2465 C1      00910      POP     BC          ;Get across counter
2466 0D      00920      DEC     C          ; & decrement
2467 20CC    00930      JR      NZ,LIB2      ;Loop on < 7
2469 0E0D    00940      LD      C,CR        ;Write a new line
246B        00950      @DASP
246B 3E02    00014      LD      A,2
246D EF      00015      RST      40
246E 18BE    00960      JR      LIB1          ;Loop
          00970 ;
2470 0E0D    00980 LIB4 LD      C,CR        ;End with new line
2472        00990      @DASP
2472 3E02    00016      LD      A,2
2474 EF      00017      RST      40
2475 219A24  01000      LD      HL,LIBMSG+10
2478 34      01010      INC     (HL)          ;Bump to next lib
2479 3A3F24  01020      LD      A,(LIBX+1)    ;Advance RST code also
247C C620    01030      ADD     A,20H
247E 323F24  01040      LD      (LIBX+1),A
2481 C1      01050      POP     BC          ;Recover Lib count
2482 109E    01060      DJNZ    LIB0          ;Loop until all done
2484 210000  01070      LD      HL,0          ;Set no error
2487 C9      01080      RET
          01090 ;
2488 0E20    01100 TAB LD      C,' '          ;Display spaces,
248A        01110      @DASP

```

248A 3E02	00018	LD	A,2
248C EF	00019	RST	40
248D 10F9	01120	DJNZ	TAB ; reg B has count
248F C9	01130	RET	
	01140 ;		
2490 00	01150 LIBMSG DB		0,'Library <A>',CR
4C 69 62	72 61 72 79 20		
3C 41 3E	0D		
2400	01160	END	LIB

@@1	0000 @@2	0000 @@3	0000
@@4	0000 @MOD2	0000 @MOD4	FFFF
CLS	2419 CR	0000 JPLIB	2420
LF	0000A LIB	2400 LIB0	2422
LIB1	242E LIB2	2435 LIB2A	2447
LIB3	244F LIB4	2470 LIBMSG	2490
LIBX	243E OVLAY\$	1E00 TAB	2488
TOF	2406 TOFERR	240F @@ABORT	748C
@@ADTSK	751F @@BANK	7A37 @@BKSP	7717
@@BREAK	7A4D @@CHNIO	7477 @@CKBRKC	7A9B
@@CKDRV	7573 @@CKEOF	772C @@CKTSK	750A
@@CLOSE	7702 @@CLS	7A85 @@CMNDI	74B6
@@CMNDR	74CB @@CTL	72DB @@DATE	744D
@@DCSTAT	75B2 @@DEBUG	74F5 @@DECHEX	79B7
@@DIRRD	7924 @@DIRWR	7939 @@DIV16	79A2
@@DIV8	798D @@DODIR	7588 @@DSP	729F
@@DSPLY	733F @@ERROR	74E0 @@EXIT	74A1
@@FEXT	7891 @@FLAGS	7A21 @@FNAME	78A6
@@FSPEC	787C @@GATRD	790F @@GATWR	794E
@@GET	72B3 @@GTDCB	78D0 @@GTDCT	78BB
@@GTMOD	78E5 @@HDFMT	765A @@HEX16	79F6
@@HEX8	79E1 @@HEXDEC	79CC @@HIGH\$	7A0B
@@INIT	76D8 @@KBD	7317 @@KEY	728B
@@KEYIN	732B @@KLTSK	755E @@LOAD	7852
@@LOC	7741 @@LOF	7756 @@LOGGER	7376
@@LOGOT	738B @@MSG	73C2 @@MUL16	7978
@@MUL8	7963 @@OPEN	76ED @@PARAM	7438
@@PAUSE	7423 @@PEOF	776B @@POSN	7780
@@PRINT	73D7 @@PRT	72EF @@PUT	72C7
@@RAMDIR	759D @@RDSEC	7630 @@RDSSC	78FA
@@READ	7795 @@REMOV	76C3 @@RENAM	76AE
@@REW	77AA @@RMTSK	7534 @@RPTSK	7549
@@RREAD	77BF @@RSLCT	761B @@RSTOR	75DC
@@RUN	7867 @@RWRT	77D4 @@SEEK	7606
@@SEEKSC	77E9 @@SKIP	77FE @@SLCT	75C7
@@STEPI	75F1 @@TIME	7462 @@VDCTL	740E
@@VER	7813 @@VRSEC	7645 @@WEOF	7828
@@WHERE	7303 @@WRITE	783D @@WRSEC	766F
@@WRSSC	7684 @@WRTRK	7699	

2400 is the transfer address
00000 Total errors

NOTES:

Command: LINK

Library: SYS6/SYS

ISAM # : 62H

```

00100 ;LBLINK/ASM - LINK Command
0000 00110 TITLE <LINK - LS-DOS 6.2>
00120 ;
000D 00130 CR EQU 13
0000 00140 *GET SVCMAC:3 ;SVC Macro equivalents
00010 ;SVC MAC/ASM - LS-DOS Version VI
00020 *LIST OFF
03900 *LIST ON
00150 ;
2400 00160 ORG 2400H
00170 ;
2400 11AA24 00180 LINK LD DE,FCB1 ;Fetch source spec
2403 00190 @@FSPEC
2403 3E4E 00001 LD A,78
2405 EF 00002 RST 40
2406 2010 00200 JR NZ,SPCERR1 ;Exit if bad name
2408 1A 00210 LD A,(DE) ;Must be a device
2409 FE2A 00220 CP '*'
240B 207E 00230 JR NZ,SPCERR
00240 ;
00250 ; Fetch the second device spec
00260 ;
240D 11AD24 00270 LD DE,FCB2 ;Fetch destination spec
2410 00280 @@FSPEC
2410 3E4E 00003 LD A,78
2412 EF 00004 RST 40
2413 2076 00290 JR NZ,SPCERR ;Exit if bad name
2415 1A 00300 LD A,(DE)
2416 FE2A 00310 CP '*' ;Must also be a device
2418 2071 00320 SPCERR1 JR NZ,SPCERR
00330 ;
00340 ; Make sure source <> destination
00350 ;
241A 2AAB24 00360 LD HL,(FCB1+1) ;If devices are the same,
241D ED5BAE24 00370 LD DE,(FCB2+1) ; then quit
2421 ED52 00380 SBC HL,DE
2423 2866 00390 JR Z,SPCERR
00400 ;
00410 ; Locate a spare DCB for the link
00420 ;
2425 110000 00430 LD DE,0
2428 00440 @@GTDCB
2428 3E52 00005 LD A,82
242A EF 00006 RST 40
242B 3E21 00450 LD A,33 ;Init "No device space..."
242D 2052 00460 JR NZ,IOERR
242F 224A24 00470 LD (LINKDCB+1),HL ;Save pointer
00480 ;
00490 ; Locate destination DCB address
00500 ;
2432 ED5BAE24 00510 LD DE,(FCB2+1) ;Grab DCB name
2436 00520 @@GTDCB ;Locate its address
2436 3E52 00007 LD A,82
2438 EF 00008 RST 40
2439 2046 00530 JR NZ,IOERR ;Jump if not found
243B 226824 00540 LD (DSTDCB+1),HL ;Save destination
00550 ;
00560 ; Locate source DCB address
00570 ;
243E ED5BAB24 00580 LD DE,(FCB1+1) ;Get 1st DCB name
2442 00590 @@GTDCB ;Locate in device tables

```

The Source	LIBRARY Files	LINK - LS-DOS 6.2	Page 00002
2442 3E52	00009	LD A,82	
2444 EF	00010	RST 40	
2445 203A	00600	JR NZ,IOERR	;Jump if not found
2447 E5	00610	PUSH HL	;Save pointer we used
2448 F3	00620	DI	;Can't interrupt
	00630 ;		
	00640 ;	Save the old device vector while stuffing new	
	00650 ;		
2449 010000	00660 LINKDCB	LD BC,\$-\$;P/u link DCB address
244C 2C	00670	INC L	;Bump to vector
244D 7E	00680	LD A,(HL)	;Save what's there
244E 71	00690	LD (HL),C	;Stuff link address
244F 4F	00700	LD C,A	; into DCB of source
2450 2C	00710	INC L	; while saving old
2451 7E	00720	LD A,(HL)	; vector for storage
2452 70	00730	LD (HL),B	; (could be a FCB)
2453 47	00740	LD B,A	
	00750 ;		
	00760 ;	Now set LINK bit and rest of LINK DCB block	
	00770 ;		
2454 E1	00780	POP HL	;Rcvr ptr to source DCB+0
2455 7E	00790	LD A,(HL)	;Init the LINK bit
2456 F5	00800	PUSH AF	;Save old TYPE byte
2457 E607	00810	AND 7	;Strip flags
2459 F620	00820	OR 20H	;Set Link bit
245B 77	00830	LD (HL),A	;Show source is linked
245C 2A4A24	00840	LD HL,(LINKDCB+1)	;P/u link DCB address
245F F1	00850	POP AF	;Rcvr source TYPE
2460 77	00860	LD (HL),A	;New LINK TYPE
2461 2C	00870	INC L	
2462 71	00880	LD (HL),C	;Stuff source vector
2463 2C	00890	INC L	
2464 70	00900	LD (HL),B	
2465 2C	00910	INC L	;Bypass dest TYPE
2466 2C	00920	INC L	
2467 010000	00930 DSTDCB	LD BC,\$-\$;P/u destination DCB addr
246A 71	00940	LD (HL),C	; & stuff into link DCB
246B 2C	00950	INC L	
246C 70	00960	LD (HL),B	
246D 2C	00970	INC L	
246E E5	00980	PUSH HL	;Save name field pointer
246F 114C2F	00990	LD DE,'/L'	;Let's find a link name
2472 14	01000 NAMLP	INC D	;Bump "2nd" character
2473	01010	@GTDCB	;If we find this name
2473 3E52	00011	LD A,82	
2475 EF	00012	RST 40	
2476 28FA	01020	JR Z,NAMLP	; look for another
2478 E1	01030	POP HL	;Get name pointer
2479 73	01040	LD (HL),E	; & stuff in the
247A 2C	01050	INC L	; selected link name
247B 72	01060	LD (HL),D	
247C FB	01070	EI	;Start tasks again
247D 210000	01080	LD HL,0	;Show no error
2480 C9	01090	RET	
	01100 ;		
	01110 ;	Error processing	
	01120 ;		
2481 6F	01130 IOERR	LD L,A	;Move error # into HL
2482 2600	01140	LD H,0	
2484 F6C0	01150	OR 0C0H	;Abbrev & return
2486 4F	01160	LD C,A	


```

2487      01170      @@ERROR
2487 3E1A      00013      LD      A,26
2489 EF      00014      RST      40
248A C9      01180      RET
248B 219524   01190 SPCERR LD      HL,SPCERR$      ;Bad devspec found
248E      01200      @@LOGOT
      00015      IFEQ      00H,1
      00016      LD      HL,
      00017      ENDIF
248E 3E0C      00018      LD      A,12
2490 EF      00019      RST      40
2491 21FFFF   01210      LD      HL,-1
2494 C9      01220      RET
2495 44      01230 SPCERR$ DB      'Device spec required',CR
      65 76 69 63 65 20 73 70
      65 63 20 72 65 71 75 69
      72 65 64 0D
0003      01240 FCB1      DS      3      ;Only 3-bytes needed
0020      01250 FCB2      DS      32
2400      01260      END      LINK

```

@@1	0000 @@2	0000 @@3	0000
@@4	0000 @MOD2	0000 @MOD4	FFFF
CR	000D DSTDCB	2467 FCB1	24AA
FCB2	24AD IOERR	2481 LINK	2400
LINKDCB	2449 NAMLP	2472 SPCERR	248B
SPCERR\$	2495 SPCERR1	2418 @@ABORT	7678
@@ADTSK	770B @@BANK	7C23 @@BKSP	7903
@@BREAK	7C39 @@CHNIO	7663 @@CKBRKC	7C87
@@CKDRV	775F @@CKEOF	7918 @@CKTSK	76F6
@@CLOSE	78EE @@CLS	7C71 @@CMNDI	76A2
@@CMNDR	76B7 @@CTL	74C7 @@DATE	7639
@@DCSTAT	779E @@DEBUG	76E1 @@DECHEX	7BA3
@@DIRRD	7B10 @@DIRWR	7B25 @@DIV16	7B8E
@@DIV8	7B79 @@DODIR	7774 @@DSP	748B
@@DSPLY	752B @@ERROR	76CC @@EXIT	768D
@@FEXT	7A7D @@FLAGS	7C0D @@FNAME	7A92
@@FSPEC	7A68 @@GATRD	7AFB @@GATWR	7B3A
@@GET	749F @@GTDCB	7ABC @@GTDCI	7AA7
@@GTMOD	7AD1 @@HDFMT	7846 @@HEX16	7BE2
@@HEX8	7BCD @@HEXDEC	7BB8 @@HIGH\$	7BF7
@@INIT	78C4 @@KBD	7503 @@KEY	7477
@@KEYIN	7517 @@KLTSK	774A @@LOAD	7A3E
@@LOC	792D @@LOF	7942 @@LOGGER	7562
@@LOGOT	7577 @@MSG	75AE @@MUL16	7B64
@@MUL8	7B4F @@OPEN	78D9 @@PARAM	7624
@@PAUSE	760F @@PEOF	7957 @@POSN	796C
@@PRINT	75C3 @@PRT	74DB @@PUT	74B3
@@RAMDIR	7789 @@RDSEC	781C @@RDSSC	7AE6
@@READ	7981 @@REMOV	78AF @@RENAM	789A
@@REW	7996 @@RMTSK	7720 @@RPTSK	7735
@@RREAD	79AB @@RSLCT	7807 @@RSTOR	77C8
@@RUN	7A53 @@RWRT	79C0 @@SEEK	77F2
@@SEEKSC	79D5 @@SKIP	79EA @@SLCT	77B3
@@STPI	77DD @@TIME	764E @@VDCTL	75FA
@@VER	79FF @@VRSEC	7831 @@WEOF	7A14
@@WHERE	74EF @@WRITE	7A29 @@WRSEC	785B
@@WRSSC	7870 @@WRTRK	7885	

2400 is the transfer address
00000 Total errors

NOTES:

Command: LIST

Library: SYS6/SYS

ISAM # : 41H

```

00100 ;LBLIST/ASM - List Command
0000 00110 TITLE <LIST - LS-DOS 6.2>
00120 ;
0000 00130 *GET SVCMAC:3 ;SVC Macro equivalents
00010 ;SVMAC/ASM - LS-DOS Version VI
00020 *LIST OFF
03900 *LIST ON
0000 00140 *GET VALUES:3 ;Misc. equates
03920 ;VALUES/ASM - Version 6
03930 *LIST OFF
04200 *LIST ON
00150 ;
0006 00160 @PRT EQU 6
0002 00170 @DSP EQU 2
00180 ;
2400 00190 ORG 2400H
00200 ;
00210 START
2400 00220 @@CKBRKC ;Check break
2400 3E6A 00001 LD A,106
2402 EF 00002 RST 40
2403 2804 00230 JR Z,LISTA ;Continue if not
2405 21FFFF 00240 LD HL,-1 ; else abort
2408 C9 00250 RET
00260 ;
00270 ; <BREAK> not hit, Execute Module
00280 ;
00290 LISTA
2409 ED731124 00300 LD (SAVE$P+1),SP ;Save SP
240D CD2D24 00310 CALL LIST ;List a file
2410 310000 00320 SAVESP LD SP,$-$ ;P/u SP address
2413 00330 @@CKBRKC ;Reset if user pressed break
2413 3E6A 00003 LD A,106
2415 EF 00004 RST 40
2416 C9 00340 RET ;Abort
00350 ;
00360 ; I/O Error Processing - Display & Abort
00370 ;
2417 2600 00380 IOERR LD H,0 ;Set HL = Error #
2419 6F 00390 LD L,A ;
241A F6C0 00400 OR C0H ;Short error message
241C 4F 00410 LD C,A ;Stuff error # in C
241D 00420 @@ERROR ;Display error
241D 3E1A 00005 LD A,26
241F EF 00006 RST 40
2420 18EE 00430 JR SAVESP ;Abort
00440 ;
00450 ; Internal Error Message Display Routine
00460 ;
2422 210727 00470 SPCREQ LD HL,SPCREQ$ ;File spec Required
2425 00480 @@LOGOT ;Log error message
00007 IFEQ 00H,1
00008 LD HL,
00009 ENDIF
2425 3E0C 00010 LD A,12
2427 EF 00011 RST 40
2428 21FFFF 00490 ABORT LD HL,-1 ;Internal error = -1
242B 18E3 00500 JR SAVESP ;P/u stack & return
00510 ;
00520 ; LIST - List a file in hex or ASCII
00530 ;

```

```

242D CD9E26 00540 LIST CALL RESKFL ;Reset pause, & enter
00550 ;
00560 ; Find parameter entries if existent
00570 ;
2430 E5 00580 PUSH HL ;Save command ptr
2431 7E 00590 LD A,(HL) ;P/u character
2432 FE28 00600 CP '(' ;Parameter(s) ?
2434 2807 00610 JR Z,GETPRM ;Yes - go get 'em
2436 FE0E 00620 CP CR+1 ;End of line ?
2438 380C 00630 JR C,RESTPTR ;Yes - restore ptr
243A 23 00640 INC HL ;No - bump til end
243B 18F4 00650 JR FPLP ;Do til eol or "("
00660 ;
00670 ; Process any parameters entered
00680 ;
243D 113227 00690 GETPRM LD DE,PRMTBL$ ;DE => Parameter table
2440 00700 @PARAM ;@PARAM
2440 3E11 00012 LD A,17
2442 EF 00013 RST 40
2443 C21724 00710 JP NZ,IOERR ;NZ - "Parameter Error"
2446 E1 00720 RESTPTR POP HL ;Recover ptr
00730 ;
00740 ; Skip command line blanks
00750 ;
2447 7E 00760 IGSPC LD A,(HL) ;P/u character
2448 23 00770 INC HL ;Bump ptr
2449 FE20 00780 CP ' ' ;Space ?
244B 28FA 00790 JR Z,IGSPC ;Yes - ignore them
244D 2B 00800 DEC HL ;HL => First non-space
00810 ;
00820 ; Check if the filespec is legal in format
00830 ;
244E 118527 00840 LD DE,FCB1 ;Fetch file spec
2451 00850 @@FSPEC ;Filespec legal ?
2451 3E4E 00014 LD A,78
2453 EF 00015 RST 40
2454 C22224 00860 JP NZ,SPCREQ ;No - filespec required
00870 ;
00880 ; If this is a device - don't LIST
00890 ;
2457 1A 00900 LD A,(DE) ;Is this a device ?
2458 FE2A 00910 CP '*' ;
245A CA2224 00920 JP Z,SPCREQ ;Yes - Filespec req
00930 ;
00940 ; Save the original filespec
00950 ;
245D EB 00960 EX DE,HL ;HL => Source FCB
245E 11A527 00970 LD DE,FCB2 ;DE => Duplicate FCB
2461 012000 00980 LD BC,32 ;32 bytes tm xfer
2464 E5 00990 PUSH HL ;Save source ptr
2465 EDB0 01000 LDIR ;Xfer
2467 D1 01010 POP DE ;DE => Source FCB
01020 ;
01030 ; Stuff default extension of /TXT to source
01040 ;
2468 212F27 01050 LD HL,TXTEXT ;HL => TXT
246B 01060 @@FEXT ;Stuff TXT if no ext
246B 3E4F 00016 LD A,79
246D EF 00017 RST 40
01070 ;
01080 ; Open the file with LRL of 256

```

```

01090 ;
246E 0600 01100 LD B,0 ;Set LRL (B) = 256
2470 CDF 926 01110 CALL OPEN ;Open file with LRL=256
2473 2816 01120 JR Z,INITLRL ;Good - LIST it
01130 ;
01140 ; Error - Was it a "File not Found" Error ?
01150 ;
2475 FE18 01160 CP 24 ;"File not Found" ?
2477 C21724 01170 JP NZ,IOERR ;No - I/O Error
01180 ;
01190 ; OPEN original filespec instead of TXT
01200 ;
247A 0E20 01210 LD C,32 ;BC = 32 bytes to xfer
247C D5 01220 PUSH DE ;DE => FCB1
247D E5 01230 PUSH HL ;Save I/O buff ptr
247E 21A527 01240 LD HL,FCB2 ;HL => Filespec
2481 EDB0 01250 LDIR ;Xfer original filespec
2483 E1 01260 POP HL ;HL => I/O buffer
2484 D1 01270 POP DE ;DE => unopen FCB
01280 ;
01290 ; Open Original Filespec without extension
01300 ;
2485 CDF 926 01310 CALL OPEN ;Open the file
2488 C21724 01320 JP NZ,IOERR ;I/O error - abort
01330 ;
01340 ; Pick up the DEC from the FCB
01350 ;
248B D5 01360 INITLRL PUSH DE ;Xfer fcb to IX
248C DDE1 01370 POP IX
248E DD4607 01380 LD B,(IX+7) ;P/u DEC
2491 DD4E06 01390 LD C,(IX+6) ; & drive #
2494 01400 @DIRRD ;Read in directory record
2494 3E57 00018 LD A,87
2496 EF 00019 RST 40
01410 ;
01420 ; Was the LRL parm specified ?
01430 ;
2497 3A6427 01440 LD A,(LRESP) ;P/u response
249A B7 01450 OR A
249B 2008 01460 JR NZ,SKIPLRL ;Go if entered
01470 ;
01480 ; No LRL parm, get from directory record
01490 ;
249D 7D 01500 LD A,L ;Position to DIR+4
249E C604 01510 ADD A,4
24A0 6F 01520 LD L,A ;HL => LRL byte (DIR+4)
01530 ;
01540 ; Pick up LRL & stuff into LRL parameter
01550 ;
24A1 7E 01560 LD A,(HL) ;P/u LRL
24A2 329E25 01570 LD (LPARM+1),A ;Set dir LRL into parm
01580 ;
01590 ; Pick up LRL & stuff into FCB
01600 ;
24A5 3A9E25 01610 SKIPLRL LD A,(LPARM+1) ;P/u possible new LRL
24A8 DD7709 01620 LD (IX+9),A ;Put into FCB
24AB DDCB01FE 01630 SET 7,(IX+1) ;Mark for byte I/O
01640 ;
01650 ; Check if TAB (T) parm is flag or numeric
01660 ;
24AF 3A7027 01670 LD A,(TRESP) ;P/u response byte

```

The Source	LIBRARY Files	LIST - LS-DOS 6.2	Page 00004
24B2 E640	01680	AND 40H ;Flag response ?	
24B4 3A3425	01690	LD A,(TPARM+1) ;P/u value	
24B7 2805	01700	JR Z,NOTFLG ;No - numeric	
	01710 ;		
	01720 ;	Flag Response - Is it ON (T=8) or OFF (T=1)	
	01730 ;		
24B9 3C	01740	INC A ;Tab=OFF (NZ) or ON (Z) ?	
24BA 2002	01750	JR NZ,NOTFLG ;Tab = OFF ---> TABEXP=1	
24BC 3E08	01760	LD A,8 ;Tab = ON ---> TABEXP=8	
	01770 ;		
	01780 ;	P/u TAB (T) parm value & stuff into routine	
	01790 ;		
24BE 323E25	01800 NOTFLG	LD (TABEXP+1),A ;Stuff away	
24C1 3D	01810	DEC A ;Range can	
24C2 FE20	01820	CP 32 ;Only be between 1-32.	
24C4 3E2C	01830	LD A,PAR ERR ;Greater - Parm Error	
24C6 D21724	01840	JP NC,IOERR	
	01850 ;		
	01860 ;	Was the P (print) parameter entered ?	
	01870 ;		
24C9 010000	01880 PPARM	LD BC,\$-\$;P/u P parameter	
24CC 78	01890	LD A,B ;Was it specified ?	
24CD B1	01900	OR C	
24CE 2805	01910	JR Z,HPARM ;No - use @DSP	
	01920 ;		
	01930 ;	Stuff @PRT SVC # in output routine	
	01940 ;		
24D0 3E06	01950	LD A,@PRT ;@PRT SVC Number	
24D2 325B26	01960	LD (PUTOUT1+1),A ;Overwrite @DSP	
	01970 ;		
	01980 ;	Hex Parameter Entered ?	
	01990 ;		
24D5 010000	02000 HPARM	LD BC,\$-\$;P/u Hex parm	
24D8 78	02010	LD A,B ;Entered ?	
24D9 B1	02020	OR C	
24DA C27E25	02030	JP NZ,RPARM ;Yes - check out Records	
	02040 ;		
	02050 ;	Routine to LIST a file in ASCII	
	02060 ;		
24DD 327427	02070	LD (BYTCTR),A ;Init counter to 0	
	02080 ;		
24E0 010100	02090 LINPRM	LD BC,1 ;P/u start line parm	
24E3 0B	02100	DEC BC ;Count down for start	
24E4 78	02110	LD A,B ;Ready to list ?	
24E5 B1	02120	OR C	
24E6 2812	02130	JR Z,BGNLIN ;Go list/print if ready	
24E8 118527	02140	LD DE,FCB1 ;DE => FCB	
	02150 ;		
	02160 ;	Ignore all lines until specified start posn	
	02170 ;		
24EB	02180 FNDIST	@GET ;Get a character	
24EB 3E03	00020	LD A,3	
24ED EF	00021	RST 40	
24EE C21724	02190	JP NZ,IOERR ;Error - abort	
24F1 FE0D	02200	CP CR ;End of line?	
24F3 20F6	02210	JR NZ,FNDIST ;Keep reading	
	02220 ;		
	02230 ;	Finished with line - decrement LINE count	
	02240 ;		
24F5 0B	02250	DEC BC ;Dec line counter	
24F6 78	02260	LD A,B ;Finished	

The Source	LIBRARY Files	LIST - LS-DOS 6.2	Page 00005
24F7 B1	02270	OR C	;Ignoring lines ?
24F8 20F1	02280	JR NZ,FNDIST	;No - loop until LINES=0
	02290 ;		
	02300 ;	Start LISTing File	
	02310 ;		
24FA 2AE124	02320 BGNLIN	LD HL,(LINPRM+1)	;P/u original line #
24FD 011A27	02330	LD BC,VARDOT	;Convert to Decimal ASCII
2500 CDB226	02340	CALL CVTDEC	; and stuff into string
	02350 ;		
	02360 ;	Read in a character from the file	
	02370 ;		
2503 118527	02380 GETCHR	LD DE,FCB1	;DE => FCB
2506	02390	@GET	;Get a byte
2506 3E03	00022	LD A,3	
2508 EF	00023	RST 40	
2509 205D	02400	JR NZ,GOTERR	;Go if read error
250B 322125	02410	LD (PUCHAR+1),A	;Hang on to char
	02420 ;		
	02430 ;	Test if NUM parameter was entered	
	02440 ;		
250E 010000	02450 NPARAM	LD BC,\$-\$;P/u N parameter
2511 78	02460	LD A,B	;Was it entered ?
2512 B1	02470	OR C	
2513 280B	02480	JR Z,PUCHAR	;No - don't print line #
	02490 ;		
	02500 ;	N parm entered - print line # & Increment it	
	02510 ;		
2515 211A27	02520	LD HL,VARDOT	;HL => Buffer with line #
2518 E5	02530	PUSH HL	;Save ptr
2519 CD7926	02540	CALL PUTLINE	;Output line @ HL
251C E1	02550	POP HL	;Restore line # ptr
251D CDE826	02560	CALL INCNUM	;Increment line #
	02570 ;		
	02580 ;	Pick up character and Check if high bit set	
	02590 ;		
2520 3E00	02600 PUCHAR	LD A,\$-\$;P/u character
2522 07	02610 DLOOP	RLCA	;Get HB into Carry flag
	02620 ;		
	02630 ;	Reset High bit unless A8 parameter entered	
	02640 ;		
2523 110000	02650 A8PARAM	LD DE,\$-\$;P/u A8 parm
2526 14	02660	INC D	;Was it entered ?
2527 15	02670	DEC D	
2528 2003	02680	JR NZ,A8BIT	;Yes - don't change byte
252A CB3F	02690	SRL A	;Reset Bit 7
252C 1E	02700	DB 1EH	;LD E,n instruction
252D 0F	02710 A8BIT	RRCA	;Use all 8 bits
	02720 ;		
	02730 ;	Is the character a Tab ?	
	02740 ;		
252E F5	02750	PUSH AF	;Save C flag
252F FE09	02760	CP TAB	;Was it a tab ?
2531 201B	02770	JR NZ,NOTTAB	;No - don't check TAB
	02780 ;		
	02790 ;	Character is a Tab - Was T=N specified ?	
	02800 ;		
2533 110800	02810 TPARAM	LD DE,0008H	;P/u TAB parm (Default=8)
2536 1C	02820	INC E	;TAB = N ?
2537 1D	02830	DEC E	
2538 2814	02840	JR Z,NOTTAB	;Yes - don't expand
	02850 ;		

```

02860 ;      P/u column # & calculate # of spaces to pad
02870 ;
253A 3A7427 02880 LD      A,(BYTCTR)      ;P/u column number
253D 0E08    02890 TABEXP LD      C,8      ;P/u TAB expansion #
253F 91      02900 CLOOP SUB      C
2540 30FD    02910 JR      NC,CLOOP
2542 ED44    02920 NEG                      ;Subtract A from Col #
02930 ;
02940 ;      Output A blank spaces for tab expansion
02950 ;
2544 47      02960 LD      B,A      ;Put # spaces in B
2545 3E20    02970 TP1   LD      A,' '    ;Pad with a space
2547 CD5826 02980 CALL    PUTOUT      ;Output character
254A 10F9    02990 DJNZ   TP1      ;B spaces to output
254C 1803    03000 JR      WASTAB      ;Check byte counter
03010 ;
03020 ;      Character was not a tab, display it
03030 ;
254E CD5826 03040 NOTTAB CALL    PUTOUT      ;Print the character
2551 3A7427 03050 WASTAB LD      A,(BYTCTR) ;P/u byte counter
2554 B7      03060 OR      A      ;If C/R printed
2555 CC8626 03070 CALL    Z,CKPAWS      ;Then check for <PAUSE>
03080 ;
03090 ;      Check for <PAUSE> if hi bit set on character
03100 ;
2558 F1      03110 POP     AF      ;Get back char
2559 DC8626 03120 CALL    C,CKPAWS      ;If high bit was set
03130 ;
03140 ;      If character = C/R then read in another line
03150 ;
255C FE0D    03160 CP      CR      ;Was it C/R ?
255E 28A3    03170 JR      Z,GETCHR      ;Yes - get new line
03180 ;
03190 ;      Get another byte from file
03200 ;
2560 118527 03210 LD      DE,FCB1      ;DE => FCB
2563        03220 @GET      ;Get a byte
2563 3E03    00024 LD      A,3
2565 EF      00025 RST      40
2566 28BA    03230 JR      Z,DLOOP      ;Good - check character
03240 ;
03250 ;      I/O Error on @GET - Output a Carriage Return
03260 ;
2568 F5      03270 GOTERR PUSH   AF      ;Save error code
2569 3E0D    03280 LD      A,CR      ;Write end of line
256B CD5826 03290 CALL    PUTOUT
256E F1      03300 POP     AF      ;Rcvr error code
03310 ;
03320 ;      If End of File Error - Exit normally
03330 ;
256F 210000 03340 LD      HL,0      ;Set HL = 0 (normal exit)
2572 FE1C    03350 CP      1CH      ;EOF?
2574 CA1024 03360 GTBK   JP      Z,SAVESP      ;Exit if so
2577 FE1D    03370 CP      1DH      ;NRN > ERN?
2579 28F9    03380 JR      Z,GTBK      ;Yes - leave
257B C31724 03390 JP      IOERR      ;Other - Abort
03400 ;
03410 ;      LIST a file in HEX format
03420 ;
257E 010000 03430 RPARM   LD      BC,$-$      ;P/u starting Record #
2581 118527 03440 LD      DE,FCB1      ;DE => FCB

```

The Source	LIBRARY Files	LIST - LS-DOS 6.2	Page 00007
2584	03450	@@POSN	;Position to Record BC
2584 3E42	00026	LD A,66	
2586 EF	00027	RST 40	
2587 C21724	03460	JP NZ,IOERR	;Abort on position error
	03470 ;		
	03480 ;		
	03490 ;		Reset byte counter to Zero
258A AF	03500 DOHEX	XOR A	;Init byte counter to 0
258B 32A225	03510	LD (DOHEX1+1),A	
	03520 ;		
	03530 ;		Stuff Record Number in Line Number buffer
	03540 ;		
258E ED5B7F25	03550	LD DE,(RPARM+1)	;P/u Record Number
2592 212327	03560	LD HL,VARCLN+1	;HL => Hex ASCII dest
2595	03570	@@HEX16	;Convert DE to ASCII
2595 3E63	00028	LD A,99	
2597 EF	00029	RST 40	
	03580 ;		
	03590 ;		Bump Record Number & stuff into RPARM
	03600 ;		
2598 13	03610	INC DE	;Bump by one
2599 ED537F25	03620	LD (RPARM+1),DE	; & store for next time
	03630 ;		
259D 0100000	03640 LPARM	LD BC,\$-\$;P/u LRL
	03650 ;		
	03660 ;		Convert Byte counter to Hex & stuff in buff
	03670 ;		
25A0 C5	03680 DISBYTE	PUSH BC	;Save bytes left in Rec
25A1 0E00	03690 DOHEX1	LD C,\$-\$;P/u byte counter
25A3 212827	03700	LD HL,VAREQU	;HL => Hex ASCII dest
25A6	03710	@@HEX8	;Cvrt C to ASCII @ HL
25A6 3E62	00030	LD A,98	
25A8 EF	00031	RST 40	
25A9 C1	03720	POP BC	;BC = Bytes left in Rec
	03730 ;		
	03740 ;		Display Record Number/Starting byte string
	03750 ;		
25AA 212227	03760	LD HL,VARCLN	;HL => Display buffer
25AD CD7926	03770	CALL PUTLINE	;Display Rec # byte
	03780 ;		
	03790 ;		P/u byte counter & add 16 (BPL) & stuff away
	03800 ;		
25B0 3AA225	03810	LD A,(DOHEX1+1)	;P/u byte counter
25B3 0610	03820	LD B,16	;Set B = 16 bytes
25B5 80	03830	ADD A,B	;Add 16 to byte count
25B6 32A225	03840	LD (DOHEX1+1),A	; & stuff into LD C inst
	03850 ;		
25B9 217527	03860	LD HL,LINBUF	;HL => Line buffer
	03870 ;		
	03880 ;		Get a byte from the File
	03890 ;		
25BC 118527	03900 DOHEX2	LD DE,FCB1	;DE => FCB
25BF	03910	@@GET	;Get a byte
25BF 3E03	00032	LD A,3	
25C1 EF	00033	RST 40	
25C2 280B	03920	JR Z,DOHEX4	;Good - stuff byte
	03930 ;		
	03940 ;		End of File Error ?
	03950 ;		
25C4 F5	03960	PUSH AF	;Save error code
25C5 FE1C	03970	CP 1CH	;EOF?

The Source	LIBRARY Files	LIST - LS-DOS 6.2	Page 00008
25C7 2805	03980	JR Z,DOHEX3 ;Yes - Okay	
	03990 ;		
	04000 ;	Past End of File Error ?	
	04010 ;		
25C9 FE1D	04020	CP 1DH ;NRN>ERN?	
25CB C21724	04030	JP NZ,IOERR ;Another error	
	04040 ;		
	04050 ;	Recover Flags & check type of Error	
	04060 ;		
25CE F1	04070 DOHEX3	POP AF ;Recover error code	
25CF 2018	04080 DOHEX4	JR NZ,DOHEX5 ;Bypass if at end of file	
	04090 ;		
	04100 ;	Stuff Character in buffer & bump	
	04110 ;		
25D1 77	04120	LD (HL),A ;Stuff byte in buffer	
25D2 23	04130	INC HL ;Bump ptr	
	04140 ;		
	04150 ;	Output byte in Hex & follow with a space	
	04160 ;		
25D3 CD4726	04170	CALL CVTHEX ;Output the byte in hex	
25D6 3E20	04180	LD A,' ' ; followed by a space	
25D8 CD5826	04190	CALL PUTOUT	
	04200 ;		
	04210 ;	Output an extra space if halfway in line	
	04220 ;		
25DB 78	04230	LD A,B ;P/u byte counter	
25DC FE09	04240	CP 9 ;Halfway yet?	
25DE CC7526	04250	CALL Z,WR1SPA ;Yes - display space	
	04260 ;		
	04270 ;	Dec Chars/Line & # of chars left in Rec	
	04280 ;		
25E1 05	04290	DEC B ;Count down	
25E2 0D	04300	DEC C ;Count down the LRL	
25E3 2804	04310	JR Z,DOHEX5 ;Done - get next record	
	04320 ;		
	04330 ;	Finished with Line ?	
	04340 ;		
25E5 78	04350	LD A,B ;Done with line ?	
25E6 B7	04360	OR A	
25E7 20D3	04370	JR NZ,DOHEX2 ;No - do til 16 chars	
	04380 ;		
	04390 ;	Finished with Line or Logical Record	
	04400 ;		
25E9 F5	04410 DOHEX5	PUSH AF ;End the line	
25EA 78	04420	LD A,B ;P/u byte counter	
25EB FE10	04430	CP 16 ;Done with this line ?	
25ED 2826	04440	JR Z,PRTLIN2 ;Yes - get another record	
	04450 ;		
	04460 ;	Display ASCII equivalent of line	
	04470 ;		
25EF C5	04480 PRTLIN	PUSH BC ;Save counters	
	04490 ;		
	04500 ;	Multiply # of chars not printed by three	
	04510 ;		
25F0 78	04520	LD A,B ;P/u # chars not printed	
25F1 87	04530	ADD A,A ;X 2	
25F2 80	04540	ADD A,B ;X 3	
25F3 47	04550	LD B,A ;Stuff in B	
	04560 ;		
	04570 ;	Add two extra spaces if more than halfway	
	04580 ;		

```

25F4 FE1B 04590 CP 27 ;Need extra space if
25F6 3F 04600 CCF ;Not halfway
25F7 3E01 04610 LD A,1 ;Plus 1 more space
25F9 88 04620 ADC A,B ;Carry set for 3 spaces
04630 ;
04640 ; Position to ASCII portion of line
04650 ;
25FA 47 04660 LD B,A ;Set loop counter
25FB CD6F26 04670 CALL WRSPA ;Output B spaces
25FE C1 04680 POP BC ;Recover the counters
04690 ;
04700 ; Calculate # of characters to print
04710 ;
25FF 3E10 04720 LD A,16 ;Get # to print
2601 90 04730 SUB B
2602 47 04740 LD B,A ;Xfer to B for DJNZ
2603 217527 04750 LD HL,LINBUF ;HL => Line buffer
2606 C5 04760 PUSH BC ;Save C
2607 0E08 04770 LD C,8 ;Space after 8
04780 ;
04790 ; Display ASCII part of line
04800 ;
2609 7E 04810 PRTLIN1 LD A,(HL) ;P/u character
260A 23 04820 INC HL ;Bump ptr
260B CD3B26 04830 CALL CVTDOT ;Output each char
260E 0D 04840 DEC C ;Space yet ?
260F CC7526 04850 CALL Z,WR1SPA ;Yes - display space
2612 10F5 04860 DJNZ PRTLIN1 ;Output line
2614 C1 04870 POP BC ;Recover C
04880 ;
04890 ; End of Line - Output C/R & check for EOF
04900 ;
2615 3E0D 04910 PRTLIN2 LD A,CR ;Output C/R
2617 CD5826 04920 CALL PUTOUT
261A F1 04930 POP AF ;Recover @GET ret code
261B 3E1C 04940 LD A,1CH ;Init to EOF
261D C26825 04950 JP NZ,GOTERR ;End of file - Abort
04960 ;
2620 CD8626 04970 CALL CKPAWS ;<PAUSE> or <BREAK> ?
04980 ;
04990 ; Are we done with the Record ?
05000 ;
2623 79 05010 LD A,C ;P/u # of bytes left
2624 B7 05020 OR A ;Finished ?
2625 C2A025 05030 JP NZ,DISBYTE ;No - get next line
05040 ;
05050 ; Finished with record - Output space & C/R
05060 ;
2628 3E20 05070 LD A,' ' ;Space
262A CD5826 05080 CALL PUTOUT
262D 3E0D 05090 LD A,CR ;Carriage Return
262F CD5826 05100 CALL PUTOUT
05110 ;
05120 ; Increment Line Number
05130 ;
2632 212227 05140 LD HL,VARCLN ;HL => Line Number
2635 CDE826 05150 CALL INCNUM ;Increment
2638 C38A25 05160 JP DOHEX ;Loop for more
05170 ;
05180 ; CVTDOT - Output chars & convert non-printables
05190 ;

```

The Source	LIBRARY Files	LIST - LS-DOS 6.2	Page 00010
263B FE20	05200 CVTDOT	CP ' ' ;Don't print controls	
263D 3804	05210	JR C,CVTDOT1	
263F FE7F	05220	CP 7FH ;Print X'20' thru X'7E'	
2641 3815	05230	JR C,PUTOUT	
2643 3E2E	05240 CVTDOT1	LD A,'.' ;Otherwise change to "."	
2645 1811	05250	JR PUTOUT	
	05260 ;		
	05270 ;	CVTHEX - Convert A to Hex ASCII & Output it	
	05280 ;		
2647 F5	05290 CVTHEX	PUSH AF ;Save Lower digit	
2648 0F	05300	RRCA ;Get most sig nibble	
2649 0F	05310	RRCA	
264A 0F	05320	RRCA	
264B 0F	05330	RRCA	
264C CD5026	05340	CALL CVTH1 ;Output upper nibble	
264F F1	05350	POP AF ;Recover #	
	05360 ;		
2650 E60F	05370 CVTH1	AND 0FH ;Mask off upper nibble	
2652 C690	05380	ADD A,90H ;Convert to Hex digit	
2654 27	05390	DAA	
2655 CE40	05400	ADC A,40H	
2657 27	05410	DAA ;Fall into Output byte	
	05420 ;		
	05430 ;	PUTOUT - Put out a byte to *DO/*PR	
	05440 ;		
2658 C5	05450 PUTOUT	PUSH BC ;Save BC	
2659 4F	05460	LD C,A ;Xfer char to C	
	05470 ;		
	05480 ;	Output byte to *DO or *PR	
	05490 ;		
265A 3E02	05500 PUTOUT1	LD A,@DSP ;@DSP or @PRT SVC #	
265C EF	05510	RST 28H	
265D C21724	05520	JP NZ,IOERR ;NZ - I/O Error	
	05530 ;		
	05540 ;	Increment byte counter	
	05550 ;		
2660 E5	05560	PUSH HL ;Save HL	
2661 217427	05570	LD HL,BYTCTR ;HL => Byte counter	
2664 34	05580	INC (HL) ;Bump counter	
2665 79	05590	LD A,C ;P/u byte	
2666 FE0D	05600	CP CR ;End of line ?	
2668 2002	05610	JR NZ,NOTCR ;No - rest regs & RETURN	
	05620 ;		
266A 3600	05630	LD (HL),0 ;Reset byte counter	
266C E1	05640 NOTCR	POP HL ;Restore registers	
266D C1	05650	POP BC	
266E C9	05660	RET ; & RETURN	
	05670 ;		
	05680 ;		
	05690 ;	Output B spaces to Display or Printer	
	05700 ;		
266F CD7526	05710 WRSPA	CALL WR1SPA ;Write a space	
2672 10FB	05720	DJNZ WRSPA ;Do it B times	
2674 C9	05730	RET ;RETURN	
	05740 ;		
	05750 ;	Output a space to Display or Printer	
	05760 ;		
2675 3E20	05770 WR1SPA	LD A,' ' ;Space Character	
2677 18DF	05780	JR PUTOUT ;Output byte	
	05790 ;		
	05800 ;	PUTLINE - Output a line to the video or printer	

```

05810 ;          HL => Line of data to output
05820 ;
2679 7E 05830 PUTLINE LD      A,(HL)          ;P/u byte
267A 23 05840         INC      HL              ;Prepare for next
267B FE03 05850         CP       ETX           ;Check if done none
267D C8 05860         RET      Z              ; return if so
267E CD5826 05870        CALL    PUTOUT        ;Char OK output it
2681 FE0D 05880         CP       CR            ;Check for CR
2683 C8 05890         RET      Z              ; return if so
2684 18F3 05900        JR       PUTLINE
05910 ;
05920 ;          CKPAWS - Check for Pause (SHIFT @)
05930 ;
2686 05940 CKPAWS  @@FLAGS          ;IY => System flags
2686 3E65 00034        LD       A,101
2688 EF 00035        RST       40
05950 ;
05960 ;          Was the <BREAK> key pressed ?
05970 ;
2689 05980        @@CKBRKC          ;<BREAK> hit ?
2689 3E6A 00036        LD       A,106
268B EF 00037        RST       40
268C 201C 05990        JR       NZ,GOTBRK      ;Quit if so
06000 ;
06010 ;          Was the <SHIFT><@> pressed ?
06020 ;
268E FDCB0A4E 06030       BIT      1,(IY+KFLAG$) ;Is the <PAUSE> bit set ?
2692 C8 06040       RET      Z              ;Return if not <SHIFT><@>
06050 ;
06060 ;          Pause - Wait for key to continue
06070 ;
2693 06080 CKWAIT  @@KEY              ;Wait for key press
2693 3E01 00038        LD       A,1
2695 EF 00039        RST       40
2696 FE60 06090 CKWAIT1 CP       60H          ;Was key a <SHIFT @>?
2698 28F9 06100        JR       Z,CKWAIT      ;Ignore if it was
269A FE80 06110        CP       80H          ;Was key a Break?
269C 280C 06120        JR       Z,GOTBRK      ;Quit if so
06130 ;
06140 ;          Reset <PAUSE> & <ENTER> bits & RETURN
06150 ;
269E 06160 RESKFL  @@FLAGS          ;IY => Flag Table
269E 3E65 00040        LD       A,101
26A0 EF 00041        RST       40
26A1 FD7E0A 06170       LD       A,(IY+KFLAG$) ;P/u KFLAG$
26A4 E6F9 06180       AND       0F9H          ;Reset Pause and Enter
26A6 FD770A 06190       LD       (IY+KFLAG$),A ;Stuff into KFLAG$
26A9 C9 06200       RET
06210 ;
06220 ;          <BREAK> hit - Display C/R & Abort
06230 ;
26AA 3E0D 06240 GOTBRK LD      A,CR          ;Send end of line
26AC CD5826 06250       CALL    PUTOUT        ;Output byte
26AF C32824 06260       JP      ABORT         ; and abort due to BREAK
06270 ;
06280 ;          CVTDEC - Convert HL to Decimal & stuff in BC
06290 ;
26B2 111027 06300 CVTDEC LD      DE,100000    ;Divide by 100000
26B5 CDCD26 06310       CALL    CVD1
26B8 11E803 06320       LD      DE,1000      ;Divide by 1000
26BB CDCD26 06330       CALL    CVD1

```

The Source	LIBRARY Files	LIST - LS-DOS 6.2	Page 00012
26BE 116400	06340	LD DE,100 ;Divide by 100	
26C1 CDCD26	06350	CALL CVD1	
26C4 110A00	06360	LD DE,10 ;Divide by 10	
26C7 CDCD26	06370	CALL CVD1	
26CA 110100	06380	LD DE,1 ;Divide by 1	
	06390 ;		
	06400 ;	Divide Quotient in HL by value in DE	
	06410 ;		
26CD AF	06420 CVD1	XOR A ;Clear carry set A=0	
26CE ED52	06430 CVD2	SBC HL,DE ;Subtract Divisor	
26D0 3803	06440	JR C,CVD3 ;Done - add back divisor	
26D2 3C	06450	INC A ;Bump counter	
26D3 18F9	06460	JR CVD2 ;Subtract until a Carry	
	06470 ;		
	06480 ;	Add divisor to neg rem & cvrt A to ASCII	
	06490 ;		
26D5 19	06500 CVD3	ADD HL,DE ;HL = New quotient	
26D6 C630	06510	ADD A,'0' ;A = ASCII numeric digit	
26D8 FE30	06520	CP '0' ;Zero ?	
26DA 2009	06530	JR NZ,CVD4 ;No - stuff in buff (BC)	
	06540 ;		
	06550 ;	Char is a Zero - use space if leading zero	
	06560 ;		
26DC 0B	06570	DEC BC ;Backspace buff ptr	
26DD 0A	06580	LD A,(BC) ;P/u last char	
26DE 03	06590	INC BC ;Bump to current	
26DF FE20	06600	CP ' ' ;Last char a space ?	
26E1 2802	06610	JR Z,CVD4 ;Yes - don't use lead 0	
26E3 3E30	06620	LD A,'0' ;No - use zero	
	06630 ;		
	06640 ;	Stuff Numeric ASCII character into buffer	
	06650 ;		
26E5 02	06660 CVD4	LD (BC),A ;Stuff char in buff	
26E6 03	06670	INC BC ;Bump ptr	
26E7 C9	06680	RET ;RETurn	
	06690 ;		
	06700 ;	INCNUM - Increment Line number in buffer (HL)	
	06710 ;		
26E8 23	06720 INCNUM	INC HL ;Point to lo-order digit	
26E9 23	06730	INC HL	
26EA 23	06740	INC HL	
26EB 23	06750	INC HL	
	06760 ;		
	06770 ;	Loop to Increment digit and return if done	
	06780 ;		
26EC 7E	06790 INCNUM1	LD A,(HL) ;P/u digit	
26ED F630	06800	OR '0' ;Start with possible 0	
26EF 3C	06810	INC A ;Add 1	
26F0 77	06820	LD (HL),A ;Restuff	
26F1 D63A	06830	SUB '9'+1 ;See if went to 10	
26F3 D8	06840	RET C ;Ret if not	
26F4 3630	06850	LD (HL),'0' ;Make it 0	
26F6 2B	06860	DEC HL ;B/u one	
26F7 18F3	06870	JR INCNUM1 ; & loop	
	06880 ;		
	06890 ;	OPEN - Open a file	
	06900 ;		
26F9	06910 OPEN	@@FLAGS ;IY => System Flags	
26F9 3E65	00042	LD A,101	
26FB EF	00043	RST 40	
26FC FDCB12C6	06920	SET 0,(IY+SFLAG\$) ;P/u SFLAG\$	


```

2700 210028 06930 LD HL,IOBUFF ;HL => I/O buffer
2703 06940 @@OPEN ;Open the file
2703 3E3B 00044 LD A,59
2705 EF 00045 RST 40
2706 C9 06950 RET ; & RETurn with condition
06960 ;
2707 46 06970 SPCREQ$ DB 'File spec required',CR
69 6C 65 20 73 70 65 63
20 72 65 71 75 69 72 65
64 0D
06980 ;
06990 ;
271A 20 07000 VARDOT DB ' . ',ETX
20 20 20 20 2E 20 03
2722 20 07010 VARCLN DB ' : '
20 20 20 20 3A
2728 58 07020 VAREQU DB 'XX = . ',ETX
58 20 3D 20 20 03
272F 54 07030 TXTEXT DB 'TXT'
58 54
07040 ;
07050 ;
07060 ; PARAMETER TABLE
07070 ;
2732 80 07080 PRMTBL$ DB 80H
07090 ;
07100 ; ASCII8 (A8) - Flag input only
07110 ;
2733 46 07120 DB FLAG!6
2734 41 07130 DB 'ASCII8'
53 43 49 49 38
273A 00 07140 DB 0
273B 2425 07150 DW A8PARAM+1
07160 ;
273D 42 07170 DB FLAG!2
273E 41 07180 DB 'A8'
38
2740 00 07190 DB 0
2741 2425 07200 DW A8PARAM+1
07210 ;
07220 ; LINE - Numeric input only
07230 ;
2743 84 07240 DB NUM!4
2744 4C 07250 DB 'LINE'
49 4E 45
2748 00 07260 DB 0
2749 E124 07270 DW LINPRM+1
07280 ;
07290 ; NUM (N) - Flag input only
07300 ;
274B 53 07310 DB FLAG!ABB!3
274C 4E 07320 DB 'NUM'
55 4D
274F 00 07330 DB 0
2750 0F25 07340 DW NPARAM+1
07350 ;
07360 ; HEX (H) - Flag input only
07370 ;
2752 53 07380 DB FLAG!ABB!3
2753 48 07390 DB 'HEX'
45 58

```

```

2756 00      07400      DB      0
2757 D624    07410      DW      HPARM+1
           07420 ;
           07430 ;      REC (R) - Numeric input only
           07440 ;
2759 93      07450      DB      NUM!ABB!3
275A 52      07460      DB      'REC'
           45 43
275D 00      07470      DB      0
275E 7F25    07480      DW      RPARM+1
           07490 ;
           07500 ;      LRL (L) - Numeric input only
           07510 ;
2760 93      07520      DB      NUM!ABB!3
2761 4C      07530      DB      'LRL'
           52 4C
2764 00      07540 LRESP DB      0
2765 9E25    07550      DW      LPARM+1
           07560 ;
           07570 ;      P - Flag input only
           07580 ;
2767 41      07590      DB      FLAG!1
2768 50      07600      DB      'P'
2769 00      07610      DB      0
276A CA24    07620      DW      PPARM+1
           07630 ;
           07640 ;      TAB (T) - Flag input only
           07650 ;
276C 53      07660      DB      FLAG!ABB!3
276D 54      07670      DB      'TAB'
           41 42
2770 00      07680 TRESP DB      0
2771 3425    07690      DW      TPARM+1
           07700 ;
2773 00      07710      DB      0
           07720 ;
           07730 ;Buffer Area
           07740 ;
0001      07750 BYTCTR DS      1
0010      07760 LINBUF DS      16
0020      07770 FCB1  DS      32
0020      07780 FCB2  DS      32
           07790 ;
2800      07800      ORG      $<-8+1<+8
0100      07810 IOBUFF DS      256
           07820 ;
2400      07830      END      START

```

@@1	0000	@@2	0000	@@3	0000
@@4	0000	@DSP	0002	@MOD2	0000
@MOD4	FFFF	@PRT	0006	A8BIT	252D
A8PARM	2523	ABB	0010	ABORT	2428
AP	0027	BGNLIN	24FA	BREAK	0080
BS	0008	BYTCTR	2774	CFLAG\$	0002
CKPAWS	2686	CKWAIT	2693	CKWAIT1	2696
CLOOP	253F	CR	000D	CVD1	26CD
CVD2	26CE	CVD3	26D5	CVD4	26E5
CVTDEC	26B2	CVTDOT	263B	CVTDOT1	2643
CVTH1	2650	CVTHEX	2647	DFLAG\$	0003
DISBYTE	25A0	DLOOP	2522	DOHEX	258A
DOHEX1	25A1	DOHEX2	25BC	DOHEX3	25CE
DOHEX4	25CF	DOHEX5	25E9	ETX	0003
FCB1	2785	FCB2	27A5	FLAG	0040
FNDIST	24EB	FPLP	2431	GETCHR	2503
GETPRM	243D	GOTBRK	26AA	GOTERR	2568
GTBK	2574	HPARM	24D5	IGSPC	2447
INCNUM	26E8	INCNUM1	26EC	INITLRL	248B
IOBUFF	2800	IOERR	2417	KFLAG\$	000A
LF	000A	LINBUF	2775	LINPRM	24E0
LIST	242D	LISTA	2409	LPARM	259D
LRESP	2764	NOTCR	266C	NOTFLG	24BE
NOTTAB	254E	NPARM	250E	NUM	0080
OPEN	26F9	PAR_ERR	002C	PPARM	24C9
PRMTBL\$	2732	PRTLIN	25EF	PRTLIN1	2609
PRTLIN2	2615	PUCHAR	2520	PUTLINE	2679
PUTOUT	2658	PUTOUT1	265A	RESKFL	269E
RESTPTR	2446	RPARM	257E	SAVESP	2410
SFLAG\$	0012	SKIPLRL	24A5	SPCREQ	2422
SPCREQ\$	2707	START	2400	STR	0020
TAB	0009	TABEXP	253D	TP1	2545
TPARM	2533	TRESP	2770	TXTEXT	272F
VARCLN	2722	VARDOT	271A	VAREQU	2728
VFLAG\$	0015	WASTAB	2551	WR1SPA	2675
WRSPA	266F	@@ABORT	AF 29	@@ADTSK	AFBC
@@BANK	B4D4	@@BKSP	B1B4	@@BREAK	B4EA
@@CHNIO	AF 14	@@CKBRKC	B538	@@CKDRV	B010
@@CKEOF	B1C9	@@CKTSK	AFA7	@@CLOSE	B19F
@@CLS	B522	@@CMNDI	AF 53	@@CMNDR	AF 68
@@CTL	AD78	@@DATE	AEEA	@@DCSTAT	B04F
@@DEBUG	AF 92	@@DECHEX	B454	@@DIRRD	B3C1
@@DIRWR	B3D6	@@DIV16	B43F	@@DIV8	B42A
@@DODIR	B025	@@DSP	AD3C	@@DSPLY	ADDC
@@ERROR	AF 7D	@@EXIT	AF 3E	@@FEXT	B32E
@@FLAGS	B4BE	@@FNAME	B343	@@FSPEC	B319
@@GATRD	B3AC	@@GATWR	B3EB	@@GET	AD50
@@GTDCB	B36D	@@GTDCT	B358	@@GTMOD	B382
@@HDFMT	B0F7	@@HEX16	B493	@@HEX8	B47E
@@HEXDEC	B469	@@HIGH\$	B4A8	@@INIT	B175
@@KBD	ADB4	@@KEY	AD28	@@KEYIN	ADC8
@@KLTSK	AFFB	@@LOAD	B2EF	@@LOC	B1DE
@@LOF	B1F3	@@LOGGER	AE13	@@LOGOT	AE28
@@MSG	AE5F	@@MUL16	B415	@@MUL8	B400
@@OPEN	B18A	@@PARAM	AED5	@@PAUSE	AEC0
@@PEOF	B208	@@POSN	B21D	@@PRINT	AE74
@@PRT	AD8C	@@PUT	AD64	@@RAMDIR	B03A
@@RDSEC	B0CD	@@RDSSC	B397	@@READ	B232
@@REMOV	B160	@@RENAM	B14B	@@REW	B247
@@RMTSK	AFD1	@@RPTSK	AFE6	@@RREAD	B25C

@@RSLCT	B0B8 @@RSTOR	B079 @@RUN	B304
@@RWGIT	B271 @@SEEK	B0A3 @@SEEKSC	B286
@@SKIP	B29B @@SLCT	B064 @@STPI	B08E
@@TIME	AEFF @@VDCTL	AEAB @@VER	B2B0
@@VRSEC	B0E2 @@WEOF	B2C5 @@WHERE	ADA0
@@WRITE	B2DA @@WRSEC	B10C @@WRSSC	B121
@@WRTRK	B136		

2400 is the transfer address
00000 Total errors

NOTES:

Command: LOAD, RUN

Library: SYS6/SYS

ISAM # : 81H, 82H

```

00100 ;LBLOAD/ASM - LOAD & RUN Commands
0000 00110 TITLE <LOAD/RUN - LS-DOS 6.2>
00120 ;
000D 00130 CR EQU 13
004D 00140 @RUN EQU 77
0028 00150 RST28 EQU 28H
0000 00160 *GET SVCMAC:3 ;SVC Macro equivalents
00010 ;SVC MAC/ASM - LS-DOS Version VI
00020 *LIST OFF
03900 *LIST ON
00170 ;
2400 00180 ORG 2400H
00190 ;
00200 ; RUN entry point
00210 ;
2400 C32924 00220 RUN JP RUN0 ;RUN entry point
00230 ;
00240 ; LOAD entry point
00250 ;
2403 CDC924 00260 LOAD CALL COMMON ;Parse parms & filespec
2406 203F 00270 JR NZ,IOERR
2408 3AFC24 00280 LD A,(XPARAM+1) ;If not (X), go to it
240B B7 00290 OR A
240C 200A 00300 JR NZ,NEEDPR
240E 119425 00310 LD DE,FCB ;Pt to fcb
2411 00320 @@LOAD
2411 3E4C 00001 LD A,76
2413 EF 00002 RST 40
2414 2031 00330 JR NZ,IOERR ;Go on error
2416 1844 00340 JR EXIT ; or just exit
00350 ;
00360 ; Need to prompt for the system disk
00370 ;
2418 CD0F25 00380 NEEDPR CALL LOADIT ;Load the file
241B 202A 00390 JR NZ,IOERR ;Exit if error
241D 210000 00400 LD HL,0 ;Init no error
00410 ;
00420 ; Get the system disk back in system drive
00430 ;
2420 E5 00440 PMTSYS PUSH HL ;Save cmd line ptr
2421 213A25 00450 LD HL,PMTSYS$
2424 CD6324 00460 CALL FLASH
2427 E1 00470 POP HL ;Rcvr cmd line ptr
2428 C9 00480 RET
00490 ;
00500 ; RUN command entry
00510 ;
2429 CDC924 00520 RUN0 CALL COMMON ;Parse common args
242C 2019 00530 JR NZ,IOERR ;Go on error
242E 3AFC24 00540 LD A,(XPARAM+1)
2431 B7 00550 OR A
2432 119425 00560 LD DE,FCB
2435 2005 00570 JR NZ,RUN1 ;Prompt if (X)
2437 3E4D 00580 LD A,@RUN ;RUN SVC number
2439 C32800 00590 JP RST28
00600 ;
243C E5 00610 RUN1 PUSH HL ;Save cmd line ptr
243D 119425 00620 LD DE,FCB
2440 CD0F25 00630 CALL LOADIT
2443 E3 00640 EX (SP),HL ;Get cmd ptr & save ept
2444 28DA 00650 JR Z,PMTSYS ;Run if prog OK or

```

```

2446 E1      00660      POP      HL      ; pop TRAADR & error
            00670      ;
            00680      ;      Error handling
            00690      ;
2447 FE3F    00700      IOERR    CP      63      ;If extended error
2449 280A    00710      JR      Z,EXTERR      ; handle by @LOGOT
244B 6F      00720      LD      L,A      ;Put error # into HL
244C 2600    00730      LD      H,0
244E F6C0    00740      OR      0C0H      ;Set short error and ret
2450 4F      00750      LD      C,A
2451         00760      @@ERROR
2451 3E1A    00003      LD      A,26
2453 EF      00004      RST      40
2454 C9      00770      RET
2455         00780      EXTERR  @@LOGOT
            00005      IFEQ      00H,1
            00006      LD      HL,
            00007      ENDIF
2455 3E0C    00008      LD      A,12
2457 EF      00009      RST      40
2458 21FFFF  00790      LD      HL,-1
245B C9      00800      RET
245C 210000  00810      EXIT      LD      HL,0
245F C9      00820      RET
            00830      ;
            00840      ;      Flash the prompt & await reply
            00850      ;
2460 CDB924  00860      FLASH0   CALL      RESKFLG      ;Reset 3-bit field
2463 01FD41  00870      FLASH    LD      BC,16893      ;Delay for 250 ms
2466         00880      @@PAUSE
2466 3E10    00010      LD      A,16
2468 EF      00011      RST      40
2469 FD7E0A  00890      LD      A,(IY+'K'-'A')
246C E605    00900      AND      4!1      ;Wait for no ENTER!BRK
246E 20F0    00910      JR      NZ,FLASH0
2470 CDB924  00920      CALL      RESKFLG      ;Reset in case BREAK
2473         00930      FLS1      @@DSPLY      ;Display the message
            00012      IFEQ      00H,1
            00013      LD      HL,
            00014      ENDIF
2473 3E0A    00015      LD      A,10
2475 EF      00016      RST      40
2476 C24724  00940      JP      NZ,IOERR      ;Abort on error
2479 010040  00950      LD      BC,4000H
247C CD9024  00960      CALL      FLS2      ;Blink start
247F 2042    00970      JR      NZ,GOTBRK      ;Handle BREAK
2481 0E1E    00980      LD      C,1EH      ;Cursor erase to EOL
2483 CD2025  00990      CALL      DSP
2486 013333  01000      LD      BC,3333H      ;Wait
2489 CD9024  01010      CALL      FLS2      ;Wait & ck enter
248C 2035    01020      JR      NZ,GOTBRK      ;Handle BREAK
248E 18E3    01030      JR      FLS1      ;Loop until ENTER
            01040      ;
            01050      ;      FLS2 - Delay a while & ck on <BREAK/ENTER>
            01060      ;
2490         01070      FLS2      @@CKBRKC      ;<BREAK> hit ?
2490 3E6A    00017      LD      A,106
2492 EF      00018      RST      40
2493 2808    01080      JR      Z,CKENT      ;No - check <ENTER>
2495 0E1E    01090      LD      C,1EH      ;Erase Line
2497 CD2025  01100      CALL      DSP      ;Output byte

```



```

249A AF      01110      XOR      A      ;Set NZ
249B 3C      01120      INC      A      ;
249C C9      01130      RET              ;And RETurn
          01140 ;
249D FDCB0A56 01150 CKENT  BIT      2,(IY+'K'-'A') ;Ck ENTER bit
24A1 2006     01160      JR      NZ,FLS4 ;Go on ENTER down
24A3 0B      01170      DEC      BC     ;Count down
24A4 78      01180      LD      A,B
24A5 B1      01190      OR      C
24A6 20E8     01200      JR      NZ,FLS2
24A8 C9      01210      RET              ;Return with Z-flag
          01220 ;
          01230 ; ENTER condition found
          01240 ;
24A9 F1      01250 FLS4   POP      AF     ;Pop return code
24AA         01260 FLS5   @@KBD         ;Clear type ahead buffer
24AA 3E08     00019      LD      A,8
24AC EF      00020      RST      40
24AD 28FB     01270      JR      Z,FLS5   ;Loop if have character
24AF 0E1E     01280      LD      C,1EH   ;Wipe line
24B1 CD2025   01290      CALL     DSP
24B4 0E0E     01300      LD      C,14    ;Cursor on
24B6 CD2025   01310      CALL     DSP
24B9 FD7E0A   01320 RESKFLG LD      A,(IY+'K'-'A') ;Reset 3-bit field
24BC E6F8     01330      AND      0F8H
24BE FD770A   01340      LD      (IY+'K'-'A'),A
24C1 AF      01350      XOR      A      ;Set Z-flag
24C2 C9      01360      RET
          01370 ;
24C3 217825   01380 GOTBRK LD      HL,STOP$ ;Point to error message
24C6 3E3F     01390      LD      A,63    ;Init extended error
24C8 C9      01400      RET              ; & return NZ
          01410 ;
          01420 ; Common initialization routine
          01430 ;
24C9 118A25   01440 COMMON LD      DE,PRMTBL ;Parm of X?
24CC         01450      @PARAM
24CC 3E11     00021      LD      A,17
24CE EF      00022      RST      40
24CF C0      01460      RET      NZ     ;Ret with error code
24D0         01470 COMM1  @@FLAGS      ;Get flag table pointer
24D0 3E65     00023      LD      A,101
24D2 EF      00024      RST      40
24D3 7E      01480 COMM1A LD      A,(HL) ;Skip past spaces
24D4 FE20     01490      CP
24D6 2003     01500      JR      NZ,COMM2
24D8 23      01510      INC      HL
24D9 18F8     01520      JR      COMM1A
24DB 119425   01530 COMM2 LD      DE,FCB ;Get filespec
24DE         01540      @@FSPEC
24DE 3E4E     00025      LD      A,78
24E0 EF      00026      RST      40
24E1 2005     01550      JR      NZ,COMM3 ;Go on error
24E3 1A      01560      LD      A,(DE) ;Device specs not allowed
24E4 FE2A     01570      CP      '*'
24E6 2007     01580      JR      NZ,COMM4 ;Go if OK
24E8 212725   01590 COMM3 LD      HL,SPCREQ$ ;Point to error message
24EB 3E3F     01600      LD      A,63    ;Init extended error
24ED B7      01610      OR      A      ;Set NZ condition
24EE C9      01620      RET
          01630 ;

```

The Source	LIBRARY Files	LOAD/RUN - LS-DOS 6.2	Page 00004
24EF E5	01640 COMM4	PUSH HL	;Save cmdline ptr
24F0 219125	01650	LD HL,CMDEXT	;Default to CMD
24F3	01660	@@FEXT	
24F3 3E4F	00027	LD A,79	
24F5 EF	00028	RST 40	
24F6 CD0C25	01670	CALL GOSYS2	;Get SYS2 for open
24F9 E1	01680	POP HL	;Pop the INBUF\$ pointer
24FA C0	01690	RET NZ	
24FB 110000	01700 XPARM	LD DE,0	;Ck on X parm
24FE 7A	01710	LD A,D	
24FF B3	01720	OR E	
2500 C8	01730	RET Z	;Back on no (X)
2501 E5	01740	PUSH HL	;Save pointer
2502 215925	01750	LD HL,PMTSRC\$;Init prompt
2505 CD6324	01760	CALL FLASH	;Prompt for source disk
2508 D1	01770	POP DE	;Pointer to DE
2509 C0	01780	RET NZ	;Back on error in HL
250A EB	01790	EX DE,HL	;If no error, pointer
250B C9	01800	RET	; back to HL
	01810 ;		
	01820 ;	Call SYS2 for open routine	
	01830 ;		
250C 3E84	01840 GOSYS2	LD A,84H	;Load sys2
250E EF	01850	RST 28H	
	01860 ;		
	01870 ;	Loading routine	
	01880 ;		
250F 119425	01890 LOADIT	LD DE,FCB	
2512 FDCB12D6	01900	SET 2,(IY+'S'-'A')	;Turn on RUN flag
2516	01910	@@LOAD	;Load the file
2516 3E4C	00029	LD A,76	
2518 EF	00030	RST 40	
2519 C8	01920	RET Z	
251A F5	01930	PUSH AF	;Save error ret code
251B CD2024	01940	CALL PMTSYS	;Get system disk back
251E F1	01950	POP AF	;Rcvr error ret code
251F C9	01960	RET	
	01970 ;		
2520	01980 DSP	@@DSP	;Display byte
2520 3E02	00031	LD A,2	
2522 EF	00032	RST 40	
2523 C8	01990	RET Z	;Return if OK
2524 C34724	02000	JP IOERR	
	02010 ;		
	02020 ;		
2527 46	02030 SPCREQ\$ DB	'File spec required',CR	
69 6C 65	20 73 70 65 63		
20 72 65	71 75 69 72 65		
64 0D			
253A 0F	02040 PMTSYS\$ DB	15,29,30,'Insert SYSTEM disk <ENTER>',29,3	
1D 1E 49	6E 73 65 72 74		
20 53 59	53 54 45 4D 20		
64 69 73	6B 20 3C 45 4E		
54 45 52	3E 1D 03		
2559 0F	02050 PMTSRC\$ DB	15,29,30,'Insert SOURCE disk <ENTER>',29,3	
1D 1E 49	6E 73 65 72 74		
20 53 4F	55 52 43 45 20		
64 69 73	6B 20 3C 45 4E		
54 45 52	3E 1D 03		
2578 0E	02060 STOP\$ DB	14,29,'Command aborted',CR	
1D 43 6F	6D 6D 61 6E 64		

```
20 61 62 6F 72 74 65 64
0D
02070 ;
258A 80 02080 PRMTBL DB 80H,41H,'X',0
41 58 00
258E FC24 02090 DW XPARAM+1
2590 00 02100 NOP
02110 ;
2591 43 02120 CMDEXT DB 'CMD'
4D 44
2594 00 02130 FCB DB 0
001F 02140 DS 31
02150 ;
2403 02160 END LOAD
```

@@1	0000 @@2	0000 @@3	0000
@@4	0000 @MOD2	0000 @MOD4	FFFF
@RUN	004D CKENT	249D CMDEXT	2591
COMM1	24D0 COMM1A	24D3 COMM2	24DB
COMM3	24E8 COMM4	24EF COMMON	24C9
CR	000D DSP	2520 EXIT	245C
EXTERR	2455 FCB	2594 FLASH	2463
FLASH0	2460 FLS1	2473 FLS2	2490
FLS4	24A9 FLS5	24AA GOSYS2	250C
GOTBRK	24C3 IOERR	2447 LOAD	2403
LOADIT	250F NEEDPR	2418 PMTSRC\$	2559
PMTSYS	2420 PMTSYS\$	253A PRMTBL	258A
RESKFLG	24B9 RST28	0028 RUN	2400
RUN0	2429 RUN1	243C SPCREQ\$	2527
STOP\$	2578 XPARAM	24FB @@ABORT	7D1F
@@ADTSK	7DB2 @@BANK	82CA @@BKSP	7FAA
@@BREAK	82E0 @@CHNIO	7D0A @@CKBRKC	832E
@@CKDRV	7E06 @@CKEOF	7FBF @@CKTSK	7D9D
@@CLOSE	7F95 @@CLS	8318 @@CMNDI	7D49
@@CMNDR	7D5E @@CTL	7B6E @@DATE	7CE0
@@DCSTAT	7E45 @@DEBUG	7D88 @@DECHEX	824A
@@DIRRD	81B7 @@DIRWR	81CC @@DIV16	8235
@@DIV8	8220 @@DODIR	7E1B @@DSP	7B32
@@DSPLY	7BD2 @@ERROR	7D73 @@EXIT	7D34
@@FEXT	8124 @@FLAGS	82B4 @@FNAME	8139
@@FSPEC	810F @@GATRD	81A2 @@GATWR	81E1
@@GET	7B46 @@GTDCB	8163 @@GTDCCT	814E
@@GTMOD	8178 @@HDFMT	7EED @@HEX16	8289
@@HEX8	8274 @@HEXDEC	825F @@HIGH\$	829E
@@INIT	7F6B @@KBD	7BAA @@KEY	7B1E
@@KEYIN	7BBE @@KLTSK	7DF1 @@LOAD	80E5
@@LOC	7FD4 @@LOF	7FE9 @@LOGGER	7C09
@@LOGOT	7C1E @@MSG	7C55 @@MUL16	820B
@@MUL8	81F6 @@OPEN	7F80 @@PARAM	7CCB
@@PAUSE	7CB6 @@PEOF	7FFE @@POSN	8013
@@PRINT	7C6A @@PRT	7B82 @@PUT	7B5A
@@RAMDIR	7E30 @@RDSEC	7EC3 @@RDSSC	818D
@@READ	8028 @@REMOV	7F56 @@RENAM	7F41
@@REW	803D @@RMTSK	7DC7 @@RPTSK	7DDC
@@RREAD	8052 @@RSLCT	7EAE @@RSTOR	7E6F
@@RUN	80FA @@RWRIT	8067 @@SEEK	7E99
@@SEEKSC	807C @@SKIP	8091 @@SLCT	7E5A
@@STEPI	7E84 @@TIME	7CF5 @@VDCTL	7CA1
@@VER	80A6 @@VRSEC	7ED8 @@WEOF	80BB
@@WHERE	7B96 @@WRITE	80D0 @@WRSEC	7F02
@@WRSSC	7F17 @@WRTRK	7F2C	

2403 is the transfer address

00000 Total errors

NOTES:

Command: MEMORY

Library: SYS6/SYS

ISAM # : 1EH

```

00100 ;LBMEMORY/ASM - MEMORY Command
0000 00110 TITLE <MEMORY - LS-DOS 6.2>
00120 ;
0000 00130 *GET SVCMAC:3 ;SVC Macro equivalents
00010 ;SVC MAC/ASM - LS-DOS Version VI
00020 *LIST OFF
03900 *LIST ON
0000 00140 *GET VALUES:3 ;Misc. equates
03920 ;VALUES/ASM - Version 6
03930 *LIST OFF
04200 *LIST ON
00150 ;
002C 00160 PAR_ERR EQU 44 ;Parameter Error
2600 00170 LOWEST EQU 2600H
00180 ;
2400 00190 ORG 2400H
00200 ;
2400 00210 MEMORY @@CKBRKC ;Break key down?
2400 3E6A 00001 LD A,106
2402 EF 00002 RST 40
2403 2804 00220 JR Z,BEGINA ;Ok if not
2405 21FFFF 00230 LD HL,-1 ; else abort
2408 C9 00240 RET
00250 ;
2409 ED73F725 00260 BEGINA LD (EXIT+1),SP ;Save SP Address
00270 ;
00280 ; Process any Parameters entered
00290 ;
240D 112326 00300 LD DE,PRMTBL$ ;DE => Parameter Table
2410 D5 00310 PUSH DE ;Save Parm table start
2411 00320 @@PARAM ;Process parameters
2411 3E11 00003 LD A,17
2413 EF 00004 RST 40
2414 E1 00330 POP HL ;HL => Parm table start
2415 C2B525 00340 PERR JP NZ,PRMERR ;NZ - Parameter Error
00350 ;
00360 ; Legal input - were the entries acceptable ?
00370 ;
2418 CDC325 00380 GPARMS CALL CKPARAM ;Valid entries ?
241B 20F8 00390 JR NZ,PERR ;No - Parameter error
00400 ;
241D 00410 @@FLAGS ;IY => System Flags
241D 3E65 00005 LD A,101
241F EF 00006 RST 40
00420 ;
00430 ; Was the CLEAR (C) parameter entered ?
00440 ;
2420 0100FF 00450 CPARAM LD BC,0FF00H ;C = Clear byte
2423 3A5626 00460 LD A,(CRESP) ;P/u response
2426 B7 00470 OR A ;Any response ?
2427 283C 00480 JR Z,NEXTPRM ;No - get next parm
2429 FDCB0246 00490 BIT 0,(IY+CFLAG$) ;If memory frozen,
242D C2E925 00500 JP NZ,NOMEM ; can't do it
00510 ;
00520 ; CLEAR (C) parm entered - is this a flag ?
00530 ;
2430 CB77 00540 BIT 6,A ;Is this a flag ?
2432 2805 00550 JR Z,ISNUMER ;No - check if numeric
00560 ;
00570 ; Response is a FLAG - Is it NO, OFF, or N ?
00580 ;

```

The Source	LIBRARY Files	MEMORY - LS-DOS 6.2	Page 00002
2434 04	00590	INC B ;Yes or No ?	
2435 202E	00600	JR NZ,NEXTPRM ;No - get next parm	
2437 180F	00610	JR FILLMEM ;Yes - Fill mem with 0's	
	00620 ;		
	00630 ;	Response is not a FLAG, check if numeric	
	00640 ;		
2439 215E24	00650	ISNUMER LD HL,FILLBYT+1 ;HL => Byte to Fill	
243C CB7F	00660	BIT 7,A ;Numeric response ?	
243E 71	00670	LD (HL),C ;Stuff byte in LD (HL),nn	
243F 2007	00680	JR NZ,FILLMEM ;Fill mem with char	
	00690 ;		
	00700 ;	Response must be a string - Is length = 1 ?	
	00710 ;		
2441 E60F	00720	AND 00001111B ;P/u length	
2443 3D	00730	DEC A ;Better be one	
2444 20CF	00740	JR NZ,PERR ;Not - parameter error	
	00750 ;		
	00760 ;	Pick up character at address & stuff away	
	00770 ;		
2446 0A	00780	LD A,(BC) ;P/u character to fill	
2447 77	00790	LD (HL),A ;Stuff in LD (HL),nn	
	00800 ;		
	00810 ;	Set HL => HIGH\$ (if DOS) or LOW\$ (@CMNDR)	
	00820 ;		
2448 CDA025	00830	FILLMEM CALL GETHILO ;HIGH\$ (HL), LOW\$ (DE)	
244B FDCB024E	00840	BIT 1,(IY+CFLAG\$) ;Executing @CMNDR ?	
244F 2801	00850	JR Z,USEHIGH ;No - use HIGH\$	
2451 EB	00860	EX DE,HL ;@CMNDR - use LOW\$	
2452 110026	00870	USEHIGH LD DE,MEMORY+2000H ;Start clearing here	
	00880 ;		
	00890 ;	Calculate amount of memory to fill	
	00900 ;		
2455 AF	00910	XOR A ;Clear carry	
2456 ED52	00920	SBC HL,DE ;Get # to fill	
2458 44	00930	LD B,H ;Xfer to BC	
2459 4D	00940	LD C,L	
	00950 ;		
	00960 ;	Fill user area - HIGH\$/LOW\$ with spec'd byte	
	00970 ;		
245A 62	00980	LD H,D ;HL => LOW\$	
245B 6B	00990	LD L,E	
245C 13	01000	INC DE ;DE => LOW\$ + 1	
245D 3600	01010	FILLBYT LD (HL),\$\$;Stuff in fill byte	
245F EDB0	01020	LDIR ;Fill memory	
2461 210000	01030	GOODEX LD HL,0 ;Good Exit	
2464 C9	01040	RET	
	01050 ;		
	01060 ;	Was the ADDR (A) parameter specified ?	
	01070 ;		
2465 3A3726	01080	NEXTPRM LD A,(ARESP) ;P/u Address response	
2468 B7	01090	OR A ;Specified ?	
2469 CA0C25	01100	JP Z,HICHECK ;No - get next parm	
	01110 ;		
	01120 ;	Check for NUMERIC entry	
	01130 ;		
246C CB7F	01140	BIT 7,A ;Check NUM bit	
246E 201C	01150	JR NZ,APARM ; go if set	
	01160 ;		
	01170 ;	Response must be a string - Is length = 1 ?	
	01180 ;		
2470 E60F	01190	AND 00001111B ;P/u length	


```

2472 3D      01200      DEC      A      ;Better be one
2473 20A0    01210      JR      NZ,PERR ;Not - parameter error
          01220 ;
          01230 ;      P/u character representation of 'FLAG'
          01240 ;
2475 2A8D24  01250      LD      HL,(APARM+1) ;P/u pointer to char
2478 7E      01260      LD      A,(HL)      ;P/u char
2479 CBAF    01270      RES      5,A      ; and make UC
247B D641    01280      SUB      41H      ;Normalize for flag#
247D FE1A    01290      CP      26      ;Check on range
247F D2ED25  01300      JP      NC,RANGER ;Error if too high
2482 FDE5    01310      PUSH     IY
2484 E1      01320      POP      HL      ;P/u base of flags
2485 1600    01330      LD      D,0
2487 5F      01340      LD      E,A      ;Put offset in DE
2488 19      01350      ADD      HL,DE    ;Add them together
2489 228D24  01360      LD      (APARM+1),HL ;Set value in response word
          01370 ;
          01380 ;      P/u addr, cvrt to Hex ASCII & put in string
          01390 ;
248C 110000  01400 APARM LD      DE,$-$      ;P/u the address
248F 217A26  01410      LD      HL,HEXADD ;HL => Destination
2492      01420      @@HEX16 ;Cvrt DE to ASCII @ HL
2492 3E63    00007      LD      A,99
2494 EF      00008      RST      40
          01430 ;
          01440 ;      Convert DE to Decimal ASCII & put in string
          01450 ;
2495 EB      01460      EX      DE,HL      ;Set HL = Address
2496 E5      01470      PUSH     HL      ;Save addr ptr
2497 118226  01480      LD      DE,DECADD ;DE => Destination
249A      01490      @@HEXDEC ;Cvrt HL to ASCII @ DE
249A 3E61    00009      LD      A,97
249C EF      00010      RST      40
249D E1      01500      POP      HL      ;HL = Address
          01510 ;
          01520 ;      P/u word & byte at that address
          01530 ;
249E E5      01540      PUSH     HL      ;Save address ptr
249F 4E      01550      LD      C,(HL)    ;P/u byte
24A0 51      01560      LD      D,C      ;P/u the word value
24A1 23      01570      INC      HL      ; & stuff into the
24A2 5E      01580      LD      E,(HL)    ; message in hex
          01590 ;
          01600 ;      Convert Byte to Hex ASCII & stuff in string
          01610 ;
24A3 21A026  01620      LD      HL,OLDBYTE ;HL => Destination
24A6      01630      @@HEX8 ;Convert C to ASCII @ HL
24A6 3E62    00011      LD      A,98
24A8 EF      00012      RST      40
          01640 ;
          01650 ;      Convert Word to Hex ASCII & stuff in string
          01660 ;
24A9 218C26  01670      LD      HL,OLDWORD ;HL => Destination
24AC      01680      @@HEX16 ;Cvt DE to ASCII @ HL
24AC 3E63    00013      LD      A,99
24AE EF      00014      RST      40
24AF DDE1    01690      POP      IX      ;Recover Address ptr
          01700 ;
          01710 ;      Was WORD or BYTE parameter entered ?
          01720 ;

```

```

24B1 214726 01730 LD HL,BRESP ;HL => Byte Response
24B4 113F26 01740 LD DE,WRESP ;DE => Word Response
24B7 1A 01750 LD A,(DE) ;Anything entered ?
24B8 B6 01760 OR (HL)
24B9 2010 01770 JR NZ,WHICH ;Yes - which one
01780 ;
01790 ; Neither Entered - Modify string
01800 ;
24BB 219126 01810 LD HL,OLDWORD+5 ;End string
24BE 3629 01820 LD (HL),' ' ;Don't display new word
24C0 23 01830 INC HL
24C1 3620 01840 LD (HL),' '
24C3 23 01850 INC HL
24C4 3620 01860 LD (HL),' '
24C6 23 01870 INC HL
24C7 3603 01880 LD (HL),ETX
24C9 182C 01890 JR DSPSTR ;Display string
01900 ;
01910 ; One or Both was entered - ensure both not
01920 ;
24CB 1A 01930 WHICH LD A,(DE) ;Word entered ?
24CC B7 01940 OR A
24CD 2816 01950 JR Z,BPARM ;No - get byte
01960 ;
01970 ; Word Entered - Make sure byte wasn't entered
01980 ;
24CF 7E 01990 LD A,(HL) ;Entered ?
24D0 B7 02000 OR A
24D1 C2B525 02010 JP NZ,PRMERR ;Yes - parameter error
02020 ;
02030 ; Pick up Word Value & Stuff into Memory
02040 ;
24D4 110000 02050 WPARM LD DE,$-$ ;P/u word
24D7 DD7200 02060 LD (IX),D ;Stuff lsb
24DA DD7301 02070 LD (IX+1),E ;Stuff msb
02080 ;
02090 ; Cvt Word/Byte to Hex ASCII & put in string
02100 ;
24DD 219726 02110 STUFVAL LD HL,NEWWORD ;HL => Destination
24E0 02120 @HEX16 ;Cvrt DE to hex in (HL)
24E0 3E63 00015 LD A,99
24E2 EF 00016 RST 40
24E3 1812 02130 JR DSPSTR ;Display String
02140 ;
02150 ; Stuff byte into mem if between 0-255
02160 ;
24E5 11FF00 02170 BPARM LD DE,00FFH ;P/u byte
24E8 7A 02180 LD A,D ;Hi-order must = 0
24E9 B7 02190 OR A
24EA C2B525 02200 JP NZ,PRMERR ;No - Parm error
24ED DD7300 02210 LD (IX),E ;Stuff LSB into string
02220 ;
02230 ; Convert byte to Hex ASCII & stuff in string
02240 ;
24F0 4B 02250 LD C,E ;Set C = new byte
24F1 21A926 02260 LD HL,NEWBYTE ;HL => New byte
24F4 02270 @HEX8 ;Cvt C to ASCII @ HL
24F4 3E62 00017 LD A,98
24F6 EF 00018 RST 40
02280 ;
02290 ; Display address string

```

```

023000 ;
24F7 217826 02310 DSPSTR LD HL,ADDMSG ;HL => Message
24FA CDB025 02320 CALL DSPLY ;Display it
02330 ;
02340 ; Display Word/Byte string
02350 ;
24FD 218C26 02360 LD HL,OLDWORD ;HL => "Word string"
2500 3A4726 02370 LD A,(BRESP) ;Byte response
2503 B7 02380 OR A ;Specified
2504 2803 02390 JR Z,DSPSTR2 ;No - display Word string
2506 21A026 02400 LD HL,OLDBYTE ;Yes - use Byte string
2509 CDB025 02410 DSPSTR2 CALL DSPLY ;Display string
02420 ;
02430 ; HIGH$ & LOW$ check - Was HIGH$ entered ?
02440 ;
250C CDA025 02450 HICHECK CALL GETHILO ;HIGH$ (HL), LOW$ (DE)
250F 22E125 02460 LD (OLDHI),HL ;Put old HIGH$ in header
2512 3A2926 02470 LD A,(HRESP) ;HIGH$ = value ?
2515 B7 02480 OR A
2516 281E 02490 JR Z,LOCHECK ;No - check LOW$
2518 FDCB0246 02500 BIT 0,(IY+CFLAG$) ;If memory frozen,
251C C2E925 02510 JP NZ,NOMEM ; can't do it
02520 ;
02530 ; HIGH$ entered, p/u value & check range
02540 ;
251F 010000 02550 HPARM LD BC,$-$ ;P/u requested new HIGH$
2522 ED42 02560 SBC HL,BC ;New HIGH$ > old HIGH$ ?
2524 DAED25 02570 JP C,RANGER ;Yes - Range error
02580 ;
02590 ; Create Header String & establish true HIGH$
02600 ;
2527 D5 02610 PUSH DE ;Save DE
2528 50 02620 LD D,B ;Set DE = HIGH$ + 1
2529 59 02630 LD E,C
252A 13 02640 INC DE
252B 21E525 02650 LD HL,HD_ADD ;Convert DE to Hex ASCII
252E 02660 @@HEX16 ; at HL (BC is preserved)
252E 3E63 00019 LD A,99
2530 EF 00020 RST 40
2531 21F6FF 02670 LD HL,-HLEN ;Set HL = actual HIGH$
2534 09 02680 ADD HL,BC ; including header size
2535 D1 02690 POP DE ;Restore LOW$
02700 ;
02710 ; Was LOW$ entered ?
02720 ;
2536 3A3026 02730 LOCHECK LD A,(LRESP) ;P/u response
2539 B7 02740 OR A ;Entered ?
253A 2816 02750 JR Z,CHKBOTH ;No - check range
253C FDCB0246 02760 BIT 0,(IY+CFLAG$) ;If memory frozen,
2540 C2E925 02770 JP NZ,NOMEM ; can't do it
02780 ;
02790 ; LOW$ entered - cannot be below LOWEST
02800 ;
2543 110000 02810 LPARM LD DE,$-$ ;P/u value entered
2546 010026 02820 LD BC,LOWEST ;BC = lowest possible
2549 EB 02830 EX DE,HL ; memory location
254A E5 02840 PUSH HL
254B ED42 02850 SBC HL,BC ;In range ?
254D E1 02860 POP HL
254E EB 02870 EX DE,HL
254F DAED25 02880 JP C,RANGER ;No - display range error

```

```

02890 ;
02900 ;      HL = HIGH$, DE = LOW$ - do they overlap ?
02910 ;
2552 E5 02920 CHKBOTH PUSH    HL      ;HIGH$ must be
2553 B7 02930          OR      A      ; greater than or equal
2554 ED52 02940          SBC     HL,DE  ; to LOW$.
2556 E1 02950          POP     HL
2557 DAED25 02960          JP     C,RANGER ;Less - range error
02970 ;
02980 ;      LOW$ and HIGH$ are both valid - set LOW$
02990 ;
255A EB 03000          EX      DE,HL    ;Pt DE => HI$, HL => LOW$
255B 0601 03010          LD      B,1
255D 03020          @@HIGH$          ;Set LOW$
255D 3E64 03021          LD      A,100
255F EF 03022          RST     40
03030 ;
03040 ;      Was the HIGH parameter specified ?
03050 ;
2560 3A2926 03060          LD      A,(HRESP) ;Was HIGH specified ?
2563 B7 03070          OR      A
2564 2813 03080          JR      Z,DSPHI ;No - don't alter it
03090 ;
03100 ;      Yes - Change Exit message to include header
03110 ;
2566 3E0A 03120          LD      A,LF      ;Change C/R to a L/F
2568 32B026 03130          LD      (HEADMES),A
03140 ;
03150 ;      Xfer header into high memory
03160 ;
256B D5 03170          PUSH    DE      ;Save HIGH$ ptr
256C 13 03180          INC     DE      ;Pt to header destination
256D 21DF25 03190          LD      HL,HEADER ;HL => Header
2570 010A00 03200          LD      BC,HLEN ;BC = header length
2573 EDB0 03210          LDIR      ;Xfer to mem
2575 E1 03220          POP     HL      ;HL => HIGH$ (BC = 0)
2576 03230          @@HIGH$          ;Set HIGH$ (func. 0)
2576 3E64 03233          LD      A,100
2578 EF 03234          RST     40
03240 ;
03250 ;      P/u HIGH$/LOW$, cvt to Hex, & put in string
03260 ;
2579 CDA025 03270 DSPHI CALL    GETHILO ;HIGH$ (HL), LOW$ (DE)
257C E5 03280          PUSH    HL      ;Save HIGH$ ptr
257D 217226 03290          LD      HL,LOWIS1 ;HL => Destination
2580 03300          @@HEX16          ;Cvt DE to ASCII @ HL
2580 3E63 03305          LD      A,99
2582 EF 03306          RST     40
2583 D1 03310          POP     DE      ;DE = HIGH$
2584 216326 03320          LD      HL,HIGHIS1 ;HL => Destination
2587 03330          @@HEX16          ;Cvt DE to ASCII @ HL
2587 3E63 03327          LD      A,99
2589 EF 03328          RST     40
03340 ;
03350 ;      Display HIGH$ = nnnn & LOW$ = nnnn string
03360 ;
258A 215A26 03370          LD      HL,HIGHIS ;HL => HIGH$/LOW$ string
258D 03380          @@LOGOT          ;Log & display
03380          IFEQ     00H,1
03390          LD      HL,
033A0          ENDIF

```

```

258D 3E0C      00032      LD      A,12
258F EF        00033      RST      40
2590 21B026    03390      LD      HL,HEADMES      ;Display Header message
2593           03400      @@LOGOT      ; if one was inserted
                00034      IFEQ     00H,1
                00035      LD      HL,
                00036      ENDIF
2593 3E0C      00037      LD      A,12
2595 EF        00038      RST      40
                03410      ;
                03420      ;      Was a Go Parameter Entered ?
                03430      ;
2596 3A4D26    03440      GO      LD      A,(GRESP)      ;GO entered ?
2599 B7         03450      OR      A
259A CA6124    03460      JP      Z,GOODEX      ;No - RETurn HL = 0
259D C30000    03470      GPARM   JP      $-$      ;Yes - Jump to address
                03480      ;
                03490      ;      GETHILO - Get HIGH$ in HL, & LOW$ in DE
                03500      ;
25A0 210000    03510      GETHILO LD      HL,0      ;P/u LOW$
25A3 54         03520      LD      D,H      ;Set DE = 0
25A4 5D         03530      LD      E,L
25A5 0601      03540      LD      B,1
25A7           03550      @@HIGH$
25A7 3E64      00039      LD      A,100
25A9 EF        00040      RST      40
25AA EB        03560      EX      DE,HL      ;Save in DE
25AB 45        03570      LD      B,L      ;Set B = 0
25AC           03580      @@HIGH$      ;Get HIGH$ & RETurn
25AC 3E64      00041      LD      A,100
25AE EF        00042      RST      40
25AF C9        03590      RET              ;RETurn
                03600      ;
                03610      ;      DSPLY - Display a line to the video
                03620      ;
25B0           03630      DSPLY   @@DSPLY      ;Display line
                00043      IFEQ     00H,1
                00044      LD      HL,
                00045      ENDIF
                00046      LD      A,10
25B0 3E0A      00047      RST      40
25B2 EF        00047      RST      40
25B3 C8        03640      RET      Z      ;Z - RETurn
25B4 21        03650      DB      21H      ;Skip LD A,## instruction
                03660      ;
                03670      ;      IOERR - Fatal Error Handler
                03680      ;
25B5 3E2C      03690      PRMERR LD      A,PAR_ERR      ;Parameter Error
25B7 6F        03700      IOERR  LD      L,A      ;Error # to HL
25B8 2600      03710      LD      H,0
25BA F6C0      03720      OR      0C0H      ;Short error message
25BC 4F        03730      LD      C,A      ;Save in C
25BD           03740      @@ERROR      ;Display error
25BD 3E1A      00048      LD      A,26
25BF EF        00049      RST      40
25C0 1834      03750      JR      EXIT      ;Go to exit routine
                03760      ;
                03770      ;      CKPARAM - Check if Parameter types are legal
                03780      ;      HL => Beginning of Parameter Table
                03790      ;      Z <= Set if Parameters entered were legal
                03800      ;
25C2 D1        03810      GOODPRM POP      DE      ;Clear stack

```

The Source	LIBRARY Files	MEMORY - LS-DOS 6.2	Page 00008
25C3 23	03820 CKPARM	INC HL ;Bump past 80H	
25C4 7E	03830	LD A,(HL) ;P/u type byte	
25C5 47	03840	LD B,A ;Save in B	
25C6 E60F	03850	AND 0FH ;Get length	
25C8 C8	03860	RET Z ;Zero - finished	
	03870 ;		
	03880 ;	Position HL to response byte	
	03890 ;		
25C9 23	03900	INC HL ;HL => Parameter Name	
25CA E5	03910	PUSH HL ;Save start of name	
25CB 5F	03920	LD E,A ;Set DE = name	
25CC 1600	03930	LD D,0 ; length.	
25CE 19	03940	ADD HL,DE	
	03950 ;		
	03960 ;	Pick up response, mask off junk, & xfer in D	
	03970 ;		
25CF 7E	03980	LD A,(HL) ;P/u response	
25D0 E6E0	03990	AND 0E0H ;Bits 7-5 = response	
25D2 57	04000	LD D,A ;Save in D	
	04010 ;		
	04020 ;	Was the response bit acceptable by type ?	
	04030 ;		
25D3 78	04040	LD A,B ;P/u type byte	
25D4 A2	04050	AND D ;Mask off bits 4-0	
25D5 AA	04060	XOR D ;Result = Z if both set	
	04070 ;		
	04080 ;	Position HL to next parameter entry	
	04090 ;		
25D6 23	04100	INC HL ;Go past word	
25D7 23	04110	INC HL	
25D8 28E8	04120	JR Z,GOODPRM ;Z - good entry	
	04130 ;		
	04140 ;	Illegal Entry - Recover Name start & RETURN	
	04150 ;		
25DA 4B	04160	LD C,E ;Set BC = length	
25DB 0600	04170	LD B,0	
25DD E1	04180	POP HL ;HL => Parameter Name	
25DE C9	04190	RET ;RETURN NZ	
	04200 ;		
	04210 ;	6.2 Memory Header	
	04220 ;		
25DF 1808	04230 HEADER	JR MEMSTRT ;JR to start of module	
25E1 0000	04240 OLDHI	DW 0 ;HIGH\$ before this module	
25E3 05	04250	DB 5,'&' ;Use "&" to denote addr.	
	26		
25E5 6E	04260 HD_ADD	DB 'nnnn' ;Hex ASCII address	
6E 6E 6E			
25E9	04270 MEMSTRT	EQU \$;Length byte	
000A	04280 HLEN	EQU \$-HEADER ;Length of Header	
	04290 ;		
	04300 ;		
	04310 ;	Messages	
	04320 ;		
	04330 ;		
25E9 21FD25	04340 NOMEM	LD HL,NOMEM\$	
25EC DD	04350	DB 0DDH	
25ED 211726	04360 RANGER	LD HL,RANGER\$	
	04370 ;		
	04380 ;	Display & Log message, & RETURN	
	04390 ;		
25F0	04400	@@LOGOT ;Display/Log message	

```

00050 IFEQ 00H,1
00051 LD HL,
00052 ENDIF
25F0 3E0C 00053 LD A,12
25F2 EF 00054 RST 40
25F3 21FFFF 04410 ABORT LD HL,-1 ;Internal Error
25F6 310000 04420 EXIT LD SP,$-$ ;P/u old SP address
25F9 00055 @CKBRKC ;Clear any Break
25FB 3E6A 00056 LD A,106
25FC C9 00056 RST 40
04440 RET ;RETurn
04450 ;
25FD 4E 04460 NOMEM$ DB 'No memory space available',CR
6F 20 6D 65 6D 6F 72 79
20 73 70 61 63 65 20 61
76 61 69 6C 61 62 6C 65
0D
2617 52 04470 RANGER$ DB 'Range error',CR
61 6E 67 65 20 65 72 72
6F 72 0D
04480 ;
04490 ; PARAMETER TABLE
04500 ;
2623 80 04510 PRMTBL$ DB 80H ;6.x type @PARAM
04520 ;
04530 ; HIGH (H) - Accept Numeric input only
04540 ;
2624 94 04550 DB NUM!ABB!4
2625 48 04560 DB 'HIGH'
49 47 48
2629 00 04570 HRESP DB 0
262A 2025 04580 DW HPARM+1
04590 ;
04600 ; LOW (L) - Accept numeric input only
04610 ;
262C 93 04620 DB NUM!ABB!3
262D 4C 04630 DB 'LOW'
4F 57
2630 00 04640 LRESP DB 0
2631 4425 04650 DW LPARM+1
04660 ;
04670 ; ADD (A) - Accept numeric, string input
04680 ;
2633 B3 04690 DB NUM!STR!ABB!3
2634 41 04700 DB 'ADD'
44 44
2637 00 04710 ARESP DB 0
2638 8D24 04720 DW APARM+1
04730 ;
04740 ; WORD (W) - Accept Numeric input only
04750 ;
263A 94 04760 DB NUM!ABB!4
263B 57 04770 DB 'WORD'
4F 52 44
263F 00 04780 WRESP DB 0
2640 D524 04790 DW WPARM+1
04800 ;
04810 ; BYTE (B) - Accept Numeric input only
04820 ;
2642 94 04830 DB NUM!ABB!4
2643 42 04840 DB 'BYTE'

```

```

59 54 45
2647 00 04850 BRESP DB 0
2648 E624 04860 DW BPARAM+1
      04870 ;
      04880 ; GO (G) - Accept Numeric input only
      04890 ;
264A 92 04900 DB NUM!ABB!2
264B 47 04910 DB 'GO'
      4F
264D 00 04920 GRESP DB 0
264E 9E25 04930 DW GPARAM+1
      04940 ;
      04950 ; CLEAR (C) - Accept Flag, Numeric or string input
      04960 ;
2650 F5 04970 DB FLAG!NUM!STR!ABB!5
2651 43 04980 DB 'CLEAR'
      4C 45 41 52
2656 00 04990 CRESP DB 0
2657 2124 05000 DW CPARAM+1
      05010 ;
2659 00 05020 NOP
      05030 ;
265A 48 05040 HIGHIS DB 'High = X',AP
      69 67 68 20 3D 20 58 27
2663 78 05050 HIGHIS1 DB 'xxxx',AP,' Low = X',AP
      78 78 78 27 20 20 4C 6F
      77 20 3D 20 58 27
2672 78 05060 LOWIS1 DB 'xxxx',AP,ETX
      78 78 78 27 03
      05070 ;
2678 58 05080 ADDMSG DB 'X',AP
      27
267A 6E 05090 HEXADD DB 'nnnn',AP,' = '
      6E 6E 6E 27 20 3D 20
2682 64 05100 DECADD DB 'dddd (X',AP,ETX
      64 64 64 64 20 28 58 27
      03
      05110 ;
268C 6E 05120 OLDWORD DB 'nnnn',AP,' => X',AP
      6E 6E 6E 27 20 3D 3E 20
      58 27
2697 6E 05130 NEWWORD DB 'nnnn',AP,' ) ',ETX
      6E 6E 6E 27 29 20 20 03
      05140 ;
26A0 6E 05150 OLDBYTE DB 'nn',AP,' => X',AP
      6E 27 20 3D 3E 20 58 27
26A9 6E 05160 NEWBYTE DB 'nn',AP,' ) ',ETX
      6E 27 29 20 20 03
      05170 ;
26B0 0D 05180 HEADMES DB CR,'Note : Memory Header Inserted',CR
      4E 6F 74 65 20 3A 20 4D
      65 6D 6F 72 79 20 48 65
      61 64 65 72 20 49 6E 73
      65 72 74 65 64 0D
      05190 ;
2400 05200 END MEMORY

```


@@1	0000 @@2	0000 @@3	0000
@@4	0000 @MOD2	0000 @MOD4	FFFF
ABB	0010 ABORT	25F3 ADDMSG	2678
AP	0027 APARM	248C ARESP	2637
BEGINA	2409 BPARM	24E5 BREAK	0080
BRESP	2647 BS	0008 CFLAG\$	0002
CHKBOTH	2552 CKPARM	25C3 CPARM	2420
CR	000D CRESP	2656 DECADD	2682
DFLAG\$	0003 DSPHI	2579 DSPLY	25B0
DSPSTR	24F7 DSPSTR2	2509 ETX	0003
EXIT	25F6 FILLBYT	245D FILLMEM	2448
FLAG	0040 GDPARMS	2418 GETHILO	25A0
GO	2596 GOODEX	2461 GOODPRM	25C2
GPARM	259D GRESP	264D HD_ADD	25E5
HEADER	25DF HEADMES	26B0 HEXADD	267A
HICHECK	250C HIGHIS	265A HIGHIS1	2663
HLEN	000A HPARM	251F HRESP	2629
IOERR	25B7 ISNUMER	2439 KFLAG\$	000A
LF	000A LOCHECK	2536 LOWEST	2600
LOWIS1	2672 LPARM	2543 LRESP	2630
MEMORY	2400 MEMSTRT	25E9 NEWBYTE	26A9
NEWWORD	2697 NEXTPRM	2465 NOMEM	25E9
NOMEM\$	25FD NUM	0080 OLDBYTE	26A0
OLDHI	25E1 OLDWORD	268C PAR_ERR	002C
PERR	2415 PRMERR	25B5 PRMTBL\$	2623
RANGER	25ED RANGER\$	2617 SFLAG\$	0012
STR	0020 STUFVAL	24DD TAB	0009
USEHIGH	2452 VFLAG\$	0015 WHICH	24CB
WPARM	24D4 WRESP	263F @@ABORT	99FB
@@ADTSK	9A8E @BANK	9FA6 @BKSPP	9C86
@@BREAK	9FBC @CHNIO	99E6 @CKBRKC	A00A
@@CKDRV	9AE2 @CKEOF	9C9B @CKTSK	9A79
@@CLOSE	9C71 @CLS	9FF4 @CMNDI	9A25
@@CMNDR	9A3A @CTL	984A @DATE	99BC
@@DCSTAT	9B21 @DEBUG	9A64 @DECHEX	9F26
@@DIRRD	9E93 @DIRWR	9EA8 @DIV16	9F11
@@DIV8	9EFC @DODIR	9AF7 @DSP	980E
@@DSPLY	98AE @ERROR	9A4F @EXIT	9A10
@@FEXT	9E00 @FLAGS	9F90 @FNAME	9E15
@@FSPEC	9DEB @GATRD	9E7E @GATWR	9EBD
@@GET	9822 @GTDCB	9E3F @GTDCT	9E2A
@@GTMOD	9E54 @HDFMT	9BC9 @HEX16	9F65
@@HEX8	9F50 @HEXDEC	9F3B @HIGH\$	9F7A
@@INIT	9C47 @KBD	9886 @KEY	97FA
@@KEYIN	989A @KLTSK	9ACD @LOAD	9DC1
@@LOC	9CB0 @LOF	9CC5 @LOGGER	98E5
@@LOGOT	98FA @MSG	9931 @MUL16	9EE7
@@MUL8	9ED2 @OPEN	9C5C @PARAM	99A7
@@PAUSE	9992 @PEOF	9CDA @POSN	9CEF
@@PRINT	9946 @PRT	985E @PUT	9836
@@RAMDIR	9B0C @RDSEC	9B9F @RDSSC	9E69
@@READ	9D04 @REMOV	9C32 @RENAM	9C1D
@@REW	9D19 @RMTSK	9AA3 @RPTSK	9AB8
@@RREAD	9D2E @RSLCT	9B8A @RSTOR	9B4B
@@RUN	9DD6 @RWRTIT	9D43 @SEEK	9B75
@@SEEKSC	9D58 @SKIP	9D6D @SLCT	9B36
@@STEPI	9B60 @TIME	99D1 @VDCTL	997D
@@VER	9D82 @VRSEC	9BB4 @WEOF	9D97
@@WHERE	9872 @WRITE	9DAC @WRSEC	9BDE
@@WRSSC	9BF3 @WRTRK	9C08	

2400 is the transfer address

00000 Total errors

NOTES:

Command: PURGE

Library: SYS7/SYS

ISAM # : 72H

```

00100 ;LBPURGE/ASM - PURGE Command
0000 00110 TITLE <PURGE - LS-DOS 6.2>
00120 ;
000A 00130 LF EQU 10
000D 00140 CR EQU 13
002C 00150 PAR_ERR EQU 44 ;Parameter Error
42E0 00160 PASSWORD EQU 42E0H
00170 ;
0000 00180 *GET SVCMAC:3 ;SVC Macro equivalents
00010 ;SVC MAC/ASM - LS-DOS Version VI
00020 *LIST OFF
03900 *LIST ON
00190 ;
2400 00200 ORG 2400H
00210 ;
2400 00220 PURGE @@CKBRKC ;Break key down?
2400 3E6A 00001 LD A,106
2402 EF 00002 RST 40
2403 2804 00230 JR Z,BEGINA ;Ok if not
2405 21FFFF 00240 LD HL,-1 ; else abort
2408 C9 00250 RET
00260 ;
2409 ED73F327 00270 BEGINA LD (SAVEBP+1),SP ;Save stack pointer
240D 7E 00280 PURGE1 LD A,(HL) ;Bypass cmd line blanks
240E 23 00290 INC HL
240F FE20 00300 CP ' '
2411 28FA 00310 JR Z,PURGE1
2413 11B828 00320 LD DE,BLANKS ;Pt to filespec area
2416 0608 00330 LD B,8 ;Init for file name
2418 FE2D 00340 CP '-' ;If -, set up flag
241A 2005 00350 JR NZ,PUR0
241C 323E25 00360 LD (MFLG+1),A
241F 7E 00370 LD A,(HL)
2420 23 00380 INC HL
2421 CD2727 00390 PUR0 CALL PRSPEC
2424 FE2F 00400 CP '/' ;Ck on file EXT entered
2426 200A 00410 JR NZ,PUR1 ;Jump if no extension
2428 11C028 00420 LD DE,BLANKS+8 ;Point to ext field
242B 0603 00430 LD B,3 ;Max 3 chars
242D 7E 00440 LD A,(HL)
242E 23 00450 INC HL
242F CD2727 00460 CALL PRSPEC ;Ck on EXT
2432 FE3A 00470 PUR1 CP ':' ;Drive entered?
2434 0E00 00480 LD C,0 ;Init to drive 0
2436 201A 00490 JR NZ,PRMERRA ;Quit if no drive #
2438 7E 00500 LD A,(HL) ;P/u drive #
2439 23 00510 INC HL ;Bump to next field
243A CD2F29 00520 CALL PATCH1 ;Ck drive range 0-7
243D 79 00530 PUR2 LD A,C ;Xfer drive to regA
243E 327C26 00540 LD (TSTMPW+1),A ; & stuff for later
2441 00550 @@CKDRV ;Ck if drive available
2441 3E21 00003 LD A,33
2443 EF 00004 RST 40
2444 3E20 00560 LD A,32 ;"drive not avail..."
2446 CD3829 00570 CALL PATCH2 ;Ck WP or missing disk
2449 00580 @@GTDCT ;DCT to reg IY
2449 3E51 00005 LD A,81
244B EF 00006 RST 40
244C 11E728 00590 LD DE,PRMTBL$ ;Get parms
244F 00600 @@PARAM
244F 3E11 00007 LD A,17

```

```

2451 EF      00008      RST      40
2452 C2DB27  00610 PRMERRA JP      NZ,PRMERR      ;Jump on error
2455 210000  00620 DATPRM LD      HL,0            ;P/u date="from-to"
2458 7C      00630      LD      A,H
2459 B5      00640      OR      L
245A 2835    00650      JR      Z,PUR3            ;Bypass if not entered
245C 7E      00660      LD      A,(HL)           ;Check for "-to"
245D FE2D    00670      CP      '-'
245F 2818    00680      JR      Z,CKTO
2461 3E80    00690      LD      A,80H            ;Set from bit
2463 32C328  00700      LD      (FTFLG),A        ;Note from entered
2466 CD5727  00710      CALL    PAKDAT           ;Pack the date entry
2469 C2DD27  00720      JP      NZ,IOERR         ;Quit if bad date
246C ED43C428 00730      LD      (FMPAKD),BC
2470 7E      00740      LD      A,(HL)
2471 FE22    00750      CP      '-'
2473 2810    00760      JR      Z,FRCTO
2475 FE2D    00770      CP      '-'            ;Check for "-to"
2477 2018    00780      JR      NZ,PUR3
2479 23      00790 CKTO   INC      HL            ;Bypass the '-'
247A 7E      00800      LD      A,(HL)           ;Ck for end of parm
247B FE22    00810      CP      '-'
247D 2812    00820      JR      Z,PUR3            ;Go on parm end
247F CD5727  00830      CALL    PAKDAT           ; else pack the date
2482 C2DD27  00840      JP      NZ,IOERR         ;Quit on bad date
2485 3AC328  00850 FRCTO  LD      A,(FTFLG)
2488 F601    00860      OR      1                ;Set TO bit
248A 32C328  00870      LD      (FTFLG),A
248D ED43C628 00880      LD      (TOPAKD),BC      ;Stuff for later
2491 3AB025  00890 PUR3   LD      A,(QPARM+1)     ;Query parm used?
2494 B7      00900      OR      A
2495 2806    00910      JR      Z,DOEVER         ;Go if not
2497 CD1527  00920      CALL    CKINDO           ;Invalid command during
249A C2DD27  00930      JP      NZ,IOERR         ; <DO> processing
249D CD7B26  00940 DOEVER CALL    TSTMPW           ;Ck on master password
24A0 C2DD27  00950      JP      NZ,IOERR         ;Go if wrong
24A3 3A7C26  00960      LD      A,(TSTMPW+1)     ;P/u drive
24A6 4F      00970      LD      C,A
24A7 FD5609  00980      LD      D,(IY+9)         ;Get DIR cylinder
24AA 1E01    00990      LD      E,1             ;Pt to HIT sector
24AC 21002C  01000      LD      HL,HITBUF
24AF 01010   01010      @@FLAGS                ;Pt IY => Flags
24AF 3E65    00009      LD      A,101
24B1 EF      00010      RST      40
24B2 01020   01020      @@RDSSC                ;Read the HIT
24B2 3E55    00011      LD      A,85
24B4 EF      00012      RST      40
24B5 3E16    01030      LD      A,16H           ;Init "HIT read error..."
24B7 C2DD27  01040      JP      NZ,IOERR         ;Abort on read error
24BA 1818    01050      JR      SCNH3
24BA 01060 ;
24BA 01070 ;      Major loop to scan HIT for files
24BA 01080 ;
24BC E1      01090 SCNHIT POP      HL
24BD C1      01100 SCNH1 POP      BC            ;Rcvr HIT ptr DEC
24BE 262C    01110      LD      H,HITBUF<-8     ;Pt to hi-order buffer
24C0 68      01120      LD      L,B            ;Set lo-order DEC
24C1 7D      01130 SCNH2 LD      A,L
24C2 C620    01140      ADD     A,32            ;Pt to next one in
24C4 6F      01150      LD      L,A            ;Same dir sector
24C5 300D    01160      JR      NC,SCNH3        ;Jump if still in sector

```

The Source	LIBRARY Files	PURGE - LS-DOS 6.2	Page 00003
24C7 2C	01170	INC L	;Bump to next dir sector
24C8 FE1F	01180	CP 1FH	;End of the line?
24CA 2008	01190	JR NZ,SCNH3	;Loop if not
24CC 0E0D	01200	LD C,CR	
24CE	01210	@@DSP	;Write new line & exit
24CE 3E02	00013	LD A,2	
24D0 EF	00014	RST 40	
24D1 C3F827	01220	JP EXIT	
	01230 ;		
	01240 ;	Routine to check on dir record in use	
	01250 ;		
24D4 7D	01260 SCNH3	LD A,L	;Ignore BOOT & DIR
24D5 E6FE	01270	AND 0FEH	
24D7 28E8	01280	JR Z,SCNH2	
24D9 7E	01290	LD A,(HL)	;P/u HIT hash byte
24DA B7	01300	OR A	
24DB 28E4	01310	JR Z,SCNH2	;Ignore if spare
24DD 45	01320	LD B,L	;Save DEC
24DE C5	01330	PUSH BC	
24DF 7D	01340	LD A,L	;Get record # in L
24E0 E6E0	01350	AND 0E0H	
24E2 6F	01360	LD L,A	
24E3 A8	01370	XOR B	;Get sector # in A
24E4 FEFF	01380 SCNH3A	CP 0FFH	;Same as what's in core?
24E6 280D	01390	JR Z,SCNH4	;Bypass if same
24E8 32E524	01400	LD (SCNH3A+1),A	;Update indicator byte
24EB	01410	@@DIRRD	;Read this directory
24EB 3E57	00015	LD A,87	
24ED EF	00016	RST 40	
24EE C2DD27	01420	JP NZ,IOERR	;Quit on read error
24F1 7C	01430	LD A,H	;Set SBUFF pointer
24F2 32F624	01440	LD (SCNH4+1),A	
24F5 2600	01450 SCNH4	LD H,0	;Pt to dir buf hi-order
24F7 7E	01460	LD A,(HL)	;L set to lo-order
24F8 CB67	01470	BIT 4,A	;Ignore if not assigned
24FA 28C1	01480	JR Z,SCNH1	
24FC CB7F	01490	BIT 7,A	;Ignore if it's an
24FE 20BD	01500	JR NZ,SCNH1	; extended dir record
2500 CB77	01510	BIT 6,A	;Jump if not a SYS file
2502 280A	01520	JR Z,CKINV	
2504 110000	01530 SPARM	LD DE,0	;P/u S-parm
2507 7A	01540	LD A,D	
2508 B3	01550	OR E	;Ignore this one if
2509 CABD24	01560	JP Z,SCNH1	; S-parm not entered
250C 180C	01570	JR CKNAM	
	01580 ;		
	01590 ;	Non-SYS file	
	01600 ;		
250E CB5F	01610 CKINV	BIT 3,A	;Jump if visible
2510 2808	01620	JR Z,CKNAM	
2512 110000	01630 IPARM	LD DE,0	;I-parm
2515 7A	01640	LD A,D	;Ignore if I-parm not
2516 B3	01650	OR E	; entered as this file
2517 CABD24	01660	JP Z,SCNH1	; is invisible
	01670 ;		
	01680 ;	Parms match, grab filename & check class	
	01690 ;		
251A E5	01700 CKNAM	PUSH HL	;Save ptr to record
251B 7D	01710	LD A,L	;Pt to filename in dir
251C C605	01720	ADD A,5	
251E 6F	01730	LD L,A	

The Source	LIBRARY Files	PURGE - LS-DOS 6.2	Page 00004
251F 11B828	01740	LD DE,BLANKS	;Pt to parsed input
2522 060B	01750	LD B,11	;Ck name/ext (11-chars)
2524 1A	01760 SCNH5	LD A,(DE)	
2525 FE24	01770	CP '\$'	;Wild char?
2527 2807	01780	JR Z,SCNH6	;Always a match
2529 BE	01790	CP (HL)	;Not global, char match?
252A 2804	01800	JR Z,SCNH6	;Ck more if match
252C FE20	01810	CP ' '	;Blank = end of ck
252E 200D	01820	JR NZ,MFLG	;If not blank, no match
2530 23	01830 SCNH6	INC HL	;Bump pointers
2531 13	01840	INC DE	
2532 10F0	01850	DJNZ SCNH5	;Loop for 11 chars
2534 3A3E25	01860	LD A,(MFLG+1)	;Bypass if a match but
2537 B7	01870	OR A	; - exclude given
2538 C2BC24	01880	JP NZ,SCNHIT	
253B 1806	01890	JR SCNH6A	
253D 3E00	01900 MFLG	LD A,0	;Ignore if no match &
253F B7	01910	OR A	; no exclude given
2540 CAB24	01920	JP Z,SCNHIT	
2543 E1	01930 SCNH6A	POP HL	;Rcvr ptr to DIR+0
2544 E5	01940	PUSH HL	
	01950 ;		
	01960 ;		
	01970 ;	Now check if date matches	
2545 23	01980	INC HL	;Pt to date field
2546 CD4627	01990	CALL UNPACK	;Alter date for cpr
2549 3AC328	02000	LD A,(FTFLG)	
254C 07	02010	RLCA	;Tst fm bit
254D 3010	02020	JR NC,SCNH6B	
254F 7A	02030	LD A,D	;Ignore if no date
2550 B3	02040	OR E	; in DIR for file
2551 CAB24	02050	JP Z,SCNHIT	
2554 2AC428	02060	LD HL,(FMPAKD)	;P/u user entry
2557 EB	02070	EX DE,HL	
2558 CDD527	02080	CALL CPHLDE	;HL-DE
255B EB	02090	EX DE,HL	
255C DABC24	02100	JP C,SCNHIT	;Go if out of range
255F 3AC328	02110 SCNH6B	LD A,(FTFLG)	
2562 0F	02120	RRCA	;Tst TO bit
2563 300E	02130	JR NC,MATCHES	;Go if no TOPARM
2565 7A	02140	LD A,D	; else ck if file is dated
2566 B3	02150	OR E	
2567 CAB24	02160	JP Z,SCNHIT	;Go if no dir date
256A 2AC628	02170	LD HL,(TOPAKD)	;P/u user's packed date
256D CDD527	02180	CALL CPHLDE	;HL-DE
2570 DABC24	02190	JP C,SCNHIT	;Go if out of range
2573 E1	02200 MATCHES	POP HL	;Rcvr pointer to DIRREC
2574 E5	02210 DONAM	PUSH HL	
2575 7D	02220	LD A,L	; & point to file name
2576 C605	02230	ADD A,5	
2578 6F	02240	LD L,A	
2579 11CC28	02250	LD DE,FCB1\$;Pt to name/ext buffer
257C 0608	02260	LD B,8	;Max 8-char name
257E 7E	02270 DONAM1	LD A,(HL)	;Move filename into
257F FE20	02280	CP ' '	; buffer until space
2581 2805	02290	JR Z,DONAME2	; or 8 characters
2583 12	02300	LD (DE),A	
2584 23	02310	INC HL	
2585 13	02320	INC DE	
2586 10F6	02330	DJNZ DONAM1	
2588 7D	02340 DONAME2	LD A,L	;Point to file ext


```

2589 80      02350      ADD      A,B
258A 6F      02360      LD        L,A
258B 7E      02370      LD        A,(HL)      ;Is there an extension?
258C FE20    02380      CP        ' '
258E 2810    02390      JR        Z,DONAM5    ;Bypass if not
2590 3E2F    02400      LD        A,'/'
2592 12      02410      LD        (DE),A      ;Stuff ext separator
2593 13      02420      INC       DE
2594 0603    02430      LD        B,3          ;Init 3-char ext max
2596 7E      02440      LD        A,(HL)      ;Transfer up to space
2597 FE20    02450      CP        ' '          ; or 3 chars
2599 2805    02460      JR        Z,DONAM5
259B 12      02470      LD        (DE),A
259C 23      02480      INC       HL
259D 13      02490      INC       DE
259E 10F6    02500      DJNZ     DONAM4
25A0 3E3A    02510      LD        A,':'      ;Add the drivespec
25A2 12      02520      LD        (DE),A
25A3 13      02530      INC       DE
25A4 3A7C26  02540      LD        A,(TSTMPW+1) ;P/u drivespec
25A7 F630    02550      OR        '0'        ;Make it ASCII & stuff
25A9 12      02560      LD        (DE),A
25AA 13      02570      INC       DE
25AB 3E03    02580      LD        A,3          ;Terminate with ETX
25AD 12      02590      LD        (DE),A
25AE D5      02600      PUSH     DE          ;Save pointer
25AF 11FFFF  02610      LD        DE,-1      ;Query each file?
25B2 7A      02620      LD        A,D
25B3 B3      02630      OR        E
25B4 CA3D26  02640      JP        Z,NOPRMPPT    ;Not if not Q=N
                02650      ;
25B7         02660      @@DSPLY PRGFIL$      ;"Purge file?..."
                00017      IFEQ     01H,1
25B7 217728  00018      LD        HL,PRGFIL$
                00019      ENDIF
25BA 3E0A    00020      LD        A,10
25BC EF      00021      RST        40
25BD D1      02670      POP       DE          ;Rcvr ptr to file buf ETX
25BE E1      02680      POP       HL          ;Rcvr ptr to 1st dir byte
25BF D5      02690      PUSH     DE
25C0 23      02700      INC       HL          ;Pt to MOD bit
25C1 CB76    02710      BIT        6,(HL)      ;Test MOD flag
25C3 2808    02720      JR        Z,SCDAT1    ;Go if not mod'ed
25C5 3E20    02730      LD        A,' '      ;Put a space
25C7 12      02740      LD        (DE),A
25C8 13      02750      INC       DE
25C9 3E2B    02760      LD        A,'+'      ; and the mod sign
25CB 12      02770      LD        (DE),A
25CC 13      02780      INC       DE
25CD 3E20    02790      LD        A,' '      ;Write a space
25CF 12      02800      LD        (DE),A
25D0 13      02810      INC       DE
25D1 23      02820      INC       HL          ;Advance to date field
25D2 EB      02830      EX        DE,HL
25D3 367B    02840      LD        (HL),'{'    ;Stuff left brace
25D5 23      02850      INC       HL
25D6 EB      02860      EX        DE,HL
25D7 7E      02870      LD        A,(HL)
25D8 B7      02880      OR        A
25D9 283D    02890      JR        Z,SCDAT1    ;Ignore if no date saved
25DB 0F      02900      RRCA          ;Has date, get day

```

```

25DC 0F      02910      RRCA
25DD 0F      02920      RRCA
25DE E61F    02930      AND      1FH
25E0 062F    02940      LD        B,2FH      ;Convert day to decimal
25E2 04      02950      INC        B      ; by counting # of 10's
25E3 D60A    02960      SUB        10      ;Sub 10 from day #
25E5 30FB    02970      JR          NC,SCDAT2
25E7 C63A    02980      ADD        A,3AH      ;Cvrt lo order to ASCII
25E9 F5      02990      PUSH     AF      ;Save day low order
25EA 78      03000      LD        A,B      ;Stuff day hi order
25EB 12      03010      LD        (DE),A
25EC 13      03020      INC        DE      ;Bump
25ED F1      03030      POP        AF      ;Rcvr lo order day #
25EE 12      03040      LD        (DE),A      ;Stuff low order
25EF 13      03050      INC        DE      ;Bump pointer to msg
25F0 3E2D    03060      LD        A,'-'      ;Init seperator
25F2 12      03070      LD        (DE),A      ; and stuff in buffer
25F3 13      03080      INC        DE      ;Pt to month field
25F4 E5      03090      PUSH     HL      ;Save DIR ptr
25F5 2B      03100      DEC        HL      ;Pt to DIR+1 (month+)
25F6 7E      03110      LD        A,(HL)      ;P/u month etc
25F7 E60F    03120      AND        0FH      ;Strip off flags
25F9 3D      03130      DEC        A      ;(mon-1)*3 indexes string
25FA 4F      03140      LD        C,A      ; conversion table
25FB 07      03150      RLCA
25FC 81      03160      ADD        A,C
25FD 4F      03170      LD        C,A
25FE 0600    03180      LD        B,0
2600 219428  03190      LD        HL,MONTBL
2603 09      03200      ADD        HL,BC      ;Add offset to tbl start
2604 0E03    03210      LD        C,3
2606 EDB0    03220      LDIR      ;Move 3-char month
2608 3E2D    03230      LD        A,'-'      ;Suff separator char
260A 12      03240      LD        (DE),A
260B 13      03250      INC        DE      ;Advance to year field
260C 3E38    03260      LD        A,'8'      ;Stuff 8 of 1980
260E 12      03270      LD        (DE),A
260F 13      03280      INC        DE      ;Bump msg ptr
2610 E1      03290      POP        HL      ;Rcvr DIR+2
2611 7E      03300      LD        A,(HL)      ;P/u year field
2612 E607    03310      AND        7      ;Remove day
2614 C630    03320      ADD        A,'0'      ;Cvrt to ASCII
2616 12      03330      LD        (DE),A      ;Stuff -> msg
2617 13      03340      INC        DE
2618 3E03    03350      LD        A,3      ;Show etx for display
261A 12      03360      LD        (DE),A
261B        03370      @@DSPLY FCB1$      ;Display filename
                00022      IFEQ      01H,1
261B 21CC28  00023      LD        HL,FCB1$
                00024      ENDIF
261E 3E0A    00025      LD        A,10
2620 EF      00026      RST        40
2621        00027      @@DSPLY QMARK$      ;Display ???
                00027      IFEQ      01H,1
2621 218428  00028      LD        HL,QMARK$
                00029      ENDIF
2624 3E0A    00030      LD        A,10
2626 EF      00031      RST        40
2627 21C828  00032      LD        HL,LILBUF$      ;Get response y,n
262A 010003  00033      LD        BC,3<8      ;For Yes, No
262D        00034      @@KEYIN

```

The Source	LIBRARY Files	PURGE - LS-DOS 6.2	Page 00007
262D 3E09	00032	LD A,9	
262F EF	00033	RST 40	
2630 DAEC27	03420	JP C,BREAK	;Abort on <BREAK>
2633 7E	03430	LD A,(HL)	;P/u response
2634 CBAF	03440	RES 5,A	;Strip l/c if entered
2636 FE59	03450	CP 'Y'	;Is it yes?
2638 C2BC24	03460	JP NZ,SCNHIT	;Bypass if not
263B E3	03470	EX (SP),HL	;Place dummy HL below
263C E5	03480	PUSH HL	; pointer
263D FDCB0A46	03490	NOPRMPT BIT 0,(IY+'K'-'A')	;Ck if BREAK bit in
2641 C2EC27	03500	JP NZ,BREAK	; KFLAG is active
2644	03510	@@LOGOT PURGE\$;Dsply "Purging: "
	00034	IFEQ 01H,1	
2644 218A28	00035	LD HL,PURGE\$	
	00036	ENDIF	
2647 3E0C	00037	LD A,12	
2649 EF	00038	RST 40	
264A E1	03520	POP HL	;Get pointer where ETX is
264B 360D	03530	LD (HL),CR	; & replace with CR
264D	03540	@@LOGOT FCB1\$;Dsply filename
	00039	IFEQ 01H,1	
264D 21CC28	00040	LD HL,FCB1\$	
	00041	ENDIF	
2650 3E0C	00042	LD A,12	
2652 EF	00043	RST 40	
2653 E1	03550	POP HL	;Pop dummy or DIRREC ptr
2654 C1	03560	POP BC	;Get drive & DEC
2655 C5	03570	PUSH BC	
2656 78	03580	LD A,B	;P/u the DEC
2657 321629	03590	LD (FCB+7),A	; & stuff
265A 3A7C26	03600	LD A,(TSTMPW+1)	;P/u drive
265D 321529	03610	LD (FCB+6),A	; & stuff
2660 3E01	03620	LD A,1	;Set up FCB for remove
2662 321029	03630	LD (FCB+1),A	
2665 3E80	03640	LD A,80H	;Show FCB as open
2667 320F29	03650	LD (FCB),A	
266A 110F29	03660	LD DE,FCB	;Remove the file
266D	03670	@@REMOV	
266D 3E39	00044	LD A,57	
266F EF	00045	RST 40	
2670 C2DD27	03680	JP NZ,IOERR	;Jump on error
2673 3EFF	03690	LD A,0FFH	;Show we don't have the
2675 32E524	03700	LD (SCNH3A+1),A	; latest dir record
2678 C3BD24	03710	JP SCNH1	;Loop
	03720 ;		
	03730 ;		
	03740 ;		
		Routine to get the master password & match it	
267B 0E00	03750	TSTMPW LD C,\$-\$;Init to drive requested
267D CD0427	03760	CALL GATRD	;Read GAT into GATBUF
2680 C0	03770	RET NZ	;Back on error
2681 2ACE2B	03780	LD HL,(GATBUF+0CEH)	
2684 11E042	03790	LD DE,PASSWORD	;Password="PASSWORD" ?
2687 AF	03800	XOR A	
2688 ED52	03810	SBC HL,DE	
268A C8	03820	RET Z	;Back if PASSWORD
	03830 ;		
	03840 ;		
	03850 ;		
		MPW is not "PASSWORD" - check entry match	
268B 110000	03860	PWPARAM LD DE,0	;P/u MPW string addr
268E 212B28	03870	LD HL,MPW\$;Init prompt
2691 CDA226	03880	CALL GETMPW	;Hash parm or entry

```

2694 C0      03890      RET      NZ
2695 EB      03900      EX       DE,HL      ;Xfer hashed MPW to DE
2696 2ACE2B   03910      LD       HL,(GATBUF+0CEH) ;Grab pack MPW &
2699 AF      03920      XOR      A          ; check if user entered
269A ED52     03930      SBC      HL,DE      ; the pack MPW
269C 214328   03940      LD       HL,BADMPW$    ;Init error pointer
269F 3E3F     03950      LD       A,63      ;Set extended error
26A1 C9      03960      RET              ;Z or NZ
          03970 ;
          03980 ;      Routine to get 8-char password
          03990 ;
26A2 CDA926   04000 GETMPW CALL     GMPW1      ;Test if user entered MPW
26A5 C0      04010      RET      NZ
26A6 3EE4     04020      LD       A,0E4H      ;Hash password (DE) to HL
26A8 EF      04030      RST      28H      ;Ret to what called
26A9 7A      04040 GMPW1 LD       A,D        ;Test if user entered MPW
26AA B3      04050      OR       E
26AB 281D     04060      JR       Z,GMPW3      ;Prompt if not
26AD 3C      04070      INC      A          ; or no operand
26AE 281A     04080      JR       Z,GMPW3
          04090 ;
          04100 ;      Place entered password into buffer
          04110 ;
26B0 21002A   04120      LD       HL,BUFFER
26B3 E5      04130      PUSH    HL
26B4 0608     04140      LD       B,8          ;Max entry of 8 chars
26B6 1A      04150 GMPW2 LD       A,(DE)      ;P/u pswd char
26B7 FE0D     04160      CP       CR        ;End of the line?
26B9 282F     04170      JR       Z,GMPW4      ;Space out if so
26BB FE2C     04180      CP       ','      ;Comma separator?
26BD 282B     04190      JR       Z,GMPW4
26BF FE22     04200      CP       '"'      ;Closing quote?
26C1 2827     04210      JR       Z,GMPW4
26C3 13      04220      INC      DE
26C4 77      04230      LD       (HL),A      ;Xfer the char
26C5 23      04240      INC      HL
26C6 10EE     04250      DJNZ    GMPW2      ;Loop for 8
26C8 1825     04260      JR       GMPW5
          04270 ;
          04280 ;      Not entered as parm, grab from keyboard
          04290 ;
26CA CD1527   04300 GMPW3 CALL     CKIND0      ;Can't prompt in <D0>
26CD C0      04310      RET      NZ
26CE         04320      @@DSPLY      ;Display request
          00046      IFEQ     00H,1
          00047      LD       HL,
          00048      ENDIF
26CE 3E0A     00049      LD       A,10
26D0 EF      00050      RST      40
26D1 C0      04330      RET      NZ
26D2 010008   04340      LD       BC,8<8      ;Max 8 chars input
26D5 21002A   04350      LD       HL,BUFFER      ;Pt to buffer
26D8 E5      04360      PUSH    HL
26D9         04370      @@KEYIN      ;Get parm input
26D9 3E09     00051      LD       A,9
26DB EF      00052      RST      40
26DC DAEC27   04380      JP       C,BREAK      ;Exit on Break
26DF EB      04390      EX       DE,HL      ;Buf start to DE
26E0 2600     04400      LD       H,0          ;Buf len to HL
26E2 68      04410      LD       L,B
26E3 19      04420      ADD     HL,DE      ;Pt to 1st unused pos

```

```

26E4 3E08 04430 LD A,8 ;Calculate spaces needed
26E6 90 04440 SUB B
26E7 2806 04450 JR Z,GMPW5 ;Ret if none needed
26E9 47 04460 LD B,A ;Set counter for spaces
26EA 3620 04470 GMPW4 LD (HL),' ' ; & put them in
26EC 23 04480 INC HL
26ED 10FB 04490 DJNZ GMPW4
04500 ;
04510 ; Convert (SP) through (SP)+7 to upper case
04520 ;
26EF E1 04530 GMPW5 POP HL ;Rcvr pointer to buf
26F0 E5 04540 PUSH HL
26F1 0608 04550 LD B,8 ;Loop through field
26F3 7E 04560 GMPW6 LD A,(HL)
26F4 FE61 04570 CP 'a'
26F6 3806 04580 JR C,GMPW7
26F8 FE7B 04590 CP 'z'+1
26FA 3002 04600 JR NC,GMPW7
26FC CBAE 04610 RES 5,(HL) ;L/c -> U/C
26FE 23 04620 GMPW7 INC HL
26FF 10F2 04630 DJNZ GMPW6
2701 D1 04640 POP DE ;Rcvr ptr to start
2702 AF 04650 XOR A ;Indicate no error
2703 C9 04660 RET
04670 ;
04680 ; Routine to read the granule allocation table
04690 ;
2704 D5 04700 GATRD PUSH DE
2705 E5 04710 PUSH HL
2706 FD5609 04720 LD D,(IY+9) ;Dir cylinder
2709 21002B 04730 LD HL,GATBUF
270C 5D 04740 LD E,L ;Set to sector 0
270D 04750 @RDSSC
270D 3E55 00053 LD A,85
270F EF 00054 RST 40
2710 E1 04760 POP HL
2711 D1 04770 POP DE
2712 3E14 04780 LD A,14H ;Init "GAT read error
2714 C9 04790 RET ;Z or NZ
04800 ;
04810 ; Routine to check if <D0> active
04820 ;
2715 FDE5 04830 CKIND0 PUSH IY
2717 04840 @@FLAGS
2717 3E65 00055 LD A,101
2719 EF 00056 RST 40
271A FDCB126E 04850 BIT 5,(IY+'S'-'A') ;Set if D0 active
271E FDE1 04860 POP IY
2720 C8 04870 RET Z
2721 210428 04880 LD HL,NOINDOS
2724 3E3F 04890 LD A,63
2726 C9 04900 RET
04910 ;
04920 ; Parse file name or ext on command line
04930 ;
2727 FE24 04940 PRSPEC CP '$' ;Wild character?
2729 2814 04950 JR Z,PRS2 ;Always a match
272B FE41 04960 CP 'A' ;Ck on filename entry
272D 3006 04970 JR NC,PRS1 ;Jump if possible alpha
272F FE3A 04980 CP '9'+1 ;Ck on 0-9
2731 D0 04990 RET NC ;Bad if > 9 and < A

```

```

2732 FE30 05000 CP '0'
2734 D8 05010 RET C ;Bad if < 0
2735 FE61 05020 PRS1 CP 'a' ;Cvrt to UC if needed
2737 3806 05030 JR C,PRS2
2739 FE7B 05040 CP 'z'+1
273B 3002 05050 JR NC,PRS2
273D CBAF 05060 RES 5,A
273F 12 05070 PRS2 LD (DE),A ;Xfer char to buffer
2740 13 05080 INC DE ;Bump dest ptr
2741 7E 05090 LD A,(HL) ;Get next char
2742 23 05100 INC HL ;Bump source ptr
2743 10E2 05110 DJNZ PRSPEC ;Loop 8 max
2745 C9 05120 RET
05130 ;
05140 ; Routine to extract date from directory
05150 ;
2746 7E 05160 UNPACK LD A,(HL) ;P/u DIR+1
2747 E60F 05170 AND 0FH ;Remove flags
2749 57 05180 LD D,A ;Save month
274A 23 05190 INC HL ;Pt to DIR+2
274B 7E 05200 LD A,(HL)
274C E6F8 05210 AND 0F8H ;Strip year
274E 5F 05220 LD E,A ;Save day in E
274F 7E 05230 LD A,(HL)
2750 AB 05240 XOR E ;Strip day fm year
2751 0F 05250 RRCA ;Shift year to 5-7
2752 0F 05260 RRCA
2753 0F 05270 RRCA
2754 B2 05280 OR D ;Merge with month
2755 57 05290 LD D,A
2756 C9 05300 RET
05310 ;
05320 ; Pack user date string
05330 ;
2757 7E 05340 PAKDAT LD A,(HL)
2758 0E2F 05350 LD C,'/' ;Init separator
275A CDA27 05360 CALL PARSDAT ;Parse entry
275D 203C 05370 JR NZ,BADFMT ;Jump on format error
275F EB 05380 EX DE,HL
2760 3AC828 05390 LD A,(LILBUF$) ;Is year a leap year?
2763 E603 05400 AND 3
2765 215C28 05410 LD HL,MAXDAYS+1 ;Set Feb to have 29 days
2768 2001 05420 JR NZ,$+3 ; if so
276A 34 05430 INC (HL)
276B 3ACA28 05440 LD A,(LILBUF$+2) ;P/u month
276E 3D 05450 DEC A ;Range check
276F FE0C 05460 CP 12
2771 3028 05470 JR NC,BADFMT ;Go if 0 or >12
2773 2B 05480 DEC HL ;Point to Jan entry
2774 85 05490 ADD A,L ;Index the month
2775 6F 05500 LD L,A
2776 7C 05510 LD A,H
2777 CE00 05520 ADC A,0
2779 67 05530 LD H,A
277A 3AC928 05540 LD A,(LILBUF$+1) ;P/u day entry
277D 3D 05550 DEC A ;Reduce for test (0->FF)
277E BE 05560 CP (HL)
277F 301A 05570 JR NC,BADFMT ;Go if too large (or 0)
2781 21CA28 05580 LD HL,LILBUF$+2 ;Pt to month
2784 7E 05590 LD A,(HL) ;P/u month
2785 2B 05600 DEC HL ;Pt to day

```

```

2786 47      05610      LD      B,A          ;Save month in B
2787 7E      05620      LD      A,(HL)       ;P/u day
2788 2B      05630      DEC      HL          ;Pt to year
2789 07      05640      RLCA          ;Shift day to 3-7
278A 07      05650      RLCA
278B 07      05660      RLCA
278C 4F      05670      LD      C,A
278D 7E      05680      LD      A,(HL)       ;P/u year
278E D650    05690      SUB      80          ;Adjust for offset
2790 3001    05700      JR      NC,$+3       ;If entry < 1980,
2792 AF      05710      XOR      A          ; then use 1980
2793 0F      05720      RRCA          ;Shift into bits 5-7
2794 0F      05730      RRCA
2795 0F      05740      RRCA
2796 B0      05750      OR       B          ; & merge with month
2797 47      05760      LD      B,A
2798 EB      05770      EX      DE,HL
2799 AF      05780      XOR      A          ;Set Z, no error
279A C9      05790      RET
                05800 ;
279B 216728 05810 BADFMT LD      HL,BADFMT$ ;Init error pointer
279E 3E3F    05820      LD      A,63         ;Set extended error
27A0 B7      05830      OR       A
27A1 C9      05840      RET
                05850 ;
                05860 ;      Routine to parse DATE/TIME entry
                05870 ;
27A2 11CA28 05880 PARSDAT LD      DE,LILBUF$+2 ;Point to buf end
27A5 0603    05890      LD      B,3         ;Process 3 fields
27A7 D5      05900 PRSD1  PUSH     DE        ;Save pointer
27A8 CDB727 05910      CALL    PRSD2       ;Get a digit pair
27AB D1      05920      POP      DE        ;Recover pointer
27AC C0      05930      RET      NZ        ;Ret if bad digit pair
27AD 12      05940      LD      (DE),A     ; else stuff the value
27AE 1B      05950      DEC      DE        ;Backup the pointer
27AF 05      05960      DEC      B         ;Loop countdown
27B0 C8      05970      RET      Z
27B1 7E      05980      LD      A,(HL)     ;Ck for valid separator
27B2 23      05990      INC      HL        ;Bump pointer
27B3 B9      06000      CP       C         ;Separator char required
27B4 28F1    06010      JR      Z,PRSD1    ;Loop if match
27B6 C9      06020      RET              ; else ret bad (NZ)
                06030 ;
                06040 ;      Routine to parse a digit pair
                06050 ;
27B7 CDCE27 06060 PRSD2  CALL    PRS4        ;Get a digit
27BA 3010    06070      JR      NC,PRSD3    ;Jump if bad digit
27BC 5F      06080      LD      E,A        ;Multiply by ten
27BD 07      06090      RLCA
27BE 07      06100      RLCA
27BF 83      06110      ADD      A,E
27C0 07      06120      RLCA
27C1 5F      06130      LD      E,A
27C2 CDCE27 06140      CALL    PRS4        ;Get another digit
27C5 3005    06150      JR      NC,PRSD3    ;Jump on bad digit
27C7 83      06160      ADD      A,E        ;Accumulate new digit
27C8 5F      06170      LD      E,A        ;Save 2-digit value
27C9 AF      06180      XOR      A        ;Clear flags
27CA 7B      06190      LD      A,E        ;Xfer field value
27CB C9      06200      RET
                06210 ;

```

```

27CC B7      06220 PRSD3  OR      A              ;Set NZ
27CD C9      06230      RET
27CE 7E      06240 PRS4  LD      A,(HL)        ;P/u a digit &
27CF 23      06250      INC      HL            ; convert to binary
27D0 D630    06260      SUB      '0'
27D2 FE0A    06270      CP      10
27D4 C9      06280      RET
           06290 ;
           06300 ;      Routine to compare DE to HL
           06310 ;
27D5 7C      06320 CPHLDE LD      A,H
27D6 92      06330      SUB      D
27D7 C0      06340      RET      NZ
27D8 7D      06350      LD      A,L
27D9 93      06360      SUB      E
27DA C9      06370      RET
           06380 ;
           06390 ;      Error processing
           06400 ;
27DB 3E2C    06410 PRMERR LD      A,PAR_ERR      ;Parameter Error
27DD FE3F    06420 IOERR CP      63             ;Extended error?
27DF 281E    06430      JR      Z,EXTERR
27E1 6F      06440      LD      L,A
27E2 2600    06450      LD      H,0
27E4 F6C0    06460      OR      0C0H           ;Abbrev & return
27E6 4F      06470      LD      C,A
27E7          06480      @ERROR
27E7 3E1A    00057      LD      A,26
27E9 EF      00058      RST      40
27EA 1806    06490      JR      SAVESP
           06500 ;
           06510 ;      BREAK handler routine
           06520 ;
27EC          06530 BREAK @@CKBRKC             ;Clear Break Bit
27EC 3E6A    00059      LD      A,106
27EE EF      00060      RST      40
27EF 21FFFF  06540 ERREXIT LD      HL,-1
27F2 310000  06550 SAVESP LD      SP,$-$       ;Restore the stack
27F5 22F927  06560      LD      (RETCOD),HL
27F8          06570 EXIT  EQU      $           ;Exit clears Break
27F8 210000  06580      LD      HL,0
27F9          06590 RETCOD EQU      $-2
27FB          06600      @@CKBRKC
27FB 3E6A    00061      LD      A,106
27FD EF      00062      RST      40
27FE C9      06610      RET
           06620 ;
27FF          06630 EXTERR @@LOGOT
           00063      IFEQ      00H,1
           00064      LD      HL,
           00065      ENDIF
27FF 3E0C    00066      LD      A,12
2801 EF      00067      RST      40
2802 18EB    06640      JR      ERREXIT
2804 49      06650 NOINDO$ DB      'Invalid command during <DO> '
           6E 76 61 6C 69 64 20 63
           6F 6D 6D 61 6E 64 20 64
           75 72 69 6E 67 20 3C 44
           4F 3E 20
2820 70      06660      DB      'processing',CR
           72 6F 63 65 73 73 69 6E

```



```

67 0D
282B 4D      06670 MPW$   DB      'Master password ?      ',3
61 73 74 65 72 20 70 61
73 73 77 6F 72 64 20 3F
20 20 20 20 20 20 03
2843 49      06680 BADMPW$ DB      'Invalid master password',CR
6E 76 61 6C 69 64 20 6D
61 73 74 65 72 20 70 61
73 73 77 6F 72 64 0D
285B 1F      06690 MAXDAYS DB      31,28,31,30,31,30,31,31,30,31,30,31
1C 1F 1E 1F 1E 1F 1F 1E
1F 1E 1F
2867 42      06700 BADFMT$ DB      'Bad date format',CR
61 64 20 64 61 74 65 20
66 6F 72 6D 61 74 0D
      06710 ;
2877 50      06720 PRGFIL$ DB      'Purge file: ',3
75 72 67 65 20 66 69 6C
65 3A 20 03
2884 7D      06730 QMARK$  DB      '} ? ',3
20 3F 20 20 03
288A 50      06740 PURGE$  DB      'Purging: ',3
75 72 67 69 6E 67 3A 20
03
2894 4A      06750 MONTBL  DM      '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
28B8 20      06760 BLANKS  DM      '
20 20 20 20 20 20 20 20
20 20
28C3 00      06770 FTFLG   DB      0
0002      06780 FMPAKD   DS      2
0002      06790 TOPAKD   DS      2
0004      06800 LILBUF$  DS      4
001B      06810 FCB1$    DS      27
      06820 ;
      06830 ;      Parameter table
      06840 ;
28E7 80      06850 PRMTBL$  DB      80H
0080      06860 VAL      EQU      80H
0040      06870 SW       EQU      40H
0020      06880 STR      EQU      20H
0010      06890 SGL      EQU      10H
28E8 53      06900          DB      SW!SGL!3,'INV',0
49 4E 56 00
28ED 1325    06910          DW      IPARM+1
28EF 53      06920          DB      SW!SGL!3,'SYS',0
53 59 53 00
28F4 0525    06930          DW      SPARM+1
28F6 73      06940          DB      SW!STR!SGL!3,'MPW',0
4D 50 57 00
28FB 8C26    06950          DW      PWPARAM+1
28FD 55      06960          DB      SW!SGL!5,'QUERY',0
51 55 45 52 59 00
2904 B025    06970          DW      QPARAM+1
2906 34      06980          DB      STR!SGL!4,'DATE',0
44 41 54 45 00
290C 5624    06990          DW      DATPRM+1

```

```

290E 00      07000      NOP
              07010 ;
0020      07020 FCB      DS      32
292F D630    07030 PATCH1 SUB      '0'      ;Cvrt to binary
2931 FE08    07040      CP      7+1
2933 D2DB27  07050      JP      NC,PRMERR
2936 4F      07060      LD      C,A
2937 C9      07070      RET
              07080 ;
2938 C2DD27  07090 PATCH2 JP      NZ,IOERR      ;Go on CKDRV error
293B 3E0F    07100      LD      A,15      ;Init WP error
293D DADD27  07110      JP      C,IOERR      ;Exit if WP
2940 C9      07120      RET
              07130 ;
2A00      07140      ORG      $<-8+1<+8
0100      07150 BUFFER DS      256
0100      07160 GATBUF DS      256
0100      07170 HITBUF DS      256
2CFF      07180 LAST  EQU      $-1
              07190 ;
2400      07200      END      PURGE

```

@@1	0000 @@2	0000 @@3	0000
@@4	0000 @MOD2	0000 @MOD4	FFFF
BADFMT	279B BADFMT\$	2867 BADMPW\$	2843
BEGINA	2409 BLANKS	28B8 BREAK	27EC
BUFFER	2A00 CKINDO	2715 CKINV	250E
CKNAM	251A CKTO	2479 CPHLDE	27D5
CR	000D DATPRM	2455 DOEVER	249D
DONAM	2574 DONAM1	257E DONAM4	2596
DONAM5	25A0 DONAME2	2588 ERREXIT	27EF
EXIT	27F8 EXTERR	27FF FCB	290F
FCB1\$	28CC FMPAKD	28C4 FRCTO	2485
FTFLG	28C3 GATBUF	2B00 GATRD	2704
GETMPW	26A2 GMPW1	26A9 GMPW2	26B6
GMPW3	26CA GMPW4	26EA GMPW5	26EF
GMPW6	26F3 GMPW7	26FE HITBUF	2C00
IOERR	27DD IPARM	2512 LAST	2CFF
LF	000A LILBUF\$	28C8 MATCHES	2573
MAXDAYS	285B MFLG	253D MONTBL	2894
MPW\$	282B MINDO\$	2804 NOPRMPT	263D
PAKDAT	2757 PARSDAT	27A2 PAR ERR	002C
PASSWORD	42E0 PATCH1	292F PATCH2	2938
PRGFIL\$	2877 PRMERR	27DB PRMERRA	2452
PRMTBL\$	28E7 PRS1	2735 PRS2	273F
PRS4	27CE PRSD1	27A7 PRSD2	27B7
PRSD3	27CC PRSPEC	2727 PUR0	2421
PUR1	2432 PUR2	243D PUR3	2491
PURGE	2400 PURGE\$	288A PURGE1	240D
PWPARAM	268B QMARK\$	2884 QPARAM	25AF
RETCOD	27F9 SAVESP	27F2 SCDAT1	25CD
SCDAT2	25E2 SCDAT4	2618 SCNH1	24BD
SCNH2	24C1 SCNH3	24D4 SCNH3A	24E4
SCNH4	24F5 SCNH5	2524 SCNH6	2530
SCNH6A	2543 SCNH6B	255F SCNHIT	24BC
SGL	0010 SPARM	2504 STR	0020
SW	0040 TOPAKD	28C6 TSTMPW	267B
UNPACK	2746 VAL	0080 @@ABORT	AB4A
@@ADTSK	ABDD @@BANK	B0F5 @@BKSP	ADD5
@@BREAK	B10B @@CHNIO	AB35 @@CKBRKC	B159
@@CKDRV	AC31 @@CKEOF	ADEA @@CKTSK	ABC8
@@CLOSE	ADC0 @@CLS	B143 @@CMNDI	AB74
@@CMNDR	AB89 @@CTL	A999 @@DATE	AB0B
@@DCSTAT	AC70 @@DEBUG	ABB3 @@DECHEX	B075
@@DIRRD	AFE2 @@DIRWR	AFF7 @@DIV16	B060
@@DIV8	B04B @@DODIR	AC46 @@DSP	A95D
@@DSPLY	A9FD @@ERROR	AB9E @@EXIT	AB5F
@@FEXT	AF4F @@FLAGS	B0DF @@FNAME	AF64
@@FSPEC	AF3A @@GATRD	AFCD @@GATWR	B00C
@@GET	A971 @@GTDCB	AF8E @@GTDCT	AF79
@@GTMOD	AFA3 @@HDFMT	AD18 @@HEX16	B0B4
@@HEX8	B09F @@HEXDEC	B08A @@HIGH\$	B0C9
@@INIT	AD96 @@KBD	A9D5 @@KEY	A949
@@KEYIN	A9E9 @@KLTSK	AC1C @@LOAD	AF10
@@LOC	ADFF @@LOF	AE14 @@LOGGER	AA34
@@LOGOT	AA49 @@MSG	AA80 @@MUL16	B036
@@MUL8	B021 @@OPEN	ADAB @@PARAM	AAF6
@@PAUSE	AAE1 @@PEOF	AE29 @@POSN	AE3E
@@PRINT	AA95 @@PRT	A9AD @@PUT	A985
@@RAMDIR	AC5B @@RDSEC	ACEE @@RDSSC	AFB8
@@READ	AE53 @@REMOV	AD81 @@RENAM	AD6C
@@REW	AE68 @@RMTSK	ABF2 @@RPTSK	AC07

The Source

LIBRARY Files

PURGE - LS-DOS 6.2

Page 00016

@@RREAD

AE7D @@RSLCT

ACD9 @@RSTOR

AC9A

@@RUN

AF25 @@RWRT

AE92 @@SEEK

ACC4

@@SEEKSC

AEA7 @@SKIP

AEBC @@SLCT

AC85

@@STEPI

ACAF @@TIME

AB20 @@VDCTL

AACC

@@VER

AED1 @@VRSEC

AD03 @@WEOF

AEE6

@@WHERE

A9C1 @@WRITE

AEFB @@WRSEC

AD2D

@@WRSSC

AD42 @@WRTRK

AD57

2400 is the transfer address

00000 Total errors

NOTES:

Command: REMOVE

Library: SYS6/SYS

ISAM # : 18H

```

00100 ;LBREMOVE/ASM - REMOVE command
0000 00110 TITLE <REMOVE - LS-DOS 6.2>
00120 ;
0000 00130 *GET SVCMAC:3 ;SVC Macro equivalents
00010 ;SVC MAC/ASM - LS-DOS Version VI
00020 *LIST OFF
00300 *LIST ON
0000 00140 *GET VALUES:3 ;Misc. equates
00320 ;VALUES/ASM - Version 6
00330 *LIST OFF
00420 *LIST ON
00150 ;
2400 00160 ORG 2400H
00170 ;
2400 00180 REMOVE @@CKBRKC ;Break key down?
2400 3E6A 00001 LD A,106
2402 EF 00002 RST 40
2403 2804 00190 JR Z,BEGINA ;Ok if not
2405 21FFFF 00200 LD HL,-1 ; else abort
2408 C9 00210 RET
00220 ;
2409 ED731B24 00230 BEGINA LD (SAVE SP+1),SP ;Save Stack
240D CD3724 00240 CALL REMOV ;Do the removing
2410 1805 00250 JR EXIT
00260 ;
00270 ; Exit Routine - Clear any <BREAK> & return
00280 ;
00290 BRKABT
2412 21FFFF 00300 LD HL,-1
2415 1803 00310 JR SAVESP ;Abort with error
00320 ;
2417 210000 00330 EXIT LD HL,0 ;Set retcod if exiting
241A 310000 00340 SAVESP LD SP,$-$ ;P/u stack
241D 00350 @@CKBRKC ;Clear <BREAK> if entered
241D 3E6A 00003 LD A,106
241F EF 00004 RST 40
2420 C9 00360 RET ; and Return
00370 ;
00380 ; Special Message Exit
00390 ;
2421 212C25 00400 SPCREQ LD HL,SPCREQ$
2424 00410 @@LOGOT
00005 IFEQ 00H,1
00006 LD HL,
00007 ENDIF
2424 3E0C 00008 LD A,12
2426 EF 00009 RST 40
2427 21FFFF 00420 ERREXIT LD HL,-1
242A 18EE 00430 JR SAVESP ;Get stack & RET
00440 ;
00450 ; I/O Error Exit
00460 ;
242C 6F 00470 IOERR LD L,A ;Error # to HL
242D 2600 00480 LD H,0
242F F6C0 00490 OR 0C0H ;Abbrev, return
2431 4F 00500 LD C,A ;Error code to reg C
2432 00510 @@ERROR
2432 3E1A 00010 LD A,26
2434 EF 00011 RST 40
2435 18E3 00520 JR SAVESP ;P/u stack & return
00530 ;

```

```

00540 ;      REMOV - Remove file(s) or device(s)
00550 ;
2437 E5      00560 REMOV  PUSH  HL          ;Save INBUF$ pointer
2438 AF      00570 REMOV0 XOR    A          ;Set Z-flag for loop
2439 E1      00580 REMOV1 POP    HL
243A C22C24  00590      JP     NZ,IOERR
00600 ;
00610 ;      Skip leading spaces - exit on terminator
00620 ;
243D      00630      @CKBRKC          ;Check for break
243D 3E6A    00012      LD     A,106
243F EF      00013      RST     40
2440 20D0    00640      JR     NZ,BRKABT      ; abort if so
00650 ;
2442 2B      00660      DEC     HL
2443 23      00670 REMOV2 INC     HL          ;Skip past spaces
2444 7E      00680      LD     A,(HL)
2445 FE20    00690      CP     ' '
2447 28FA    00700      JR     Z,REMOV2
2449 D8      00710      RET     C          ;Return on terminator
00720 ;
00730 ;      Hit a non-space char - is this a filespec ?
00740 ;
244A 115B25  00750 REMOV3 LD     DE,FCB      ;Fetch the filespec
244D      00760      @F$SPEC
244D 3E4E    00014      LD     A,78
244F EF      00015      RST     40
2450 C22124  00770      JP     NZ,SPCREQ      ;Jump on spec error
2453 E5      00780      PUSH   HL          ;Save INBUF$ pointer
00790 ;
00800 ;      Transfer Filespec to buff without password
00810 ;
2454 D5      00820      PUSH   DE          ;Save FCB ptr
2455 EB      00830      EX     DE,HL        ;Pt HL => FCB
2456 114C25  00840      LD     DE,FILESPEC   ;DE => Filespec
2459 CD0B25  00850      CALL   MOVSPC       ;Move filespec
245C 3E0D    00860      LD     A,CR         ;End spec with C/R
245E 12      00870      LD     (DE),A
245F D1      00880      POP     DE          ;Recover FCB
00890 ;
00900 ;      Are we Removing a Device ?
00910 ;
2460 1A      00920      LD     A,(DE)        ;Is it a device?
2461 FE2A    00930      CP     '*'
2463 2843    00940      JR     Z,RMVDVC
00950 ;
00960 ;      Removing a File - Does it exist ?
00970 ;
2465      00980      @OPEN          ;Open the file
2465 3E3B    00016      LD     A,59
2467 EF      00017      RST     40
2468 2804    00990      JR     Z,GOODOPN      ;Z - it does exist
246A FE2A    01000      CP     42          ;LRL Open Fault ?
246C 20CB    01010      JR     NZ,REMOV1      ;No - abort
01020 ;
01030 ;      Successful Open - Can we Remove the file ?
01040 ;
246E 3A5C25  01050 GOODOPN LD     A,(FCB+1)   ;P/u access level
2471 E607    01060      AND     7
2473 FE02    01070      CP     2          ;REMOVE access ?
2475 3E25    01080      LD     A,37        ;No - "Illegal access

```



```

2477 D22C24 01090 JP NC,IOERR ; attempted to ..."
01100 ;
01110 ; Is there a drivespec in the filespec ?
01120 ;
247A 214C25 01130 LD HL,FILESPEC ;HL => Filespec
247D 7E 01140 FLOOP LD A,(HL) ;P/u char
247E FE0E 01150 CP CR+1 ;End of Filespec ?
2480 300E 01160 JR NC,CHKDSPC
01170 ;
01180 ; Drivespec wasn't specified - put it on
01190 ;
2482 363A 01200 LD (HL),':' ;Append drivespec onto
2484 23 01210 INC HL ; end of filespec
2485 3A6125 01220 LD A,(FCB+6) ;Xfer drive # to A
2488 C630 01230 ADD A,'0' ;Convert to ASCII
248A 77 01240 LD (HL),A
248B 23 01250 INC HL ;Bump
248C 360D 01260 LD (HL),CR ;End of filespec
248E 1805 01270 JR SHOWFIL ;Dsply name & remove it
01280 ;
01290 ; Stop when ":" hit or terminator
01300 ;
2490 FE3A 01310 CHKDSPC CP ':' ;Already have one ?
2492 23 01320 INC HL
2493 20E8 01330 JR NZ,FLOOP
01340 ;
01350 ; Display "Removing Filespec" & remove it
01360 ;
2495 CD9D24 01370 SHOWFIL CALL SHOWIT ;"Removing:"
2498 01380 @@REMOV ;Remove the file
2498 3E39 00018 LD A,57
249A EF 00019 RST 40
249B 189C 01390 JR REMOV1 ;Loop for more
01400 ;
01410 ; Display Filespec or Devspec
01420 ;
249D 214225 01430 SHOWIT LD HL,SHOW$ ;Show what we are
24A0 D5 01440 PUSH DE ;Save FCB ptr
24A1 01450 @@DSPLY ; removing
00020 IFEQ 00H,1
00021 LD HL,
00022 ENDIF
24A1 3E0A 00023 LD A,10
24A3 EF 00024 RST 40
24A4 D1 01460 POP DE
24A5 2092 01470 JR NZ,REMOV1 ;NZ - abort
24A7 C9 01480 RET ; else RETurn Z
01490 ;
01500 ; Routine to Remove a device
01510 ;
24A8 D5 01520 RMVDVC PUSH DE ;Xfer FCB ptr to IX
24A9 DDE1 01530 POP IX
24AB DD5E01 01540 LD E,(IX+1) ;P/u device name
24AE DD5602 01550 LD D,(IX+2)
01560 ;
01570 ; Routine to find a device in the device tables
01580 ;
24B1 01590 @@GTDCB ;Find this DCB
24B1 3E52 00025 LD A,82
24B3 EF 00026 RST 40
24B4 2083 01600 JR NZ,REMOV1 ;Abort if not found

```

```

01610 ;
01620 ; Found device in tables, is it killable?
01630 ;
24B6 ED534025 01640 DEV4 LD (RENDEV+1),DE ;Try to rename it
24BA E5 01650 PUSH HL ;Save address pointer
24BB 213F25 01660 LD HL,RENDEV ;New & old names same
24BE 5D 01670 LD E,L
24BF 54 01680 LD D,H
24C0 01690 @@RENAM
24C0 3E38 00027 LD A,56
24C2 EF 00028 RST 40
24C3 E1 01700 POP HL ;Rcvr address pointer
24C4 FE13 01710 CP 19 ;If not "illegal name"
24C6 C23924 01720 JP NZ,REMOV1 ; error, is protected
01730 ;
01740 ; Device not system device - is it in use?
01750 ;
24C9 CB5E 01760 CKNILL BIT 3,(HL) ;Test NIL device
24CB 2006 01770 JR NZ,REMOVIT ;OK if NIL (not in use)
24CD 3E27 01780 LD A,27H ; else "device in use"
24CF B7 01790 OR A
24D0 C33924 01800 JP REMOV1 ; and return error
01810 ;
01820 ; Remove the device
01830 ;
24D3 CD0325 01840 REMOVIT CALL SCND4 ;Zero 8-byte DCB area
01850 ;
01860 ; Any other device routes to the removed one?
01870 ;
24D6 114B49 01880 SCNDCB LD DE,'IK' ;Point to begin of area
24D9 01890 @@GTDCB
24D9 3E52 00029 LD A,82
24DB EF 00030 RST 40
24DC CB66 01900 SCND1 BIT 4,(HL) ;Routed device?
24DE 2811 01910 JR Z,SCND2 ;Jump if not
24E0 E5 01920 PUSH HL ;Save DCB pointer
24E1 2C 01930 INC L ;Bypass TYPE code
24E2 7E 01940 LD A,(HL) ;P/u route vector
24E3 2C 01950 INC L ; lo-order
24E4 66 01960 LD H,(HL) ;P/u vector hi-order
24E5 6F 01970 LD L,A
24E6 7E 01980 LD A,(HL) ;P/u TYPE code of nest
24E7 E1 01990 POP HL ;Rcvr DCB pointer
24E8 B7 02000 OR A ;If TYPE=0, we killed
24E9 2006 02010 JR NZ,SCND2 ; the routed device
24EB 3E08 02020 LD A,8
24ED B6 02030 OR (HL) ;Convert to NIL if so
24EE 77 02040 LD (HL),A
24EF 18E5 02050 JR SCNDCB
02060 ;
02070 ; Point to next device
02080 ;
24F1 7D 02090 SCND2 LD A,L ;Advance to next DCB area
24F2 C608 02100 ADD A,8 ;Loop through all
24F4 6F 02110 LD L,A ; devices while checking
24F5 20E5 02120 JR NZ,SCND1 ;Loop until table end
02130 ;
02140 ; Device table cleared, now zero the DCB/FCB
02150 ;
24F7 DDE5 02160 PUSH IX ;Xfer vector to HL
24F9 E1 02170 POP HL

```

```

24FA CD0325 02180 CALL SCND4
24FD CD9D24 02190 CALL SHOWIT ;Show "Removing:"
2500 C33824 02200 JP REMOVE ;Next ...
                02210 ;
2503 0608 02220 SCND4 LD B,8 ;Clear 8 bytes
2505 AF 02230 XOR A
2506 77 02240 SCND5 LD (HL),A ;Go for it
2507 23 02250 INC HL
2508 10FC 02260 DJNZ SCND5
250A C9 02270 RET
                02280 ;
                02290 ; Transfer a filespec from HL to DE
                02300 ;
250B 7E 02310 MOVSPC LD A,(HL) ;P/u a spec character
250C FE2F 02320 CP '/' ;Extension ?
250E 2008 02330 JR NZ,CKSPACE ;No - check if space
2510 23 02340 INC HL ;Is the next character
2511 7E 02350 LD A,(HL) ; valid ?
2512 FE41 02360 CP 'A'
2514 3802 02370 JR C,CKSPACE ;No - don't output it
2516 2B 02380 DEC HL ;Back one
2517 7E 02390 LD A,(HL) ;P/u slash
2518 FE20 02400 CKSPACE CP ' '
251A D8 02410 RET C ;Stop on space or less
251B FE2E 02420 CP '.' ;Password ?
251D 2009 02430 JR NZ,MOVSPC1
251F 23 02440 SKIPPW INC HL
2520 7E 02450 LD A,(HL)
2521 FE20 02460 CP ' '
2523 D8 02470 RET C ;Back on terminator
2524 FE3A 02480 CP ':'
2526 20F7 02490 JR NZ,SKIPPW
2528 EDA0 02500 MOVSPC1 LDI ;Move the char
252A 18DF 02510 JR MOVSPC
                02520 ;
252C 46 02530 SPCREQ$ DB 'File spec required',CR
        69 6C 65 20 73 70 65 63
        20 72 65 71 75 69 72 65
        64 0D
253F 2A 02540 RENDEV DB '*LS'
        4C 53
2542 52 02550 SHOW$ DB 'Removing: '
        65 6D 6F 76 69 6E 67 3A
        20
000F 02560 FILESPC DS 15
255B 00 02570 FCB DB 0
001F 02580 DS 31
        02590 ;
2400 02600 END REMOVE

```

@@1	0000 @@2	0000 @@3	0000
@@4	0000 @MOD2	0000 @MOD4	FFFF
ABB	0010 AP	0027 BEGINA	2409
BREAK	0080 BRKABT	2412 BS	0008
CFLAG\$	0002 CHKDSPC	2490 CKNILL	24C9
CKSPACE	2518 CR	000D DEV4	24B6
DFLAG\$	0003 ERREXIT	2427 ETX	0003
EXIT	2417 FCB	255B FILESPC	254C
FLAG	0040 FLOOP	247D GOODOPN	246E
IOERR	242C KFLAG\$	000A LF	000A
MOVSPC	250B MOVSPC1	2528 NUM	0080
PAR_ERR	002C REMOV	2437 REMOV0	2438
REMOV1	2439 REMOV2	2443 REMOV3	244A
REMOVE	2400 REMOVIT	24D3 RENDEV	253F
RMVDVC	24A8 SAVESP	241A SCND1	24DC
SCND2	24F1 SCND4	2503 SCND5	2506
SCNDCB	24D6 SFLAG\$	0012 SHOW\$	2542
SHOWFIL	2495 SHOWIT	249D SKIPPW	251F
SPCREQ	2421 SPCREQ\$	252C STR	0020
TAB	0009 VFLAG\$	0015 @@ABORT	817D
@@ADTSK	8210 @@BANK	8728 @@BKSP	8408
@@BREAK	873E @@CHNIO	8168 @@CKBRKC	878C
@@CKDRV	8264 @@CKEOF	841D @@CKTSK	81FB
@@CLOSE	83F3 @@CLS	8776 @@CMNDI	81A7
@@CMNDR	81BC @@CTL	7FCC @@DATE	813E
@@DCSTAT	82A3 @@DEBUG	81E6 @@DECHEX	86A8
@@DIRRD	8615 @@DIRWR	862A @@DIV16	8693
@@DIV8	867E @@DODIR	8279 @@DSP	7F90
@@DSPLY	8030 @@ERROR	81D1 @@EXIT	8192
@@FEXT	8582 @@FLAGS	8712 @@FNAME	8597
@@FSPEC	856D @@GATRD	8600 @@GATWR	863F
@@GET	7FA4 @@GTDCB	85C1 @@GTDCT	85AC
@@GTMOD	85D6 @@HDFMT	834B @@HEX16	86E7
@@HEX8	86D2 @@HEXDEC	86BD @@HIGH\$	86FC
@@INIT	83C9 @@KBD	8008 @@KEY	7F7C
@@KEYIN	801C @@KLTSK	824F @@LOAD	8543
@@LOC	8432 @@LOF	8447 @@LOGGER	8067
@@LOGOT	807C @@MSG	80B3 @@MUL16	8669
@@MUL8	8654 @@OPEN	83DE @@PARAM	8129
@@PAUSE	8114 @@PEOF	845C @@POSN	8471
@@PRINT	80C8 @@PRT	7FE0 @@PUT	7FB8
@@RAMDIR	828E @@RDSEC	8321 @@RDSSC	85EB
@@READ	8486 @@REMOV	83B4 @@RENAM	839F
@@REW	849B @@RMTSK	8225 @@RPTSK	823A
@@RREAD	84B0 @@RSLCT	830C @@RSTOR	82CD
@@RUN	8558 @@RWRIT	84C5 @@SEEK	82F7
@@SEEKSC	84DA @@SKIP	84EF @@SLCT	82B8
@@STEPI	82E2 @@TIME	8153 @@VDCTL	80FF
@@VER	8504 @@VRSEC	8336 @@WEOF	8519
@@WHERE	7FF4 @@WRITE	852E @@WRSEC	8360
@@WRSSC	8375 @@WTRK	838A	

2400 is the transfer address
00000 Total errors

NOTES:

Command: RENAME

Library: SYS6/SYS

ISAM # : 53H

```

00100 ;LBRENAME/ASM - RENAME Command
0000 00110 TITLE <RENAME - LS-DOS 6.2>
00120 ;
0000 00130 *GET SVCMAC:3 ;SVC Macro equivalents
00010 ;SVC MAC/ASM - LS-DOS Version VI
00020 *LIST OFF
03900 *LIST ON
0000 00140 *GET VALUES:3 ;Misc. equates
03920 ;VALUES/ASM - Version 6
03930 *LIST OFF
04200 *LIST ON
00150 ;
0000 00160 INH EQU 0 ;Inhibit LRL Fault
00170 ;
2400 00180 ORG 2400H
00190 ;
2400 00200 RENAME @@CKBRKC ;Break key down?
2400 3E6A 00001 LD A,106
2402 EF 00002 RST 40
2403 2804 00210 JR Z,BEGINA ;Ok if not
2405 21FFFF 00220 LD HL,-1 ; else abort
2408 C9 00230 RET
00240 ;
2409 ED733224 00250 BEGINA LD (SAVE SP+1),SP ;Save SP
240D CD3824 00260 CALL RENAM ;Rename File/Device
2410 210000 00270 LD HL,0 ;Init successful
2413 281C 00280 JR Z,SAVE SP ;Z - successful rename
00290 ;
00300 ; I/O Error Processing
00310 ;
2415 6F 00320 IOERR LD L,A ;Error # to HL
2416 2600 00330 LD H,0
2418 F6C0 00340 OR 0C0H ;Set to brief & return
241A 4F 00350 LD C,A ;Xfer error code
241B 00360 @@ERROR
241B 3E1A 00003 LD A,26
241D EF 00004 RST 40
241E 1811 00370 JR SAVE SP ;Restore stack & RET
00380 ;
00390 ; Internal Message Error Processing
00400 ;
2420 21C325 00410 SPCERR LD HL,SPCERR$
2423 DD 00420 DB 0DDH
2424 21D725 00430 DUPNAM LD HL,DUPNAM$
2427 DD 00440 DB 0DDH
2428 21EB25 00450 TOWHAT LD HL,TOWHAT$
242B 00460 @@LOGOT
00005 IFEQ 00H,1
00006 LD HL,
00007 ENDIF
242B 3E0C 00008 LD A,12
242D EF 00009 RST 40
242E 21FFFF 00470 LD HL,-1
00480 ;
00490 ; Clean up stack & clear any pending <BREAK>s
00500 ;
2431 310000 00510 SAVE SP LD SP,$-$ ;P/u original SP
2434 00520 @@CKBRKC ;Clear any <BREAK>
2434 3E6A 00010 LD A,106
2436 EF 00011 RST 40
2437 C9 00530 RET

```

```

00540 ;
00550 ;
00560 ;
2438 E5 00570 RENAM PUSH HL ;Save cmd line ptr
2439 117526 00580 LD DE,TEMPFCB ;Xfer Filespec to buffer
243C 00590 @@FSPEC
243C 3E4E 00012 LD A,78
243E EF 00013 RST 40
243F E1 00600 POP HL ;Ignore error
2440 00610 @@FLAGS ;IY => Flag Table
2440 3E65 00014 LD A,101
2442 EF 00015 RST 40
2443 112326 00620 LD DE,OLDFCB ;Get filespec
2446 00630 @@FSPEC
2446 3E4E 00016 LD A,78
2448 EF 00017 RST 40
2449 20D5 00640 JR NZ,SPCERR ;Quit if bad source name
244B 110326 00650 LD DE,NEWFCB ;Get new name
244E 00660 @@FSPEC
244E 3E4E 00018 LD A,78
2450 EF 00019 RST 40
2451 C43C25 00670 CALL NZ,CVRTUC ;Cvrt partial spec to UC
2454 3A0326 00680 REN1 LD A,(NEWFCB) ;If new name starts out
2457 FE0F 00690 CP CR+2 ; with something less
2459 DA2824 00700 JP C,TOWHAT ; than X'0E', to what ?
245C 212326 00710 LD HL,OLDFCB
245F 110326 00720 LD DE,NEWFCB
2462 7E 00730 LD A,(HL) ;Check on device rename
2463 FE2A 00740 CP '*'
2465 CA0625 00750 JP Z,DEVREN
2468 1A 00760 LD A,(DE) ;Old is file, new must
2469 FE2A 00770 CP '*' ; be also
246B 28B3 00780 JR Z,SPCERR
00790 ;
00800 ;
00810 ;
246D 117526 00820 LD DE,TEMPFCB ;Can we OPEN it ?
2470 FDCB12C6 00830 SET INH,(IY+SFLAG$) ;Inhibit open bit set
2474 00840 @@OPEN
2474 3E3B 00020 LD A,59
2476 EF 00021 RST 40
2477 C0 00850 RET NZ ;NZ - "File not Found"
2478 ED4B7B26 00860 LD BC,(TEMPFCB+6) ;P/u drive #/DEC
00870 ;
00880 ;
00890 ;
247C E5 00900 PUSH HL ;Save ptr
247D 7E 00910 FLOOP LD A,(HL) ;P/u char
247E FE0E 00920 CP CR+1 ;End of Filespec ?
2480 300F 00930 JR NC,CHKDSPC
00940 ;
00950 ;
00960 ;
2482 363A 00970 LD (HL),':' ;Append drivespec onto
2484 23 00980 INC HL ; end of filespec
2485 79 00990 LD A,C ;Xfer drive # to A
2486 C630 01000 ADD A,'0' ;Convert to ASCII
2488 77 01010 LD (HL),A
2489 32A924 01020 LD (OLD_DRV+1),A ;Self-modify NEW FCB
248C 23 01030 INC HL ;Bump
248D 360D 01040 LD (HL),CR ;End of filespec

```



```

248F 1809      01050      JR      DOMATCH      ;Get defaults
                01060      ;
                01070      ;      Stop when ":" hit or terminator
                01080      ;
2491 FE3A      01090      CHKDSPC CP      ':'      ;Already have one ?
2493 23        01100      INC      HL
2494 20E7      01110      JR      NZ,FLOOP
2496 7E        01120      LD      A,(HL)      ;P/u drive #
2497 32A924    01130      LD      (OLD_DRV+1),A ;Self-modify NEW FCB
249A E1        01140      DOMATCH POP     HL      ;HL => Old FCB
249B 110326    01150      LD      DE,NEWFCB    ;DE => New FCB
249E CD6125    01160      CALL     MATCH
                01170      ;
                01180      ;      Make sure NEW drivespec is same as OLD one
                01190      ;
24A1 D5        01200      PUSH     DE      ;Save New
24A2 1A        01210      F2LOOP LD      A,(DE) ;Go until ":"
24A3 13        01220      INC      DE
24A4 FE3A      01230      CP      ':'
24A6 20FA      01240      JR      NZ,F2LOOP
24A8 3E00      01250      OLD_DRV LD      A,$-$ ;P/u OLD drivespec
24AA 12        01260      LD      (DE),A      ;Overwrite
24AB D1        01270      POP      DE      ;Restore DE
                01280      ;
                01290      ;      Does the NEW filename already exist ?
                01300      ;
24AC E5        01310      PUSH     HL      ;Save OLD ptr
24AD D5        01320      PUSH     DE      ;Save NEW ptr
24AE EB        01330      EX      DE,HL
24AF 117526    01340      LD      DE,TEMPFCB ;DE => Temp buffer
24B2 FDCB12C6  01350      SET      INH,(IY+SFLAG$)
24B6           01360      @@FSPEC      ;Xfer filespec
24B6 3E4E      00022      LD      A,78
24B8 EF        00023      RST      40
24B9           01370      @@OPEN      ;File already exist ?
24B9 3E3B      00024      LD      A,59
24BB EF        00025      RST      40
24BC CA2424    01380      JP      Z,DUPNAM ;Error if so
24BF D1        01390      POP      DE      ;Restore ptrs
24C0 E1        01400      POP      HL
24C1 E5        01410      REN2  PUSH     HL      ;OLD Filename/Device
24C2 D5        01420      PUSH     DE      ;NEW Filename/Device
                01430      ;
                01440      ;      Xfer the OLD & NEW specs to SPEC$ minus PASSWORD
                01450      ;
24C3 114D26    01460      LD      DE,SPEC$
24C6 CDE524    01470      CALL     MOVSPC ;Move the OLD spec
24C9 21FE25    01480      LD      HL,TO$
24CC 010400    01490      LD      BC,4
24CF EDB0      01500      LDIR      ;Move ' to '
24D1 E1        01510      POP      HL      ;Recover NEW spec
24D2 E5        01520      PUSH     HL
24D3 CDE524    01530      CALL     MOVSPC ;Move the NEW spec
24D6 3E0D      01540      LD      A,CR
24D8 12        01550      LD      (DE),A ;Terminate with CR
24D9           01560      @@LOGOT RENAM$ ;Send names to video
                00026      IFEQ     01H,1
24D9 214326    00027      LD      HL,RENAM$
                00028      ENDIF
24DC 3E0C      00029      LD      A,12
24DE EF        00030      RST      40

```

```

24DF E1      01570      POP      HL      ;Recover new
24E0 D1      01580      POP      DE      ;Recover old
24E1         01590      @@RENAM      ;Rename file
24E1 3E38    00031      LD        A,56
24E3 EF      00032      RST       40
24E4 C9      01600      RET              ;Return with condition
          01610 ;
          01620 ;      MOVSPC - Create Secondary Spec
          01630 ;
24E5 7E      01640 MOVSPC LD      A,(HL)      ;P/u a spec character
24E6 FE2F    01650      CP        '/'      ;Extension ?
24E8 2008    01660      JR        NZ,CKSPACE ;No - check if space
24EA 23      01670      INC       HL        ;Is the next character
24EB 7E      01680      LD        A,(HL)      ; valid ?
24EC FE41    01690      CP        'A'
24EE 3802    01700      JR        C,CKSPACE  ;No - don't output it
24F0 2B      01710      DEC       HL        ;Back one
24F1 7E      01720      LD        A,(HL)      ;P/u slash
24F2 FE20    01730 CKSPACE CP      ' '
24F4 D8      01740      RET       C          ;Exit on terminator
24F5 FE2E    01750      CP        '.'      ;If password, ignore it
24F7 2009    01760      JR        NZ,MOVSPC1
24F9 23      01770 SKIPPW INC      HL
24FA 7E      01780      LD        A,(HL)
24FB FE20    01790      CP        ' '
24FD D8      01800      RET       C          ;Back on terminator
24FE FE3A    01810      CP        ':'
2500 20F7    01820      JR        NZ,SKIPPW
2502 EDA0    01830 MOVSPC1 LDI             ;Move the char
2504 18DF    01840      JR        MOVSPC
          01850 ;
          01860 ;      Routine to rename a device
          01870 ;
2506 1A      01880 DEVREN LD      A,(DE)      ;Old was device, new must
2507 FE2A    01890      CP        '*'      ; also be a device spec
2509 C22024  01900      JP        NZ,SPCERR   ;Abort if bad
          01910 ;
          01920 ;      Does the Source Devspec exist ?
          01930 ;
250C E5      01940      PUSH     HL        ;Save Old Device name
250D D5      01950      PUSH     DE        ;Save New Device name
250E 23      01960      INC       HL        ;Bump past "*"
250F 5E      01970      LD        E,(HL)      ;Set DE = Device name
2510 23      01980      INC       HL
2511 56      01990      LD        D,(HL)
2512         02000      @@GTDCB      ;Does it exist ?
2512 3E52    00033      LD        A,82
2514 EF      00034      RST       40
2515 C21524  02010      JP        NZ,IOERR    ;NZ - "Dev not Available"
          02020 ;
          02030 ;      P/u the Job Log DCB Address (last DCB)
          02040 ;
2518 44      02050      LD        B,H        ;Save DCB ptr in BC
2519 4D      02060      LD        C,L
251A 114A4C  02070      LD        DE,'LJ'      ;Find *JL
251D         02080      @@GTDCB
251D 3E52    00035      LD        A,82
251F EF      00036      RST       40
2520 23      02090      INC       HL        ;Pt HL => Past Protected
2521 B7      02100      OR        A          ; system Device table.
2522 ED42    02110      SBC       HL,BC      ;Protected Device ?

```

```

2524 3E28      02120      LD      A,40      ;Init errcode
2526 D21524    02130      JP      NC,IOERR    ;Jump on error
                02140      ;
                02150      ;      Does the destination device already exist ?
                02160      ;
2529 E1        02170      POP      HL      ;HL => New Devspec
252A E5        02180      PUSH     HL
252B 23        02190      INC      HL      ;Bump past "*"
252C 5E        02200      LD      E,(HL)    ;Set DE = Device name
252D 23        02210      INC      HL
252E 56        02220      LD      D,(HL)
252F          02230      @@GTDCB      ;Already Exist ?
252F 3E52      00037      LD      A,82
2531 EF        00038      RST      40
2532 3E27      02240      LD      A,39      ;Yes - Device in use
2534 CA1524    02250      JP      Z,IOERR
2537 D1        02260      POP      DE      ;Restore NEW & OLD ptrs
2538 E1        02270      POP      HL
2539 C3C124    02280      JP      REN2
                02290      ;
                02300      ;      Routine xfers partial filespec & cvrts to UC
                02310      ;
253C 7E        02320      CVRTUC  LD      A,(HL)
253D FE0D      02330      CP      CR
253F C8        02340      RET      Z      ;Ret if no new name
2540 2B        02350      DEC      HL      ;Backup to 1st separator
2541 7E        02360      COP0   LD      A,(HL)
2542 23        02370      INC      HL
2543 FE20      02380      CP      ' '      ;Skip past spaces
2545 28FA      02390      JR      Z,COP0
2547 2B        02400      DEC      HL
2548 0620      02410      LD      B,32      ;Max 32 chars
254A 7E        02420      COP1   LD      A,(HL)    ;Transfer the partial
254B FE61      02430      COP2   CP      'a'      ;Cvrt lc <a-z> to uc
254D 3806      02440      JR      C,COP3
254F FE7B      02450      CP      'z'+1
2551 3002      02460      JR      NC,COP3
2553 D620      02470      SUB      20H
2555 12        02480      COP3   LD      (DE),A    ;Filespec until paren
2556 FE0D      02490      CP      CR      ; or <ENTER>
2558 C8        02500      RET      Z
2559 FE28      02510      CP      '('
255B C8        02520      RET      Z
255C 23        02530      INC      HL      ; or end-of-line
255D 13        02540      INC      DE      ; or 32 chars max
255E 10EA      02550      DJNZ    COP1
2560 C9        02560      RET
                02570      ;
                02580      ;      Match source & destination for defaults
                02590      ;
2561 D5        02600      MATCH  PUSH     DE      ;Save NEW spec
2562 E5        02610      PUSH     HL      ;Save OLD spec
2563 1A        02620      LD      A,(DE)    ;P/u a dest character
2564 FE41      02630      CP      'A'
2566 DCA325    02640      CALL    C,MATCH7    ;Match if not a filename
2569 062F      02650      LD      B,'/'
256B CD7C25    02660      CALL    MATCH2
256E 063A      02670      LD      B,':'
2570 CD7C25    02680      CALL    MATCH2
2573 062E      02690      LD      B, '.'
2575 CD7C25    02700      CALL    MATCH2

```

The Source	LIBRARY Files	RENAME - LS-DOS 6.2	Page 00006
2578 E1	02710	POP HL	
2579 D1	02720	POP DE	
257A C9	02730	RET	
	02740 ;		
257B 13	02750 MATCH1	INC DE	
257C 1A	02760 MATCH2	LD A,(DE)	;Scan destination until
257D B8	02770	CP B	; the test character is
257E 280E	02780	JR Z,MATCH3	; found or until some
2580 FE41	02790	CP 'A'	; other special char
2582 30F7	02800	JR NC,MATCH1	; is reached
2584 FE30	02810	CP '0'	;Loop on <0-9>
2586 3808	02820	JR C,MATCH4	
2588 FE3A	02830	CP '9'+1	
258A 38EF	02840	JR C,MATCH1	
258C 1802	02850	JR MATCH4	
258E 13	02860 MATCH3	INC DE	
258F C9	02870	RET	
	02880 ;		
	02890 ;	Found some other special char - Need the field	
	02900 ;		
2590 E5	02910 MATCH4	PUSH HL	;Save pointer to source
2591 7E	02920 MATCH5	LD A,(HL)	;Scan source until the
2592 23	02930	INC HL	; desired field is
2593 FE03	02940	CP ETX	; found (if it is
2595 280A	02950	JR Z,MATCH6	; supplied by the user)
2597 FE0D	02960	CP CR	
2599 2806	02970	JR Z,MATCH6	
259B B8	02980	CP B	
259C 20F3	02990	JR NZ,MATCH5	
259E CDAF 25	03000	CALL MATCH9	;Move source field
25A1 E1	03010 MATCH6	POP HL	
25A2 C9	03020	RET	
	03030 ;		
	03040 ;	Routines to move a source field to destination	
	03050 ;		
25A3 7E	03060 MATCH7	LD A,(HL)	;P/u source character
25A4 FE30	03070	CP '0'	;Back when out of range
25A6 D8	03080	RET C	
25A7 FE3A	03090	CP '9'+1	
25A9 3803	03100	JR C,MATCH8	
25AB FE41	03110	CP 'A'	
25AD D8	03120	RET C	
25AE 23	03130 MATCH8	INC HL	;Advance source ptr
25AF E5	03140 MATCH9	PUSH HL	;Save HL and make it
25B0 62	03150	LD H,D	; the destination ptr
25B1 6B	03160	LD L,E	
25B2 4E	03170 MATCH10	LD C,(HL)	;Get char at destination
25B3 77	03180	LD (HL),A	; and put in new one
25B4 23	03190	INC HL	;Next dest loc.
25B5 79	03200	LD A,C	;What was there?
25B6 FE03	03210	CP ETX	;Go until ETX
25B8 2804	03220	JR Z,MATCH11	
25BA FE0D	03230	CP CR	; or end of line
25BC 20F4	03240	JR NZ,MATCH10	
25BE 77	03250 MATCH11	LD (HL),A	
25BF E1	03260	POP HL	
25C0 13	03270	INC DE	
25C1 18E0	03280	JR MATCH7	
	03290 ;		
25C3 53	03300 SPCERR\$ DB	'Specification error',CR	
70 65 63 69 66 69 63 61			

```

      74 69 6F 6E 20 65 72 72
      6F 72 0D
25D7 44      03310 DUPNAM$ DB      'Duplicate file name',CR
      75 70 6C 69 63 61 74 65
      20 66 69 6C 65 20 6E 61
      6D 65 0D
25EB 52      03320 TOWHAT$ DB      'Rename it to what?',CR
      65 6E 61 6D 65 20 69 74
      20 74 6F 20 77 68 61 74
      3F 0D
25FE 20      03330 TO$      DB      ' to ',ETX
      74 6F 20 03
2603 0D      03340 NEWFCB  DB      CR
001F      03350      DS      31      ;Init to cr
0020      03360 OLDFCB  DS      32
2643 52      03370 RENAM$  DB      'Renaming: '
      65 6E 61 6D 69 6E 67 3A
      20
0028      03380 SPECS$  DS      40
0020      03390 TEMPFCB DS      32
2695 0000    03400 OLD FIL DW      0
2697      03410 LAST    EQU      $
      03420 ;
2400      03430      END      RENAME

```

@@1	0000	@@2	0000	@@3	0000
@@4	0000	@MOD2	0000	@MOD4	FFFF
ABB	0010	AP	0027	BEGINA	2409
BREAK	0080	BS	0008	CFLAG\$	0002
CHKDSPC	2491	CKSPACE	24F2	COP0	2541
COP1	254A	COP2	254B	COP3	2555
CR	000D	CVRTUC	253C	DEVREN	2506
DFLAG\$	0003	DOMATCH	249A	DUPNAM	2424
DUPNAM\$	25D7	ETX	0003	F2LOOP	24A2
FLAG	0040	FLOOP	247D	INH	0000
IOERR	2415	KFLAG\$	000A	LAST	2697
LF	000A	MATCH	2561	MATCH1	257B
MATCH10	25B2	MATCH11	25BE	MATCH2	257C
MATCH3	258E	MATCH4	2590	MATCH5	2591
MATCH6	25A1	MATCH7	25A3	MATCH8	25AE
MATCH9	25AF	MOVSPC	24E5	MOVSPC1	2502
NEWFCB	2603	NUM	0080	OLDFCB	2623
OLD DRV	24A8	OLD FIL	2695	PAR ERR	002C
RENT	2454	RENZ	24C1	RENAM	2438
RENAM\$	2643	RENAME	2400	SAVESP	2431
SFLAG\$	0012	SKIPPW	24F9	SPCERR	2420
SPCERR\$	25C3	SPECS\$	264D	STR	0020
TAB	0009	TEMPFCB	2675	TO\$	25FE
TOWHAT	2428	TOWHAT\$	25EB	VFLAG\$	0015
@@ABORT	890E	@@ADTSK	89A1	@@BANK	8EB9
@@BKSP	8B99	@@BREAK	8ECF	@@CHNIO	88F9
@@CKBRKC	8F1D	@@CKDRV	89F5	@@CKEOF	8BAE
@@CKTSK	898C	@@CLOSE	8B84	@@CLS	8F07
@@CMNDI	8938	@@CMNDR	894D	@@CTL	875D
@@DATE	88CF	@@DCSTAT	8A34	@@DEBUG	8977
@@DECHEX	8E39	@@DIRRD	8DA6	@@DIRWR	8DBB
@@DIV16	8E24	@@DIV8	8E0F	@@DODIR	8A0A
@@DSP	8721	@@DSPLY	87C1	@@ERROR	8962
@@EXIT	8923	@@FEXT	8D13	@@FLAGS	8EA3
@@FNAME	8D28	@@FSPEC	8CFE	@@GATRD	8D91
@@GATWR	8DD0	@@GET	8735	@@GTDCB	8D52
@@GTDCT	8D3D	@@GTMOD	8D67	@@HDFMT	8ADC
@@HEX16	8E78	@@HEX8	8E63	@@HEXDEC	8E4E
@@HIGH\$	8E8D	@@INIT	8B5A	@@KBD	8799
@@KEY	870D	@@KEYIN	87AD	@@KLTSK	89E0
@@LOAD	8CD4	@@LOC	8BC3	@@LOF	8BD8
@@LOGGER	87F8	@@LOGOT	880D	@@MSG	8844
@@MUL16	8DFA	@@MUL8	8DE5	@@OPEN	8B6F
@@PARAM	88BA	@@PAUSE	88A5	@@PEOF	8BED
@@POSN	8C02	@@PRINT	8859	@@PRT	8771
@@PUT	8749	@@RAMDIR	8A1F	@@RDSEC	8AB2
@@RDSSC	8D7C	@@READ	8C17	@@REMOV	8B45
@@RENAM	8B30	@@REW	8C2C	@@RMTSK	89B6
@@RPTSK	89CB	@@RREAD	8C41	@@RSLCT	8A9D
@@RSTOR	8A5E	@@RUN	8CE9	@@RWIT	8C56
@@SEEK	8A88	@@SEEKSC	8C6B	@@SKIP	8C80
@@SLCT	8A49	@@STEPI	8A73	@@TIME	88E4
@@VDCTL	8890	@@VER	8C95	@@VRSEC	8AC7
@@WEOF	8CAA	@@WHERE	8785	@@WRITE	8CBF
@@WRSEC	8AF1	@@WRSSC	8B06	@@WRTRK	8B1B

2400 is the transfer address
 00000 Total errors

NOTES:

Command: RESET

Library: SYS6/SYS

ISAM # : 63H


```

00100 ;LBRESET/ASM - RESET Command
0000 00110 TITLE <RESET - LS-DOS 6.2>
00120 ;
000D 00130 CR EQU 13
0000 00140 *GET SVCMAC:3 ;SVC Macro equivalents
00010 ;SVMAC/ASM - LS-DOS Version VI
00020 *LIST OFF
03900 *LIST ON
00150 ;
2400 00160 ORG 2400H
00170 ;
00180 ; Save stack & call reset code
00190 ;
2400 ED732124 00200 RESET LD (SAVEP+1),SP ;Save Stack ptr
2404 CD2724 00210 CALL RESET1 ;Call Reset routine
2407 210000 00220 LD HL,0 ;Init successful
240A 2814 00230 JR Z,SAVEP ;Z - Exit clean
00240 ;
00250 ; I/O Error Processing
00260 ;
240C 6F 00270 IOERR LD L,A ;Error # to HL
240D 2600 00280 LD H,0
240F F6C0 00290 OR 0C0H ;Abbrev, return
2411 4F 00300 LD C,A
2412 00310 @@ERROR
2412 3E1A 00001 LD A,26
2414 EF 00002 RST 40
2415 1809 00320 JR SAVEP ;P/u stack & return
00330 ;
00340 ; Internal Message Exit
00350 ;
2417 21F824 00360 SPCREQ LD HL,SPCREQ$ ;"Devspec req..."
241A 00370 @@LOGOT
00003 IFEQ 00H,1
00004 LD HL,
00005 ENDIF
241A 3E0C 00006 LD A,12
241C EF 00007 RST 40
241D 21FFFF 00380 LD HL,-1 ;Set abort code
00390 ;
00400 ; P/u stack & clear any pending <BREAK>
00410 ;
2420 310000 00420 SAVESP LD SP,$-$ ;P/u stack ptr
2423 00430 @@CKBRKC ;Clear any Break
2423 3E6A 00008 LD A,106
2425 EF 00009 RST 40
2426 C9 00440 RET
00450 ;
00460 ; RESET1 - Reset a Filespec or Devspec
00470 ;
2427 112D25 00480 RESET1 LD DE,FCBDEV ;Get file/device spec
242A 00490 @@FSPEC
242A 3E4E 00010 LD A,78
242C EF 00011 RST 40
242D C21724 00500 JP NZ,SPCREQ ;Must reset something
2430 1A 00510 LD A,(DE) ;File reset used to
2431 FE2A 00520 CP '*' ; reset the
2433 C26624 00530 JP NZ,RESFIL ; "file open bit
2436 ED5B2E25 00540 LD DE,(FCBDEV+1) ;P/u the device name
243A 00550 @@GTDCB ;Find in device tables
243A 3E52 00012 LD A,82

```

The Source	LIBRARY Files	RESET - LS-DOS 6.2	Page 00002
243C EF	00013	RST 40	
243D C0	00560	RET NZ	;NZ - "Device not Avail"
243E E5	00570	PUSH HL	;Save pointer to table
243F CD8124	00580	CALL CLSFILS	;Reset routes to files
2442 E1	00590	POP HL	;Get DCB pointer
2443 C0	00600	RET NZ	;NZ - I/O Error
	00610 ;		
	00620 ;	Unhook the device chain	
	00630 ;		
2444 E5	00640	PUSH HL	
2445 F3	00650	DI	;Don't stop me now
2446 CD9D24	00660	CALL FIXDCB	;Fixup the DCB
2449 FB	00670	EI	;Now you can interrupt
244A 212D25	00680	LD HL,FCBDEV	;Determine if system
244D 54	00690	LD D,H	; device by attempting
244E 5D	00700	LD E,L	; to rename it
244F	00710	@@RENAM	;The error code will be
244F 3E38	00014	LD A,56	
2451 EF	00015	RST 40	
2452 E1	00720	POP HL	; either 19 or 40
2453 FE28	00730	CP 40	;Protected system device?
2455 2804	00740	JR Z,SYSDVC	
2457 3608	00750	LD (HL),8	;Show device is NIL
2459 AF	00760	XOR A	;Set Z
245A C9	00770	RET	;Z - successful
	00780 ;		
	00790 ;	RESET of system device	
	00800 ;		
245B E5	00810 SYSDVC	PUSH HL	;Save DCB ptr
245C 2C	00820	INC L	;If DCB vector is X'0000'
245D 7E	00830	LD A,(HL)	; then do NOT reset the
245E 2C	00840	INC L	; NIL bit
245F B6	00850	OR (HL)	
2460 E1	00860	POP HL	
2461 C8	00870	RET Z	;Z - return
2462 CB9E	00880	RES 3,(HL)	;Make sure NIL is off
2464 AF	00890	XOR A	;Set Z & Return
2465 C9	00900	RET	
	00910 ;		
	00920 ;	Reset the "file open bit" of a file	
	00930 ;		
2466	00940 RESFIL	@@FLAGS	;Get flag table pointer
2466 3E65	00016	LD A,101	
2468 EF	00017	RST 40	
2469 FDCB12C6	00950	SET 0,(IY+'S'-'A')	;Inhibit file open bit
246D	00960	@@OPEN	
246D 3E3B	00018	LD A,59	
246F EF	00019	RST 40	
2470 C0	00970	RET NZ	;NZ - I/O Error
2471 7E	00980	LD A,(HL)	;Make sure access level
2472 E607	00990	AND 7	; is at least UPDATE
2474 FE05	01000	CP 5	
2476 3E25	01010	LD A,37	;Init "Illegal access..."
2478 C0	01020	RET NZ	;NZ - I/O error
2479 EB	01030	EX DE,HL	
247A CBF6	01040	SET 6,(HL)	;Set "close authority
247C EB	01050	EX DE,HL	; to reset dir bit
247D	01060	@@CLOSE	
247D 3E3C	00020	LD A,60	
247F EF	00021	RST 40	
2480 C9	01070	RET	;Return w/ condition

```

01080 ;
01090 ; Find the last device route & close any open file
01100 ;
2481 CB66 01110 CLSFILS BIT 4,(HL) ;Jump if no route
2483 2807 01120 JR Z,CLSFIL1
2485 23 01130 INC HL ;Else p/u link address
2486 7E 01140 LD A,(HL) ; and test that one
2487 23 01150 INC HL ; for a chain
2488 66 01160 LD H,(HL)
2489 6F 01170 LD L,A
248A 18F5 01180 JR CLSFILS
248C CB7E 01190 CLSFIL1 BIT 7,(HL) ;A file?
248E C8 01200 RET Z ;Ret if not
248F 114D25 01210 LD DE,FCBFIL ;Pt to fcb area
2492 D5 01220 PUSH DE
2493 012000 01230 LD BC,32
2496 EDB0 01240 LDIR ;Fill from device vector
2498 D1 01250 POP DE ;Recover start
2499 01260 @@CLOSE ;Close the file
2499 3E3C 00022 LD A,60
249B EF 00023 RST 40
249C C9 01270 RET ;Ret with Z, NZ status
01280 ;
01290 ; Routine to fix up a system DCB
01300 ;
249D CB66 01310 FIXDCB BIT 4,(HL) ;If routed, recover the
249F 2810 01320 JR Z,FIX1 ; original data from
24A1 E5 01330 PUSH HL ; DCB+3 to DCB+5
24A2 54 01340 LD D,H ;DCB start to DE
24A3 5D 01350 LD E,L
24A4 2C 01360 FIX0 INC L ;Pt to old stored
24A5 2C 01370 INC L ; information
24A6 2C 01380 INC L
24A7 010300 01390 LD BC,3
24AA EDB0 01400 LDIR ;Xfer to DCB
24AC E1 01410 POP HL
24AD CBA6 01420 RES 4,(HL) ;Reset Routed bit
24AF 18EC 01430 JR FIXDCB
24B1 CB6E 01440 FIX1 BIT 5,(HL) ;If linked, recover the
24B3 2819 01450 JR Z,FIX2 ; original data from
24B5 E5 01460 PUSH HL ; the LINK DCB source &
24B6 2C 01470 INC L ; clear the LINK DCB
24B7 5E 01480 LD E,(HL) ;P/u the LINK vector
24B8 2C 01490 INC L
24B9 56 01500 LD D,(HL)
24BA E1 01510 POP HL ;Recover DCB ptr
24BB E5 01520 PUSH HL
24BC EB 01530 EX DE,HL ;Link to HL, DCB to DE
24BD E5 01540 PUSH HL ;Save for clearing
24BE 010300 01550 LD BC,3
24C1 EDB0 01560 LDIR
24C3 E1 01570 POP HL
24C4 0608 01580 LD B,8
24C6 3600 01590 FIX1A LD (HL),0 ;Clear the LINK DCB
24C8 2C 01600 INC L
24C9 10FB 01610 DJNZ FIX1A
24CB E1 01620 POP HL ;Rcvr DCB pointer
24CC 18CF 01630 JR FIXDCB
24CE CB76 01640 FIX2 BIT 6,(HL) ;If filtered, recover the
24D0 C8 01650 RET Z ; original data by
24D1 E5 01660 PUSH HL ; swapping back the

```

The Source	LIBRARY Files	RESET - LS-DOS 6.2	Page 00004
24D2 54	01670	LD D,H	
24D3 5D	01680	LD E,L	
24D4 2C	01690	INC L	; 1st three bytes with
24D5 7E	01700	LD A,(HL)	; the FILTER DCB
24D6 2C	01710	INC L	
24D7 66	01720	LD H,(HL)	
24D8 6F	01730	LD L,A	
24D9 010400	01740	LD BC,4	;HL now points to the
24DC 09	01750	ADD HL,BC	; entry point. Get its
24DD 4E	01760	LD C,(HL)	; DCB address by peeking
24DE 0C	01770	INC C	; past the name field
24DF 09	01780	ADD HL,BC	
24E0 7E	01790	LD A,(HL)	;Get low-order
24E1 23	01800	INC HL	
24E2 66	01810	LD H,(HL)	;Get hi-order
24E3 6F	01820	LD L,A	
24E4 E5	01830	PUSH HL	;If DCB is itself, then
24E5 ED52	01840	SBC HL,DE	; bring in the NIL
24E7 E1	01850	POP HL	
24E8 28BA	01860	JR Z,FIX0	
24EA 0603	01870	LD B,3	; else swap the 1st three
24EC 4E	01880	LD C,(HL)	; bytes of the DCBs
24ED 1A	01890	LD A,(DE)	
24EE 77	01900	LD (HL),A	
24EF 79	01910	LD A,C	
24F0 12	01920	LD (DE),A	
24F1 2C	01930	INC L	
24F2 1C	01940	INC E	
24F3 10F7	01950	DJNZ FIX2A	
24F5 E1	01960	POP HL	
24F6 18A5	01970	JR FIXDCB	
	01980 ;		
	01990 ;	Data area	
	02000 ;		
24F8 44	02010	SPCREQ\$ DB	'Device spec required',CR
65 76 69	63 65 20 73 70		
65 63 20	72 65 71 75 69		
72 65 64	00		
250D 00	02020	DC	32,0 ;Patch space
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		
252D 00	02030	FCBDEV DB	0
001F	02040	DS	31
254D 00	02050	FCBFIL DB	0
001F	02060	DS	31
	02070 ;		
2400	02080	END	RESET

@@1	0000 @@2	0000 @@3	0000
@@4	0000 @MOD2	0000 @MOD4	FFFF
CLSFIL1	248C CLSFILS	2481 CR	000D
FCBDEV	252D FCBFIL	254D FIX0	24A4
FIX1	24B1 FIX1A	24C6 FIX2	24CE
FIX2A	24EC FIXDCB	249D IOERR	240C
RESET	2400 RESET1	2427 RESFIL	2466
SAVESP	2420 SPCREQ	2417 SPCREQ\$	24F8
SYSDVC	245B @@ABORT	7D6A @@ADTSK	7DFD
@@BANK	8315 @@BKSP	7FF5 @@BREAK	832B
@@CHNIO	7D55 @@CKBRKC	8379 @@CKDRV	7E51
@@CKEOF	800A @@CKTSK	7DE8 @@CLOSE	7FE0
@@CLS	8363 @@CMNDI	7D94 @@CMNDR	7DA9
@@CTL	7BB9 @@DATE	7D2B @@DCSTAT	7E90
@@DEBUG	7DD3 @@DECHEX	8295 @@DIRRD	8202
@@DIRWR	8217 @@DIV16	8280 @@DIV8	826B
@@DODIR	7E66 @@DSP	7B7D @@DSPLY	7C1D
@@ERROR	7DBE @@EXIT	7D7F @@FEXT	816F
@@FLAGS	82FF @@FNAME	8184 @@FSPEC	815A
@@GATRD	81ED @@GATWR	822C @@GET	7B91
@@GTDCB	81AE @@GTDCCT	8199 @@GTMOD	81C3
@@HDFMT	7F38 @@HEX16	82D4 @@HEX8	82BF
@@HEXDEC	82AA @@HIGH\$	82E9 @@INIT	7FB6
@@KBD	7BF5 @@KEY	7B69 @@KEYIN	7C09
@@KLTSK	7E3C @@LOAD	8130 @@LOC	801F
@@LOF	8034 @@LOGGER	7C54 @@LOGOT	7C69
@@MSG	7CA0 @@MUL16	8256 @@MUL8	8241
@@OPEN	7FCB @@PARAM	7D16 @@PAUSE	7D01
@@PEOF	8049 @@POSN	805E @@PRINT	7CB5
@@PRT	7BCD @@PUT	7BA5 @@RAMDIR	7E7B
@@RDSEC	7F0E @@RDSSC	81D8 @@READ	8073
@@REMOV	7FA1 @@RENAM	7F8C @@REW	8088
@@RMTSK	7E12 @@RPTSK	7E27 @@RREAD	809D
@@RSLCT	7EF9 @@RSTOR	7EBA @@RUN	8145
@@RWRTIT	80B2 @@SEEK	7EE4 @@SEEKSC	80C7
@@SKIP	80DC @@SLCT	7EA5 @@STEPI	7ECF
@@TIME	7D40 @@VDCTL	7CEC @@VER	80F1
@@VRSEC	7F23 @@WEOF	8106 @@WHERE	7BE1
@@WRITE	811B @@WRSEC	7F4D @@WRSSC	7F62
@@WRTRK	7F77		

2400 is the transfer address

00000 Total errors

NOTES:

NOTES:

Command: ROUTE

Library: SYS6/SYS

ISAM # : 64H


```

00100 ;LBROUTE/ASM - ROUTE Command
0000 00110 TITLE <ROUTE - LS-DOS 6.2>
00120 ;
000D 00130 CR EQU 13
002C 00140 PAR_ERR EQU 44 ;Parameter Error
00150 ;
0000 00160 *GET SVCMAC:3 ;SVC Macro equivalents
00010 ;SVMAC/ASM - LS-DOS Version VI
00020 *LIST OFF
03900 *LIST ON
00170 ;
2400 00180 ORG 2400H
00190 ;
00200 ; Save stack & call Route routine
00210 ;
2400 ED732724 00220 ROUTE LD (SAVE$P+1),SP ;Save Stack
2404 CD2D24 00230 CALL ROUTE1 ;Call route routine
2407 210000 00240 EXIT LD HL,0 ;Clean Exit
240A 181A 00250 JR SAVESP
00260 ;
00270 ; I/O Error Handling
00280 ;
240C 3E2C 00290 PRMERR LD A,PAR_ERR ;Parameter Error
240E 6F 00300 IOERR LD L,A
240F 2600 00310 LD H,0
2411 F6C0 00320 OR C,00H ;Set abbrev & return
2413 4F 00330 LD C,A
2414 00340 @@ERROR
2414 3E1A 00001 LD A,26
2416 EF 00002 RST 40
2417 180D 00350 JR SAVESP ;P/u stack & return
00360 ;
00370 ; Internal Error Message Handling
00380 ;
2419 215525 00390 CANT LD HL,CANT$ ;"No mem space..."
241C DD 00400 DB 0DDH
241D 216F25 00410 SPCERR LD HL,SPCERR$ ;"Devspec req..."
2420 00420 @@LOGOT
00003 IFEQ 00H,1
00004 LD HL,
00005 ENDIF
2420 3E0C 00006 LD A,12
2422 EF 00007 RST 40
2423 21FFFF 00430 ERREXIT LD HL,-1 ;Set abort code
00440 ;
00450 ; P/u stack & Clear any pending <BREAK>
00460 ;
2426 310000 00470 SAVESP LD SP,$-$ ;P/u stack
2429 00480 @@CKBRKC ;Clear any Break
2429 3E6A 00008 LD A,106
242B EF 00009 RST 40
242C C9 00490 RET
00500 ;
00510 ; ROUTE1 - Route spec to spec
00520 ;
242D 11E525 00530 ROUTE1 LD DE,FCBSRC ;Fetch source spec
2430 00540 @@FSPEC
2430 3E4E 00010 LD A,78
2432 EF 00011 RST 40
2433 2026 00550 JR NZ,SPCER ;Jump on error
2435 1A 00560 LD A,(DE)

```

```

2436 FE2A      00570      CP      '*'      ;Must be a device
2438 2021      00580      JR      NZ,SPCER      ;Jump if not
243A 118425    00590      LD      DE,PRMTBL$      ;Get parameters
243D           00600      @@PARAM
243D 3E11      00012      LD      A,17
243F EF        00013      RST      40
2440 20CA      00610      JR      NZ,PRMERR      ;Jump on parm error
2442 ED5BE625  00620      LD      DE,(FCBSRC+1)      ;Stuff source name
2446 ED539F25  00630      LD      (RTENAM+3),DE
244A           00640      @@FLAGS      ;Get flag table pointer
244A 3E65      00014      LD      A,101
244C EF        00015      RST      40
                00650      ;
                00660      ;      Test NIL parameter
                00670      ;
244D 010000    00680      NPARAM LD      BC,0      ;P/u NIL parm
2450 78        00690      LD      A,B
2451 B1        00700      OR      C
2452 C22825    00710      JP      NZ,NILDCB      ;Jump if NIL entered
                00720      ;
                00730      ;      Route to device/file - check which
                00740      ;
2455 11A525    00750      LD      DE,FCBDST      ;Fetch destination spec
2458           00760      @@FSPEC
2458 3E4E      00016      LD      A,78
245A EF        00017      RST      40
245B C21D24    00770      SPCER  JP      NZ,SPCERR      ;Jump on error
245E D5        00780      PUSH     DE
245F 118425    00790      LD      DE,PRMTBL$
2462           00800      @@PARAM      ;Need in case REWIND
2462 3E11      00018      LD      A,17
2464 EF        00019      RST      40
2465 D1        00810      POP      DE
2466 20A4      00820      JR      NZ,PRMERR      ;Exit on parm error
2468 1A        00830      LD      A,(DE)
2469 FE2A      00840      CP      '*'      ;Test device/file
246B 2040      00850      JR      NZ,INITFCB      ;Jump on file
                00860      ;
                00870      ;      Destination spec is a device
                00880      ;
246D ED5BA625  00890      LD      DE,(FCBDST+1)      ;P/u device name
2471 2AE625    00900      LD      HL,(FCBSRC+1)      ;Make sure SRC<>DST
2474 ED52      00910      SBC      HL,DE      ; CF is reset
2476 CA1D24    00920      JP      Z,SPCERR
2479           00930      @@GTDCB      ;Find in tables
2479 3E52      00020      LD      A,82
247B EF        00021      RST      40
247C C20E24    00940      JP      NZ,IOERR      ;Jump if not found
247F E5        00950      CKDCBS PUSH     HL      ;Save DCB address of dest
2480 CD0D25    00960      CALL     CKSRC      ;Locate source DCB
2483 C20E24    00970      JP      NZ,IOERR
2486           00980      CKDCB1 EQU      $
2486 F3        00990      DI
2487 C1        01000      POP      BC      ;Rcvr dest route vector
2488 E5        01010      PUSH     HL      ;Save DCB+0
                01020      ;
                01030      ;      Save the old device vector while stuffing new
                01040      ;
2489 2C        01050      INC      L      ;Bump to vector
248A 7E        01060      LD      A,(HL)      ;Save what's there
248B 71        01070      LD      (HL),C      ;Stuff dest route

```

```

248C 4F      01080      LD      C,A          ; into DCB of source
248D 2C      01090      INC      L              ; while saving old
248E 7E      01100      LD      A,(HL)         ; vector for storage
248F 70      01110      LD      (HL),B        ; (could be a FCB)
2490 47      01120      LD      B,A
                01130 ;
                01140 ;      Now set ROUTE bit and rest of DCB block
                01150 ;
2491 E1      01160      POP      HL          ;Rcvr ptr to DCB+0
2492 7E      01170      LD      A,(HL)         ;Init the ROUTE bit
2493 F5      01180      PUSH     AF          ;Save old TYPE byte
2494 E607     01190      AND      7            ;Strip any flag bits
2496 F610     01200      OR       10H         ;
2498 77      01210      LD      (HL),A        ;Show source is routed
2499 7D      01220      LD      A,L
249A C607     01230      ADD      A,7          ;Point to name field
249C 6F      01240      LD      L,A
249D 72      01250      LD      (HL),D        ;And stuff in the name
249E 2D      01260      DEC      L            ; in case this is a
249F 73      01270      LD      (HL),E        ; new DCB block
24A0 F1      01280      POP      AF          ;P/u old TYPE byte &
24A1 CB67     01290      BIT      4,A         ; save old data if
24A3 2006     01300      JR      NZ,CKDCB2    ; not already routed
24A5 2D      01310      DEC      L
24A6 70      01320      LD      (HL),B        ;Stuff old vector
24A7 2D      01330      DEC      L            ; for reclamation
24A8 71      01340      LD      (HL),C
24A9 2D      01350      DEC      L
24AA 77      01360      LD      (HL),A        ;Stuff old TYPE
24AB          01370 CKDCB2 EQU      $
24AB FB      01380      EI
24AC C9      01390      RET                  ;Successful
                01400 ;
                01410 ;      Destination is file - init it & posn to end
                01420 ;
24AD D5      01430 INITFCB PUSH     DE
24AE 119C25   01440      LD      DE,RTENAM    ;See if space already
24B1          01450      @GTMOD              ; allocated for this
24B1 3E53     00022      LD      A,83
24B3 EF      00023      RST      40
24B4 D1      01460      POP      DE
24B5 200C     01470      JR      NZ,NOTRES    ; device name
                01480 ;
                01490 ;      Space in memory, re-use it
                01500 ;
24B7 23      01510      INC      HL          ;Get last byte used
24B8 23      01520      INC      HL          ; into HL
24B9 7E      01530      LD      A,(HL)
24BA 23      01540      INC      HL
24BB 66      01550      LD      H,(HL)
24BC 6F      01560      LD      L,A
24BD AF      01570      XOR      A            ;Set a 0 to show
24BE 32F724   01580      LD      (CKIFRES+1),A ; already resident
24C1 180E     01590      JR      SETBUF
                01600 ;
                01610 ;      Not yet resident, get space
                01620 ;
24C3 FDCB0246 01630 NOTRES BIT      0,(IY+'C'-'A') ;Can we alter HIGH$?
24C7 C21924   01640      JP      NZ,CANT     ;Can't if frozen
24CA 210000    01650      LD      HL,0        ;Get high
24CD 45      01660      LD      B,L

```

```

24CE      01670      @@HIGH$
24CE 3E64      00024      LD      A,100
24D0 EF      00025      RST      40
24D1 229925    01680 SETBUF LD      (RTEDVR+2),HL ;Stuff highest used
24D4 23      01690      INC      HL ;Reserve a page for
24D5 25      01700      DEC      H ; the I/O buffer
24D6 E5      01710      PUSH     HL ;Don't lose it
24D7 0600     01720      LD      B,0 ;LRL = 0
24D9      01730      @@INIT    ;Init the file
24D9 3E3A     00026      LD      A,58
24DB EF      00027      RST      40
24DC 200D     01740      JR      NZ,INITF1 ;What? an error?
24DE 010000    01750 RPARM  LD      BC,0 ;Ck on rewind (no peof)
24E1 04      01760      INC      B ;Keep file at start
24E2 2807     01770      JR      Z,INITF1 ; if REWIND specified
24E4      01780      @@PEOF    ; else posn file
24E4 3E41     00028      LD      A,65
24E6 EF      00029      RST      40
24E7 2802     01790      JR      Z,INITF1 ; to the end
24E9 FE1C     01800      CP      1CH ;At End Of File?
24EB E1      01810 INITF1 POP     HL ;Get back buffer pointer
24EC C20E24    01820      JP      NZ,IOERR ;Any other error, JuMp
24EF 012E00    01830      LD      BC,32+14 ;Back up another 32
24F2 AF      01840      XOR      A ; for the FCB storage
24F3 ED42     01850      SBC      HL,BC ; + 14 for linkage
24F5 E5      01860      PUSH     HL ;Save module start
          01870 ;
          01880 ; Bypass HIGH$ stuff if "ISRES"
          01890 ;
24F6 F6FF     01900 CKIFRES OR      -1 ;"OR 0" if "ISRES"
24F8 2804     01910      JR      Z,ISRES1
24FA 2B      01920      DEC      HL ;Reset HIGH$ (B=0)
24FB      01930      @@HIGH$ ;Stuff new high$
24FB 3E64     00030      LD      A,100
24FD EF      00031      RST      40
24FE D1      01940 ISRES1 POP     DE ;Rcvr module pointer
24FF D5      01950      PUSH     DE
2500 219725    01960      LD      HL,RTEDVR ;Move module to memory
2503 EDB0     01970      LDIR
2505 D1      01980      POP      DE ;Now adjust to true
2506 210E00    01990      LD      HL,14 ; FCB loc'n
2509 19      02000      ADD      HL,DE
250A C37F24    02010      JP      CKDCBS ;Go check dcbs
          02020 ;
          02030 ; Scan device tables for source device
          02040 ;
250D ED5BE625 02050 CKSRC  LD      DE,(FCBSRC+1) ;P/u source device name
2511 D5      02060      PUSH     DE ; & save it for later
2512      02070      @@GTDCB ;Find device in table
2512 3E52     00032      LD      A,82
2514 EF      00033      RST      40
2515 280A     02080      JR      Z,CKSRC1 ;Use it if found
2517 110000    02090      LD      DE,0 ; else find a spare
251A      02100      @@GTDCB ; DCB block
251A 3E52     00034      LD      A,82
251C EF      00035      RST      40
251D 3E21     02110      LD      A,33 ;Init "No device space..."
251F 2005     02120      JR      NZ,CKSRC2 ;Abort if no space
2521 E5      02130 CKSRC1 PUSH     HL
2522 CD3925    02140      CALL    CLSFILS ;Close any existing
2525 E1      02150      POP      HL ; file routes

```

```

2526 D1      02160 CKSRC2 POP      DE              ;Recover source name
2527 C9      02170              RET
              02180 ;
              02190 ;      NIL entered, close up any open file
              02200 ;
2528 CD0D25  02210 NILDCB CALL     CKSRC              ;Check on devqice
252B 7E      02220              LD      A,(HL)        ;Get type byte
252C F608    02230              OR      8
252E 77      02240              LD      (HL),A        ;Show is NIL device
252F 7D      02250              LD      A,L          ;Pt to name field
2530 C606    02260              ADD     A,6
2532 6F      02270              LD      L,A
2533 F3      02280              DI
2534 73      02290              LD      (HL),E        ;Stuff in our name
2535 2C      02300              INC     L            ; in case it's a new
2536 72      02310              LD      (HL),D        ; DCB block
2537 FB      02320              EI
2538 C9      02330              RET              ;Successful
              02340 ;
              02350 ;      Find the last device route & close any open file
              02360 ;
2539 CB66    02370 CLSFILS BIT     4,(HL)              ;Jump if no route
253B 2807    02380              JR      Z,CLSFIL1
253D 23      02390              INC     HL              ;Else p/u link address
253E 7E      02400              LD      A,(HL)        ; and test that one
253F 23      02410              INC     HL              ; for a chain
2540 66      02420              LD      H,(HL)
2541 6F      02430              LD      L,A
2542 18F5    02440              JR      CLSFILS
2544 CB7E    02450 CLSFIL1 BIT     7,(HL)              ;A file?
2546 C8      02460              RET      Z          ;Ret if not
2547 11C525  02470              LD      DE,FCBFIL        ;Pt to fcb area
254A D5      02480              PUSH    DE
254B 012000  02490              LD      BC,32
254E EDB0    02500              LDIR              ;Fill from device vector
2550 D1      02510              POP      DE          ;Recover start
2551         02520              @CLOSE        ;Close the file
2551 3E3C    00036              LD      A,60
2553 EF      00037              RST      40
2554 C9      02530              RET              ;Ret with Z, NZ status
              02540 ;
              02550 ;      Messages
              02560 ;
2555 4E      02570 CANT$  DB      'No memory space available',CR
        6F 20 6D 65 6D 6F 72 79
        20 73 70 61 63 65 20 61
        76 61 69 6C 61 62 6C 65
        0D
256F 44      02580 SPCERR$ DB      'Device spec required',CR
        65 76 69 63 65 20 73 70
        65 63 20 72 65 71 75 69
        72 65 64 0D
              02590 ;
2584 80      02600 PRMTBL$ DB      80H,53H,'NIL',0
        53 4E 49 4C 00
258A 4E24    02610              DW      NPARM+1
258C 56      02620              DB      56H,'REWIND',0
        52 45 57 49 4E 44 00
2594 DF 24   02630              DW      RPARM+1
2596 00      02640              NOP
              02650 ;

```

The Source	LIBRARY Files	ROUTE - LS-DOS 6.2	Page 00006
2597 18FE	02660 RTEDVR JR	\$;No real jump	
2599 0000	02670 DW	\$-\$;Stuff of high	
259B 05	02680 DB	5	
259C 52	02690 RTENAM DB	'RTExx'	
54 45 78 78			
25A1 0000	02700 DW	0,0	
0000			
	02710 ;		
25A5 00	02720 FCBDST DB	0	
001F	02730 DS	31	
25C5 00	02740 FCBF IL DB	0	
001F	02750 DS	31	
25E5 00	02760 FCBSRC DB	0	
001F	02770 DS	31	
	02780 ;		
2400	02790 END	ROUTE	

@@1	0000 @@2	0000 @@3	0000
@@4	0000 @MOD2	0000 @MOD4	FFFF
CANT	2419 CANT\$	2555 CKDCB1	2486
CKDCB2	24AB CKDCBS	247F CKIFRES	24F6
CKSRC	250D CKSRC1	2521 CKSRC2	2526
CLSFIL1	2544 CLSFILS	2539 CR	000D
ERREXIT	2423 EXIT	2407 FCB DST	25A5
FCBFIL	25C5 FCBSRC	25E5 INITF1	24EB
INITFCB	24AD IOERR	240E ISRES1	24FE
NILDCB	2528 NOTRES	24C3 NPARM	244D
PAR ERR	002C PRMERR	240C PRMTBL\$	2584
ROUTE	2400 ROUTE1	242D RPARM	24DE
RTEDVR	2597 RTENAM	259C SAVESP	2426
SETBUF	24D1 SPCER	245B SPCERR	241D
SPCERR\$	256F @@ABORT	83D2 @@ADTSK	8465
@@BANK	897D @@BKSP	865D @@BREAK	8993
@@CHNIO	83BD @@CKBRKC	89E1 @@CKDRV	84B9
@@CKEOF	8672 @@CKTSK	8450 @@CLOSE	8648
@@CLS	89CB @@CMNDI	83FC @@CMNDR	8411
@@CTL	8221 @@DATE	8393 @@DCSTAT	84F8
@@DEBUG	843B @@DECHEX	88FD @@DIRRD	886A
@@DIRWR	887F @@DIV16	88E8 @@DIV8	88D3
@@DODIR	84CE @@DSP	81E5 @@DSPLY	8285
@@ERROR	8426 @@EXIT	83E7 @@FEXT	87D7
@@FLAGS	8967 @@FNAME	87EC @@FSPEC	87C2
@@GATRD	8855 @@GATWR	8894 @@GET	81F9
@@GTDCB	8816 @@GTDCT	8801 @@GTMOD	882B
@@HDFMT	85A0 @@HEX16	893C @@HEX8	8927
@@HEXDEC	8912 @@HIGH\$	8951 @@INIT	861E
@@KBD	825D @@KEY	81D1 @@KEYIN	8271
@@KLTSK	84A4 @@LOAD	8798 @@LOC	8687
@@LOF	869C @@LOGGER	82BC @@LOGOT	82D1
@@MSG	8308 @@MUL16	88BE @@MUL8	88A9
@@OPEN	8633 @@PARAM	837E @@PAUSE	8369
@@PEOF	86B1 @@POSN	86C6 @@PRINT	831D
@@PRT	8235 @@PUT	820D @@RAMDIR	84E3
@@RDSEC	8576 @@RDSSC	8840 @@READ	86DB
@@REMOV	8609 @@RENAM	85F4 @@REW	86F0
@@RMTSK	847A @@RPTSK	848F @@RREAD	8705
@@RSLCT	8561 @@RSTOR	8522 @@RUN	87AD
@@RWRT	871A @@SEEK	854C @@SEEKSC	872F
@@SKIP	8744 @@SLCT	850D @@STEP1	8537
@@TIME	83A8 @@VDCTL	8354 @@VER	8759
@@VRSEC	858B @@WEOF	876E @@WHERE	8249
@@WRITE	8783 @@WRSEC	85B5 @@WRSSC	85CA
@@WRTRK	85DF		

2400 is the transfer address

00000 Total errors

NOTES:

NOTES:

Command: SET, FILTER

Library: SYS6/SYS

ISAM # : 65H, 66H

```

00100 ;LBSET/ASM - Set and Filter commands
0000 00110 TITLE <SET/FILTER - LS-DOS 6.2>
00120 ;
000D 00130 CR EQU 13
0000 00140 *GET SVCMAC:3 ;SVC Macro equivalents
00010 ;SVCMAC/ASM - LS-DOS Version VI
00020 *LIST OFF
00300 *LIST ON
00150 ;
00160 ; ORG this up here to allow driver programs to
00170 ; load at X'2400' without clobbering this program
00180 ;
2C00 00190 ORG 2C00H
00200 ;
00210 ; FILTER entry point
00220 ;
2C00 C3B22C 00230 JP FILTER ;Filter entry point
00240 ;
00250 ; SET entry point
00260 ;
2C03 00270 SET @@FLAGS ;Flag table pointer
2C03 3E65 00001 LD A,101
2C05 EF 00002 RST 40
2C06 FDCB0246 00280 BIT 0,(IY+'C'-'A') ;Can't use if memory
2C0A C2272D 00290 JP NZ,CANT ; is frozen
2C0D 11CD2D 00300 LD DE,DEVFCB ;Get filespec
2C10 00310 @@FSPEC
2C10 3E4E 00003 LD A,78
2C12 EF 00004 RST 40
2C13 C2232D 00320 JP NZ,DEVREQ ;Quit if bad name
2C16 1A 00330 LD A,(DE) ;Ck on devicespec
2C17 FE2A 00340 CP '*'
2C19 C2232D 00350 JP NZ,DEVREQ ;Must have device
2C1C 11ED2D 00360 LD DE,PGMFCB ;Get driver or filter
2C1F 00370 @@FSPEC ; filespec
2C1F 3E4E 00005 LD A,78
2C21 EF 00006 RST 40
2C22 C22B2D 00380 JP NZ,SPCREQ ;Must be entered
2C25 1A 00390 LD A,(DE) ;Target cannot be device
2C26 FE2A 00400 CP '*' ; since this is SET
2C28 CA2B2D 00410 JP Z,SPCREQ
2C2B E5 00420 PUSH HL ;Save INBUF$ pointer
2C2C D5 00430 PUSH DE ; and FCB start
2C2D 21AD2D 00440 LD HL,SAVSPEC ;Save the filter/driver
2C30 EB 00450 EX DE,HL ; filespec to try /DVR
2C31 012000 00460 LD BC,32 ; if /FLT is not found
2C34 EDB0 00470 LDIR
2C36 D1 00480 POP DE ;Recover FCB
2C37 210B2D 00490 LD HL,FLTEXT ;Default extension is FLT
2C3A 00500 @@FEXT ;Use default EXT if none
2C3A 3E4F 00007 LD A,79
2C3C EF 00008 RST 40
2C3D E1 00510 POP HL ;Recover cmdline posn
00520 ;
00530 ; Make sure device is not in system
00540 ;
2C3E E5 00550 PUSH HL ;Save INBUF$ pointer
2C3F ED5BCE2D 00560 LD DE,(DEVFCB+1) ;P/u device name
2C43 00570 @@GTDCB ;Find device DCB address
2C43 3E52 00009 LD A,82
2C45 EF 00010 RST 40

```

```

2C46 2016      00580      JR      NZ,NEWDCB      ;Go if not found
2C48 CB5E      00590      BIT      3,(HL)      ; else check if NIL
2C4A 3E27      00600      LD      A,39      ;Init "Device in use..."
2C4C 2840      00610      JR      Z,ERRPOP      ;Error if not NIL
                00620 ;
                00630 ;      Inhibit SETting any system device
                00640 ;
2C4E E5      00650      PUSH   HL      ;Save DCB pointer
2C4F 21CD2D      00660      LD      HL,DEVFCB      ;Determine if system
2C52 54      00670      LD      D,H      ; device by attempting
2C53 5D      00680      LD      E,L      ; to rename it
2C54      00690      @@RENAM      ;The error code will be
2C54 3E38      00011      LD      A,56
2C56 EF      00012      RST      40
2C57 E1      00700      POP      HL      ; either 19 or 40
2C58 FE28      00710      CP      40      ;Protected system device?
2C5A 2832      00720      JR      Z,ERRPOP
2C5C 180A      00730      JR      GOTDCB      ; else we have it
                00740 ;
                00750 ;      Device not found - Locate spare DCB
                00760 ;
2C5E 110000      00770 NEWDCB      LD      DE,0      ;Find spare device
2C61      00780      @@GTDCB      ; table position
2C61 3E52      00013      LD      A,82
2C63 EF      00014      RST      40
2C64 3E21      00790      LD      A,33      ;"no device space avail
2C66 2026      00800      JR      NZ,ERRPOP      ;Exit on error
                00810 ;
                00820 ;      DCB available - Load the driver/filter
                00830 ;
2C68 E5      00840 GOTDCB      PUSH   HL      ;Save table address
2C69 FDCB12D6      00850      SET      2,(IY+'S'-'A')      ;Allow use with EXEC only
2C6D 11ED2D      00860      LD      DE,PGMFCB      ;Load the target file
2C70      00870      @@LOAD      ;Transfer address in HL
2C70 3E4C      00015      LD      A,76
2C72 EF      00016      RST      40
2C73 281D      00880      JR      Z,LOADOK      ;Go if file found
2C75 E63F      00890      AND      3FH      ;Strip flags
2C77 FE1F      00900      CP      31      ;Program not found?
2C79 2012      00910      JR      NZ,LOADERR      ;Abort on any other
                00920 ;
                00930 ;      No FILTER found - Check on DRIVER
                00940 ;
2C7B 11AD2D      00950      LD      DE,SAVSPEC      ;Original filename
2C7E 210E2D      00960      LD      HL,DVREXT      ;Try with /DVR
2C81      00970      @@FEXT
2C81 3E4F      00017      LD      A,79
2C83 EF      00018      RST      40
2C84 FDCB12D6      00980      SET      2,(IY+'S'-'A')      ;Allow use with EXEC only
2C88      00990      @@LOAD
2C88 3E4C      00019      LD      A,76
2C8A EF      00020      RST      40
2C8B 2805      01000      JR      Z,LOADOK      ;Go if file found
2C8D E1      01010 LOADERR      POP      HL      ;Clean the stack
2C8E E1      01020 ERRPOP      POP      HL
2C8F C3112D      01030      JP      IOERR      ;Abort on load error
                01040 ;
                01050 ;      Move device name into string buffer
                01060 ;
2C92 D1      01070 LOADOK      POP      DE      ;Rcvr table address
2C93 D5      01080      PUSH   DE

```

```

2C94 F3      010900      DI                      ;Don't interrupt me
2C95 3E08    011000      LD      A,8                ;Set up as NIL first
2C97 12      011100      LD      (DE),A
2C98 1C      011200      NOSET INC      E                ;Transfer device name
2C99 1C      011300      INC      E                ; entered in command
2C9A 1C      011400      INC      E                ; to the device table
2C9B 3E08    011500      LD      A,8                ;Show RESET as NIL
2C9D 12      011600      LD      (DE),A
2C9E 1C      011700      INC      E
2C9F 1C      011800      INC      E                ;Point to name field
2CA0 1C      011900      INC      E
2CA1 3ACE2D  012000      LD      A,(DEVFCB+1)    ;Move name to DCB
2CA4 12      012100      LD      (DE),A
2CA5 1C      012200      INC      E
2CA6 3ACF2D  012300      LD      A,(DEVFCB+2)
2CA9 12      012400      LD      (DE),A
2CAA FB      012500      EI                      ;Interrupts back on
2CAB D1      012600      GODOIT POP     DE            ;Recover DCB address
2CAC E3      012700      EX      (SP),HL        ;Stack prog's TRAADR
2CAD FDCB02DE 012800      SET     3,(IY+'C'-'A') ;Set system request
2CB1 C9      012900      RET                     ; & go to it
          013000 ;
          013100 ;      FILTER *dev *dev routine
          013200 ;
2CB2 11CD2D  013300      FILTER LD      DE,DEVFCB        ;Get first spec
2CB5      013400      @@FSPEC
2CB5 3E4E    00021      LD      A,78
2CB7 EF      00022      RST      40
2CB8 C2232D  013500      JP      NZ,DEVREQ    ;Quit on bad name
2CBB 1A      013600      LD      A,(DE)        ;Ck on devicespec
2CBC FE2A    013700      CP      '*'
2CBE C2232D  013800      JP      NZ,DEVREQ    ;Must have device
2CC1 11ED2D  013900      LD      DE,PGMFCB    ;Get filter device spec
2CC4      014000      @@FSPEC
2CC4 3E4E    00023      LD      A,78
2CC6 EF      00024      RST      40
2CC7 C22B2D  014100      JP      NZ,SPCREQ    ;Must be entered
2CCA 1A      014200      LD      A,(DE)        ;Target must be a device
2CCB FE2A    014300      CP      '*'          ; since this is FILTER
2CCD C2232D  014400      JP      NZ,DEVREQ
2CD0 ED5BEE2D 014500      LD      DE,(PGMFCB+1) ;Get filter DCB address
2CD4      014600      @@GTDCB
2CD4 3E52    00025      LD      A,82
2CD6 EF      00026      RST      40
2CD7 2038    014700      JR      NZ,IOERR      ;Quit if not found
2CD9 CB76    014800      BIT     6,(HL)        ;Must be a filter
2CDB 2842    014900      JR      Z,NOTFLT      ;Quit if not
          015000 ;
          015100 ;      FILTER must be inactive to use it
          015200 ;
2CDD 54      015300      LD      D,H            ;Xfer FILTER DCB pointer
2CDE 5D      015400      LD      E,L            ; to DE & locate the
2CDF 2C      015500      INC      L            ; DCB pointer in the
2CE0 7E      015600      LD      A,(HL)        ; the FILTER module
2CE1 2C      015700      INC      L
2CE2 66      015800      LD      H,(HL)
2CE3 6F      015900      LD      L,A
2CE4 010400  016000      LD      BC,4          ;HL now points to the
2CE7 09      016100      ADD     HL,BC          ; entry point. Get its
2CE8 4E      016200      LD      C,(HL)        ; DCB address by peeking
2CE9 0C      016300      INC      C            ; past the name field

```

```

2CEA 09      01640      ADD      HL,BC
2CEB 7E      01650      LD        A,(HL)      ;Get low-order
2CEC 23      01660      INC        HL
2CED 66      01670      LD        H,(HL)      ;Get hi-order
2CEE 6F      01680      LD        L,A
                01690      ;If DCB is NOT pointing
2CEF ED52    01700      SBC        HL,DE      ; to itself, then it
2CF1 2028    01710      JR         NZ,ACTFLT    ; is an active filter
                01720      ;
                01730      ;
                01740      ;
                01750      ;
2CF3 D5      01750      PUSH       DE      ;Save filter DCB
2CF4 ED5BCE2D 01760      LD         DE,(DEVFCB+1) ;Find the device DCB
2CF8         01770      @@GTDCB
2CF8 3E52    00027      LD         A,82
2CFA EF      00028      RST        40
2CFB D1      01780      POP        DE
2CFC 2013    01790      JR         NZ,IOERR      ;Quit if not found
                01800      ;
                01810      ;
                01820      ;
                01830      ;
2CFE 0603    01830      LD         B,3
2D00 4E      01840      LD         C,(HL)
2D01 1A      01850      LD         A,(DE)
2D02 77      01860      LD         (HL),A
2D03 79      01870      LD         A,C
2D04 12      01880      LD         (DE),A
2D05 2C      01890      INC        L
2D06 1C      01900      INC        E
2D07 10F7    01910      DJNZ       SWAP
2D09 182A    01920      JR         EXIT
                01930      ;
2D0B 46      01940      FLTEXT    DB      'FLT'
                4C 54
2D0E 44      01950      DVREXT    DB      'DVR'
                56 52
                01960      ;
2D11 6F      01970      IOERR     LD         L,A      ;Transfer error code
2D12 2600    01980      LD         H,0      ; to HL
2D14 F6C0    01990      OR        0C0H      ;Set abbrev & return
2D16 4F      02000      LD         C,A
2D17         02010      @@ERROR
2D17 3E1A    00029      LD         A,26
2D19 EF      00030      RST        40
2D1A C9      02020      RET
                02030      ;
2D1B 21952D  02040      ACTFLT    LD         HL,ACTFLT$
2D1E DD      02050      DB         0DDH
2D1F 217E2D  02060      NOTFLT    LD         HL,NOTFLT$
2D22 DD      02070      DB         0DDH
2D23 214F2D  02080      DEVREQ    LD         HL,DEVREQ$
2D26 DD      02090      DB         0DDH
2D27 21642D  02100      CANT      LD         HL,CANT$
2D2A DD      02110      DB         0DDH
2D2B 213C2D  02120      SPCREQ    LD         HL,SPCREQ$
2D2E         02130      @@LOGOT
                00031      IFEQ      00H,1
                00032      LD         HL,
                00033      ENDIF
2D2E 3E0C    00034      LD         A,12
2D30 EF      00035      RST        40

```

The Source	LIBRARY Files	SET/FILTER - LS-DOS 6.2	Page 00005
2D31 21FFFF	02140 LD	HL,-1	
2D34 C9	02150 RET		
	02160 ;		
2D35 210000	02170 EXIT LD	HL,0	
2D38	02180 @@CKBRKC		;Clear out break
2D38 3E6A	00036 LD	A,106	
2D3A EF	00037 RST	40	
2D3B C9	02190 RET		
	02200 ;		
2D3C 46	02210 SPCREQ\$ DB	'File spec required',CR	
69 6C 65	20 73 70 65 63		
20 72 65	71 75 69 72 65		
64 0D			
2D4F 44	02220 DEVREQ\$ DB	'Device spec required',CR	
65 76 69	63 65 20 73 70		
65 63 20	72 65 71 75 69		
72 65 64	0D		
2D64 4E	02230 CANT\$ DB	'No memory space available',CR	
6F 20 6D	65 6D 6F 72 79		
20 73 70	61 63 65 20 61		
76 61 69	6C 61 62 6C 65		
0D			
2D7E 44	02240 NOTFLT\$ DB	'Device is not a filter',CR	
65 76 69	63 65 20 69 73		
20 6E 6F	74 20 61 20 66		
69 6C 74	65 72 0D		
2D95 46	02250 ACTFLT\$ DB	'FILTER module is in use',CR	
49 4C 54	45 52 20 6D 6F		
64 75 6C	65 20 69 73 20		
69 6E 20	75 73 65 0D		
	02260 ;		
0020	02270 SAVSPEC DS	32	
0020	02280 DEVFCB DS	32	;Device file control block
0020	02290 PGMFCB DS	32	;Driver/filter FCB
	02300 ;		
2C03	02310 END	SET	

@@1	0000 @@2	0000 @@3	0000
@@4	0000 @MOD2	0000 @MOD4	FFFF
ACTFLT	2D1B ACTFLT\$	2D95 CANT	2D27
CANT\$	2D64 CR	000D DEVFCB	2DCD
DEVREQ	2D23 DEVREQ\$	2D4F DVREXT	2D0E
ERRPOP	2C8E EXIT	2D35 FILTER	2CB2
FLTEXT	2D0B GODOIT	2CAB GOTDCB	2C68
IOERR	2D11 LOADERR	2C8D LOADOK	2C92
NEWDCB	2C5E NOSET	2C98 NOTFLT	2D1F
NOTFLT\$	2D7E PGMFCB	2DED SAVSPEC	2DAD
SET	2C03 SPCREQ	2D2B SPCREQ\$	2D3C
SWAP	2D00 @@ABORT	80A0 @@ADTSK	8133
@@BANK	864B @@BKSP	832B @@BREAK	8661
@@CHNIO	808B @@CKBRKC	86AF @@CKDRV	8187
@@CKEOF	8340 @@CKTSK	811E @@CLOSE	8316
@@CLS	8699 @@CMNDI	80CA @@CMNDR	80DF
@@CTL	7EEF @@DATE	8061 @@DCSTAT	81C6
@@DEBUG	8109 @@DECHEX	85CB @@DIRRD	8538
@@DIRWR	854D @@DIV16	85B6 @@DIV8	85A1
@@DODIR	819C @@DSP	7EB3 @@DSPLY	7F53
@@ERROR	80F4 @@EXIT	80B5 @@FEXT	84A5
@@FLAGS	8635 @@FNAME	84BA @@FSPEC	8490
@@GATRD	8523 @@GATWR	8562 @@GET	7EC7
@@GTDCB	84E4 @@GTDCT	84CF @@GTMOD	84F9
@@HDFMT	826E @@HEX16	860A @@HEX8	85F5
@@HEXDEC	85E0 @@HIGH\$	861F @@INIT	82EC
@@KBD	7F2B @@KEY	7E9F @@KEYIN	7F3F
@@KLTSK	8172 @@LOAD	8466 @@LOC	8355
@@LOF	836A @@LOGGER	7F8A @@LOGOT	7F9F
@@MSG	7FD6 @@MUL16	858C @@MUL8	8577
@@OPEN	8301 @@PARAM	804C @@PAUSE	8037
@@PEOF	837F @@POSN	8394 @@PRINT	7FEB
@@PRT	7F03 @@PUT	7EDB @@RAMDIR	81B1
@@RDSEC	8244 @@RDSSC	850E @@READ	83A9
@@REMOV	82D7 @@RENAM	82C2 @@REW	83BE
@@RMTSK	8148 @@RPTSK	815D @@RREAD	83D3
@@RSLCT	822F @@RSTOR	81F0 @@RUN	847B
@@RWRT	83E8 @@SEEK	821A @@SEEKSC	83FD
@@SKIP	8412 @@SLCT	81DB @@STEPI	8205
@@TIME	8076 @@VDCTL	8022 @@VER	8427
@@VRSEC	8259 @@WEOF	843C @@WHERE	7F17
@@WRITE	8451 @@WRSEC	8283 @@WRSSC	8298
@@WRTRK	82AD		

2C03 is the transfer address

00000 Total errors

NOTES:

Command: SETCOM

Library: SYS8/SYS

ISAM # : B2H

```

00100 ;LBSETCOM/ASM - Set RS232 Parameters
0000 00110 TITLE <SETCOM - LS-DOS 6.2>
00120 ;
00130 ; Data area offsets
00140 ;
0004 00150 OFFSET EQU 4
0004 00160 MSMASK EQU OFFSET+0
0005 00170 UCIMAGE EQU OFFSET+1
0006 00180 BAUDRT EQU OFFSET+2
0007 00190 BRK EQU OFFSET+3
00200 ;
00210 ; Default settings
00220 ;
0000 00230 DEFMS EQU 000H
00A5 00240 DEFUC EQU 0A5H
0055 00250 DEFBA EQU 055H
0003 00260 DEFBR EQU 003H
00270 ;
00280 ; ASCII/init equivalences
00290 ;
0002 00300 _INIT EQU 02H ;Ctl value to init driver
0003 00310 _ETX EQU 03H
000D 00320 _CR EQU 0DH
000A 00330 _LF EQU 0AH
0027 00340 _APOS EQU 27H
002C 00350 PAR_ERR EQU 44 ;Parameter Error
00360 ;
00370 ; @PARAM evaluation types
00380 ;
0040 00390 FLAG EQU 01000000B
0010 00400 ABB EQU 00010000B
0080 00410 NUM EQU 10000000B
0020 00420 STR EQU 00100000B
00430 ;
0000 00440 *GET SVCMAC:3 ;SVC Macro equivalents
00010 ;SVCMAC/ASM - LS-DOS Version VI
00020 *LIST OFF
00390 *LIST ON
00450 ;
2400 00460 ORG 2400H
00470 ;
2400 00480 BEGIN EQU $
2400 ED730B24 00490 LD (SAVEP+1),SP ;Save entry stack
2404 CD3A24 00500 CALL PGRM ;Perform operations
00510 ;
00520 ; Set up exit condition
00530 ;
2407 210000 00540 $EXIT LD HL,0 ;No errors
240A 310000 00550 SAVESP LD SP,$-$ ;P/u original SP
240D 00560 @@CKBRKC ;Clear <BREAK>
240D 3E6A 00001 LD A,106
240F EF 00002 RST 40
2410 C9 00570 RET
00580 ;
00590 ; Abort program
00600 ;
2411 21FFFF 00610 $ABORT LD HL,-1 ;Init error return
2414 18F4 00620 JR SAVESP ;Exit program
00630 ;
00640 ; Display char in C
00650 ;

```

```

2416      00660 $DSP      @@DSP      ;Display
2416 3E02      00003      LD      A,2
2418 EF      00004      RST      40
2419 C8      00670      RET      Z      ;Return if NO error
241A 1804      00680      JR      IOERR      ; else error
          00690 ;
          00700 ;      Display text @ (HL)
          00710 ;
241C      00720 $DSPLY  @@DSPLY      ;Display
          00005      IFEQ      00H,1
          00006      LD      HL,
          00007      ENDIF
241C 3E0A      00008      LD      A,10
241E EF      00009      RST      40
241F C8      00730      RET      Z      ;Return if NO error
          00740 ;
          00750 ;      I/O error, display what
          00760 ;
2420 6F      00770 IOERR  LD      L,A      ;Save error code
2421 F6C0      00780      OR      0C0H      ;Set return+short message
2423 4F      00790      LD      C,A      ;Pass error code
2424      00800      @@ERROR      ;Display error
2424 3E1A      00010      LD      A,26
2426 EF      00011      RST      40
          00810 ;
2427 2600      00820 ERROR$ LD      H,0      ;Set exit code
2429 18DF      00830      JR      SAVESP      ;Terminate program
          00840 ;
242B 21E126      00850 BADMOD LD      HL,BADDCB      ;'COM/DVR not installed'
242E      00860      @@LOGOT      ;Log error
          00012      IFEQ      00H,1
          00013      LD      HL,
          00014      ENDIF
242E 3E0C      00015      LD      A,12
2430 EF      00016      RST      40
2431 2E08      00870      LD      L,8      ;'device not available'
2433 18F2      00880      JR      ERROR$      ;Return error in HL
          00890 ;
          00900 ;      Parameter error
          00910 ;
2435 3E2C      00920 PRMERR LD      A,PAR_ERR      ;"Parameter Error"
2437 C32024      00930      JP      IOERR      ;Abort
          00940 ;
          00950 ;      PGRM - Set up RS-232 Parameters
          00960 ;
243A      00970 PGRM      EQU      $
          00980 ;
          00990 ;      IF      @MOD4
243A E5      01000      PUSH      HL      ;Save command line pntr
243B 11DD26      01010      LD      DE,MDNAME      ;$CL name header
243E      01020      @@GTMOD      ;Find module header
243E 3E53      00017      LD      A,83
2440 EF      00018      RST      40
2441 20E8      01030      JR      NZ,BADMOD      ;Exit if not found
2443 D5      01040      PUSH      DE      ;Pass pointer to IX
2444 DDE1      01050      POP      IX      ;IX=>next byte after
2446 E1      01060      POP      HL      ;Rcvr cmdline ptr
          01070      ENDIF
          01080 ;
          01090 ;      Check if any parameters entered
          01100 ;

```

```

2447 2B      01110      DEC      HL      ;Setup for immed INC
2448 23      01120 SPLP      INC      HL      ;Bump to next char
2449 7E      01130      LD      A,(HL)      ;Fetch input char
244A FE20    01140      CP      ' '          ;Space?
244C 28FA    01150      JR      Z,SPLP      ;Ignore if space
244E FE28    01160      CP      '('          ;Param marker?
2450 C2E229  01170      JP      NZ,SHOW      ;No, show settings
                01180 ;
2453 110728  01190      LD      DE,PRMTBL$    ;Parameter table
2456         01200      @@PARAM              ;Evaluate user input
2456 3E11    00019      LD      A,17
2458 EF      00020      RST      40
2459 C23524  01210      JP      NZ,PRMERR      ;Go if parameter error
                01220 ;
                01230 ;
                01240      LD      A,(PORTP)    ;Get port param
                01250      OR      A            ;Anything input?
                01260      LD      DE,MDNAME1    ;$CL
                01270      JR      Z,PORTF      ;Go default
                01280      LD      BC,(PORTD)    ;Get port value
                01290      INC      B            ;Test msb
                01300      DEC      B
                01310      JR      NZ,PRMERR      ;Param error if >256
                01320      LD      A,C          ;Get lsb
                01330      OR      A            ;0?
                01340      JR      Z,PRMERR      ;Go if yes
                01350      DEC      A            ;1?
                01360      JR      Z,PORTF      ;Yes, $CL
                01370      DEC      A            ;2?
                01380      JR      NZ,PRMERR      ;Param error if not
                01390      LD      DE,MDNAME2    ;$CM
245C         01400 PORTF  @@GTMOD              ;Locate it
                00021      LD      A,83
                00022      RST      40
                01410      JR      NZ,BADMOD      ;Go if not found
                01420      PUSH     DE            ;Else pass to IX
                01430      POP      IX           ;IX => block
                01440      ENDIF
                01450 ;
                01460 ;
                01470 ;
                01480      LD      DE,(DF PARM)    ;Check D parm
245C ED5B6627 01490      LD      A,D
2460 7A      01490      LD      A,D
2461 B3      01500      OR      E
2462 C4CC26  01510      CALL    NZ,DEFALT      ;Go if Default parm used
                01520 ;
2465 ED5B4C27 01530      LD      DE,(QPARM)      ;Query parm used?
2469 7A      01540      LD      A,D            ;Check input
246A B3      01550      OR      E
246B CA2025  01560      JP      Z,CKPARM      ;Check input if not query
                01570 ;
                01580 ;
                01590 ;
                01600 ;
                01610 ;
                01620 ASKBAUD LD      HL,DMSG2    ;BAUD param display
2471 111A28  01630      LD      DE,BTYP      ;BAUD param
2474 CD5026  01640      CALL    GETIT        ;Get user input
2477 300B    01650      JR      NC,ASKWORD      ;Continue on ENTER
2479 CDC229  01660      CALL    CKBAUD        ;Test response for valid
247C 2806    01670      JR      Z,ASKWORD      ;Move on if valid entry

```

```

247E AF      01680 BADBAUD XOR    A      ; else clear
247F 321F28  01690          LD      (BRESP),A ; for re-try
2482 18EA    01700          JR      ASKBAUD   ;Bad entry - retry
          01710 ;
          01720 ;      Fetch WORD
          01730 ;
2484 216625  01740 ASKWORD LD      HL,DMSG3   ;WORD display
2487 112228  01750          LD      DE,WTYP    ;Param type
248A CD5026  01760          CALL   GETIT       ;Get user input
248D 3011    01770          JR      NC,ASKSTOP ;Nil input, leave it
248F 3A5027  01780          LD      A,(WORDP)   ;Fetch input
2492 FE05    01790          CP      5           ;<5?
2494 3804    01800          JR      C,BADWORD   ;Bad entry
2496 FE09    01810          CP      8+1         ;>8?
2498 3806    01820          JR      C,ASKSTOP   ;Go if OK
249A AF      01830 BADWORD XOR    A           ;Clear param
249B 322728  01840          LD      (WRESP),A   ;Load flag
249E 18E4    01850          JR      ASKWORD     ;Re-try
          01860 ;
          01870 ;      Fetch STOP
          01880 ;
24A0 217F25  01890 ASKSTOP LD      HL,DMSG4   ;STOP
24A3 112A28  01900          LD      DE,STYP    ;=>type byte
24A6 CD5026  01910          CALL   GETIT       ;Get user input
24A9 3014    01920          JR      NC,ASKPAR   ;Nil input, leave it
24AB 3A5227  01930          LD      A,(STOPP)   ;Chk range
24AE B7      01940          OR      A           ;0?
24AF 2804    01950          JR      Z,BADSTOP   ;Invalid if yes
24B1 FE03    01960          CP      2+1         ;>2?
24B3 380A    01970          JR      C,ASKPAR   ;Go if OK
24B5 AF      01980 BADSTOP XOR    A           ;Load zero
24B6 322F28  01990          LD      (SRESP),A   ;Clear input param
24B9 18E5    02000          JR      ASKSTOP     ;Ask again
          02010 ;
          02020 ;      Fetch PARITY
          02030 ;
24BB AF      02040 BADPAR  XOR    A           ;Load zero
24BC 323928  02050          LD      (PRESP),A   ;Clear response byte
24BF 219125  02060 ASKPAR  LD      HL,DMSG5   ;PARITY display driver
24C2 113228  02070          LD      DE,PTYP    ;Param type
24C5 CD5026  02080          CALL   GETIT       ;Get user input
24C8 3017    02090          JR      NC,ASKBRK   ;Nil input, leave
          02100 ;
          02110 ;      Check if user entered '0' or 'E' or ON/OFF
          02120 ;
24CA 3A3928  02130          LD      A,(PRESP)   ;Get response byte
24CD ED5B5427 02140          LD      DE,(PARITYP) ;Get param pointer
24D1 CB77    02150          BIT      6,A           ;Switch?
24D3 200C    02160          JR      NZ,ASKBRK   ;Yes, continue
          02170 ;
          02180 ;      Check for ODD/EVEN input
          02190 ;
24D5 1A      02200          LD      A,(DE)       ;Get input
24D6 CDB929  02210          CALL   UCASE       ;Make upper case
24D9 FE45    02220          CP      'E'           ;Even?
24DB 2804    02230          JR      Z,ASKBRK   ;Continue if yes
24DD FE4F    02240          CP      'O'           ;Odd?
24DF 20DA    02250          JR      NZ,BADPAR   ;Invalid, re-prompt
          02260 ;
          02270 ;      Fetch BREAK
          02280 ;

```

The Source	LIBRARY Files	SETCOM - LS-DOS 6.2	Page 00005
24E1 21B025	02290 ASKBRK	LD HL,DMSG6	;Display driver
24E4 110828	02300	LD DE,BRTYP	;Param type
24E7 CD5026	02310	CALL GETIT	;Get user input
	02320 ;		
	02330 ;	Fetch DTR	
	02340 ;		
24EA 21CB25	02350 ASKDTR	LD HL,DMSG8	;Display driver
24ED 114B28	02360	LD DE,DTYP	;Param type
24F0 CD5026	02370	CALL GETIT	;Get user input
	02380 ;		
	02390 ;	Fetch RTS	
	02400 ;		
24F3 21D725	02410 ASKRTS	LD HL,DMSG9	;Display driver
24F6 115228	02420	LD DE,RTYP	;Param type
24F9 CD5026	02430	CALL GETIT	;Get user input
	02440 ;		
	02450 ;	Fetch RI	
	02460 ;		
24FC 21F225	02470 ASKRI	LD HL,DMSG11	;Display driver
24FF 115928	02480	LD DE,RITYP	;Param type
2502 CD5026	02490	CALL GETIT	;Get user input
	02500 ;		
	02510 ;	Fetch DSR	
	02520 ;		
2505 21F925	02530 ASKDSR	LD HL,DMSG12	;Display driver
2508 116528	02540	LD DE,DSTYP	;Param type
250B CD5026	02550	CALL GETIT	;Get user input
	02560 ;		
	02570 ;	Fetch CD	
	02580 ;		
250E 210026	02590 ASKCD	LD HL,DMSG13	;Display driver
2511 115F28	02600	LD DE,CDTYP	;Param type
2514 CD5026	02610	CALL GETIT	;Get user input
	02620 ;		
	02630 ;	Fetch CTS	
	02640 ;		
2517 210726	02650 ASKCTS	LD HL,DMSG14	;Display driver
251A 116C28	02660	LD DE,CTTYP	;Param type
251D CD5026	02670	CALL GETIT	;Get user input
	02680 ;		
	02690 ;	Check params and issue INIT to device	
	02700 ;		
2520 CD7F28	02710 CKPARAM	CALL SETPARAM	;Check entered params
2523 DD5E00	02720	LD E,(IX+0)	;Pickup DCB address lsb
2526 DD5601	02730	LD D,(IX+1)	;Pickup DCB address msb
2529 0E02	02740	LD C,_INIT	;Init code
252B	02750	@@CTL	;Setup channel for values
252B 3E05	00023	LD A,5	
252D EF	00024	RST 40	
252E C9	02760	RET	;Completed
	02770 ;		
	02780 ;	Display drivers for text	
	02790 ;		
252F 216C27	02800 DMSG1	LD HL,MSG1	;RS232 params:
2532 CD1C24	02810	CALL \$DSPLY	;Display and return
2535 360A	02820	LD (HL),_LF	;Set for next time
2537 C9	02830	RET	
	02840 ;		
2538 218027	02850 DMSG2	LD HL,MSG2	;BAUD=
253B CD1C24	02860	CALL \$DSPLY	;Display
253E DD7E06	02870	LD A,(IX+BAUDRT)	;Get baud rate

```

2541 E60F      02880      AND      0FH      ;Low 4 bits only
2543 87        02890      ADD      A,A      ;*2
2544 21FA26    02900      LD       HL,BAUDTBL ;Baud lookup table
2547 85        02910      ADD      A,L      ;Add to lsb table
2548 6F        02920      LD       L,A      ;Update lsb table
2549 3001      02930      JR       NC,$+3    ;Go if no page cross
254B 24        02940      INC      H      ; else bump page
254C 7E        02950      LD       A,(HL)    ;Get LSB baud
254D 23        02960      INC      HL      ;Bump table pointer
254E 66        02970      LD       H,(HL)    ;Get MSB baud
254F 6F        02980      LD       L,A      ;HL = baud rate
2550 11F727    02990      LD       DE,MSG20   ;Text to load
2553          03000      @HEXDEC    ;Convert to decimal
2553 3E61      03025      LD       A,97
2555 EF        03026      RST      40
2556 3E03      03010      LD       A,ETX      ;End text char
2558 12        03020      LD       (DE),A     ;Terminate text
                03030 ;
                03040 ;      Strip leading zeroes from display
                03050 ;
2559 21F727    03060      LD       HL,MSG20   ;Text loaded
255C 3E20      03070      LD       A,' '      ;Test char
255E BE        03080      CP       (HL)      ;Leading zero?
255F 23        03090      INC      HL      ;Bump pointer
2560 28FC      03100      JR       Z,STRIP1  ;Go till non-zero found
2562 2B        03110      DEC      HL      ;Adjust pointer
2563 C31C24    03120      JP       $DSPLY    ;Display remainder
                03130 ;
2566 218627    03140      DMSG3     LD       HL,MSG3      ;WORD=
2569 CD1C24    03150      CALL    $DSPLY    ;Display header
256C DD7E05    03160      LD       A,(IX+UCIMAGE) ;Get data
256F 07        03170      RLCA      ;Align to low bits
2570 07        03180      RLCA
2571 07        03190      RLCA
2572 E603      03200      AND      3      ;2 bits only
2574 EA7925    03210      JP       PE,$+5    ;Go if set
2577 EE03      03220      XOR      3      ;Else reverse 6/7
2579 C635      03230      ADD      A,5+'0'   ;Correct + ASCII
257B 4F        03240      LD       C,A      ;Pass character
257C C31624    03250      JP       $DSP      ;Display and return
                03260 ;
257F 218C27    03270      DMSG4     LD       HL,MSG4      ;STOP=
2582 CD1C24    03280      CALL    $DSPLY    ;Display header
2585 0E31      03290      LD       C,'1'     ;Set one bit
2587 DDCB0566  03300      BIT      4,(IX+UCIMAGE) ;Get data
258B 2801      03310      JR       Z,$+3    ;Go if one stop
258D 0C        03320      INC      C      ;Else bump to '2'
258E C31624    03330      JP       $DSP      ;Display and return
                03340 ;
2591 219227    03350      DMSG5     LD       HL,MSG5      ;PARITY=
2594 CD1C24    03360      CALL    $DSPLY    ;Display
2597 21E327    03370      LD       HL,MSG16   ;OFF
259A DDCB055E  03380      BIT      3,(IX+UCIMAGE) ;Is off?
259E C21C24    03390      JP       NZ,$DSPLY ;Yes, go
25A1 21E727    03400      LD       HL,MSG17   ;ODD
25A4 DDCB057E  03410      BIT      7,(IX+UCIMAGE) ;Is odd?
25A8 2803      03420      JR       Z,$+5    ;Go if yes
25AA 21EB27    03430      LD       HL,MSG18   ;EVEN
25AD C31C24    03440      JP       $DSPLY    ;Display and return
                03450 ;
25B0 219A27    03460      DMSG6     LD       HL,MSG6      ;BREAK=

```



```

25B3 CD1C24 03470 CALL $DSPLY ;Display header
25B6 DD4E07 03480 LD C,(IX+BRK) ;Get break char
25B9 21FF27 03490 LD HL,MSG21A ;Text to load
25BC 03500 @@HEX8 ;Convert to ASCII
25BC 3E62 00027 LD A,98
25BE EF 00028 RST 40
25BF 21FD27 03510 LD HL,MSG21 ;Text start
25C2 C31C24 03520 JP $DSPLY ;Display and return
03530 ;
25C5 21A127 03540 DMSG7 LD HL,MSG7 ;Input:
25C8 C31C24 03550 JP $DSPLY ;Display and return
03560 ;
25CB 21B327 03570 DMSG8 LD HL,MSG8 ;DTR=
25CE CD1C24 03580 CALL $DSPLY ;Display
25D1 DDCB054E 03590 BIT 1,(IX+UCIMAGE) ;Is on?
25D5 180A 03600 JR DMSG89 ;Display
03610 ;
25D7 21B827 03620 DMSG9 LD HL,MSG9 ;RTS=
25DA CD1C24 03630 CALL $DSPLY ;Display header
25DD DDCB0546 03640 BIT 0,(IX+UCIMAGE) ;Is on?
03650 ;
25E1 21E027 03660 DMSG89 LD HL,MSG15 ;ON?
25E4 2803 03670 JR Z,$+5 ;Go if yes
25E6 21E327 03680 LD HL,MSG16 ;OFF
25E9 C31C24 03690 JP $DSPLY ;Display and return
03700 ;
25EC 21BD27 03710 DMSG10 LD HL,MSG10 ;Output:
25EF C31C24 03720 JP $DSPLY ;Display and return
03730 ;
25F2 21CE27 03740 DMSG11 LD HL,MSG11 ;RI=
25F5 3E11 03750 LD A,00010001B ;Enable bits
25F7 1813 03760 JR DMSG0 ;Go common
03770 ;
25F9 21D227 03780 DMSG12 LD HL,MSG12 ;DSR=
25FC 3E44 03790 LD A,01000100B ;Enable bits
25FE 180C 03800 JR DMSG0 ;Go common
03810 ;
2600 21D727 03820 DMSG13 LD HL,MSG13 ;CD=
2603 3E22 03830 LD A,00100010B ;Enable bits
2605 1805 03840 JR DMSG0 ;Go common
03850 ;
2607 21DB27 03860 DMSG14 LD HL,MSG14 ;CTS=
260A 3E88 03870 LD A,10001000B ;Enable bits
03880 ;
260C F5 03890 DMSG0 PUSH AF ;Save bits
260D CD1C24 03900 CALL $DSPLY ;Display prefix
2610 F1 03910 POP AF ;Restore
2611 DDA604 03920 AND (IX+MSMASK) ;And with mask
2614 21F027 03930 LD HL,MSG19 ;IGNORE
2617 2809 03940 JR Z,DMSG0X ;Go if yes
2619 21E027 03950 LD HL,MSG15 ;ON
261C E22226 03960 JP PO,DMSG0X ;Go if yes
261F 21E327 03970 LD HL,MSG16 ;OFF
2622 C31C24 03980 DMSG0X JP $DSPLY ;Display and return
03990 ;
2625 210328 04000 DMSG22 LD HL,MSG22 ;CR
2628 C31C24 04010 JP $DSPLY ;Display and return
04020 ;
262B 210428 04030 DMSG23 LD HL,MSG23 ;', '
262E C31C24 04040 JP $DSPLY ;Display and return
04050 ;

```

```

04060 ;      Display entire parameter set
04070 ;
2631 FDE5 04080 DSPALL PUSH IY ;Save
2633 FD211A27 04090 LD IY,DSPTBL ;Display table
2637 FD6E00 04100 DSPSET LD L,(IY+0) ;Get lsb vector
263A FD6601 04110 LD H,(IY+1) ;Get msb vector
263D 7C 04120 LD A,H ;Check for term
263E B5 04130 OR L ;HL = 0000?
263F 280C 04140 JR Z,DSPDONE ;Yes, go
2641 E5 04150 PUSH HL ;Save address
2642 214726 04160 LD HL,DSPRET ;Return vector
2645 E3 04170 EX (SP),HL ;Leave return, get vector
2646 E9 04180 JP (HL) ;Display
2647 FD23 04190 DSPRET INC IY ;Bump table
2649 FD23 04200 INC IY ;2 byte entries
264B 18EA 04210 JR DSPSET ;Continue
264D FDE1 04220 DSPDONE POP IY ;Restore
264F C9 04230 RET ;Display complete
04240 ;
04250 ;      QUERYing for parameter
04260 ;
2650 E5 04270 GETIT PUSH HL ;Save display driver
2651 D5 04280 PUSH DE ;Save type byte
2652 EB 04290 EX DE,HL ;Type byte to HL
2653 CD6326 04300 CALL CKRSP ;Check if response entered
2656 D1 04310 POP DE ;Restore type byte
2657 E1 04320 POP HL ;Restore prompt
2658 C0 04330 RET NZ ;Already have this one
04340 ;
04350 ;      Setup for prompt display
04360 ;
2659 E5 04370 PUSH HL ;Save for repeat prompt
265A D5 04380 PUSH DE
265B CD7426 04390 CALL GETRSP ;Get user response
265E D1 04400 POP DE ;Restore data
265F E1 04410 POP HL
2660 20EE 04420 JR NZ,GETIT ;Invalid, ask again
2662 C9 04430 RET ; else param loaded
04440 ;
04450 ;      Check if correct response entered
04460 ;
2663 22A026 04470 CKRSP LD (TTY),HL ;Save type position
04480 ;
2666 3E0F 04490 MVUP LD A,0FH ;Low 4 bits for length
2668 4E 04500 LD C,(HL) ;Get type byte
2669 A1 04510 AND C ;Fetch length
266A 23 04520 UP INC HL ;Bump pointer
266B 3D 04530 DEC A ;Less length
266C 20FC 04540 JR NZ,UP ;Go for length
266E 23 04550 INC HL ;Bump to next
266F 7E 04560 LD A,(HL) ;Get response byte
2670 E6E0 04570 AND NUM!FLAG!STR ;Response type bits
2672 A1 04580 AND C ;Compare to entered
2673 C9 04590 RET ;Return with status
04600 ;
04610 ;      Query on, param not entered, prompt for it
04620 ;
2674 E5 04630 GETRSP PUSH HL ;Save driver address
2675 217A26 04640 LD HL,GETRSP ;Return vector
2678 E3 04650 EX (SP),HL ;Leave get display driver
2679 E9 04660 JP (HL) ;Display prompt

```

```

04670 ;
267A 0604 04680 GETRSPR LD B,4 ;Command get cursor
267C 04690 @VDCTL ;Fetch cursor
267C 3E0F 00029 LD A,15
267E EF 00030 RST 40
267F 2E14 04700 LD L,20 ;Column to position to
2681 0603 04710 LD B,3 ;Command put cursor
2683 04720 @VDCTL ;Setup new cursor
2683 3E0F 00031 LD A,15
2685 EF 00032 RST 40
2686 21F726 04730 LD HL,PROMPT ;Prompt text
2689 CD1C24 04740 CALL $DSPY ;Display prompt
268C AF 04750 XOR A ;Load zero
268D 32E829 04760 LD (FRESP),A ;Clear response byte
04770 ;
2690 21EF29 04780 LD HL,INBUF ;Key input buffer
2693 01000A 04790 LD BC,10<8+0 ;Max input length
2696 04800 @KEYIN ;Get user input
2696 3E09 00033 LD A,9
2698 EF 00034 RST 40
2699 DA1124 04810 JP C,$ABORT ;Terminate on BREAK
04820 ;
04830 ; Evaluate user input
04840 ;
269C 04 04850 INC B ;Check if nil input
269D 05 04860 DEC B ;B=0?
269E C8 04870 RET Z ;No input, go
04880 ;
269F 210000 04890 LD HL,0 ;Get param pointer
26A0 04900 TTYP EQU $-2
26A2 7E 04910 LD A,(HL) ;Get response type
26A3 E6F0 04920 AND 0F0H ; attribute only
26A5 F601 04930 OR 1 ;Single byte char
26A7 32E629 04940 LD (FTYP),A ;Pass to new block
04950 ;
26AA 21EC29 04960 LD HL,FSTR ;Set command pointer
26AD 11E529 04970 LD DE,PTBL2 ;Mini-param table
26B0 04980 @PARAM ;Evaluate input
26B0 3E11 00035 LD A,17
26B2 EF 00036 RST 40
26B3 C0 04990 RET NZ ;Invalid, go
05000 ;
26B4 2AA026 05010 LD HL,(TTYP) ;Get response pointer
26B7 CD6626 05020 CALL MVUP ;Move to response byte
26BA 3AE829 05030 LD A,(FRESP) ;Get input response
26BD 77 05040 LD (HL),A ; to param block
26BE 23 05050 INC HL ;Bump to word pointer
26BF CDB429 05060 CALL GETHL ;Get pointer
26C2 ED4B6A27 05070 LD BC,(FPARM) ;Get temp param
26C6 71 05080 LD (HL),C ;Load into param block
26C7 23 05090 INC HL ;Bump pointer
26C8 70 05100 LD (HL),B ;New param loaded
26C9 AF 05110 XOR A ;Set NO error
26CA 37 05120 SCF ;Carry = input
26CB C9 05130 RET ;Done
05140 ;
05150 ; Init default settings
05160 ;
26CC DD360400 05170 DEFALT LD (IX+MSMASK),DEFMS
26D0 DD3605A5 05180 LD (IX+UCIMAGE),DEFUC
26D4 DD360655 05190 LD (IX+BAUDRT),DEFBA

```

```

26D8 DD360703 05200 LD (IX+BRK),DEFBR
26DC C9 05210 RET
05220 ;
05230 ; Text area
05240 ;
05250 IF @MOD4
26DD 24 05260 MDNAME DB '$CL'
43 4C
26E0 03 05270 DB _ETX
05280 ENDIF
05290 ;
05300 IF @MOD2
05310 MDNAME1 DB '$CL'
05320 DB _ETX
05330 MDNAME2 DB '$CM'
05340 DB _ETX
05350 ENDIF
05360 ;
26E1 43 05370 BDDCB DB 'COM/DVR not installed'
4F 4D 2F 44 56 52 20 6E
6F 74 20 69 6E 73 74 61
6C 6C 65 64
26F6 0D 05380 DB _CR
05390 ;
26F7 3F 05400 PROMPT DB '?'
20
26F9 03 05410 DB _ETX
05420 ;
05430 ; Valid baud rate lookup table
05440 ;
26FA 3200 05450 BAUDTBL DW 50,75,110,135,150,300,600,1200,1800
4B00 6E00 8700 9600 2C01 5802 B004 0807
270C D007 05460 DW 2000,2400,3600,4800,7200,9600,19200
6009 100E C012 201C 8025 004B
05470 ;
05480 ; Display driver lookup table
05490 ;
271A 2F25 05500 DSPTBL DW DMSG1 ;'RS232 params: '
271C 3825 05510 DW DMSG2 ;'BAUD='
271E 2B26 05520 DW DMSG23 ;','
2720 6625 05530 DW DMSG3 ;'WORD='
2722 2B26 05540 DW DMSG23 ;','
2724 7F25 05550 DW DMSG4 ;'STOP='
2726 2B26 05560 DW DMSG23 ;','
2728 9125 05570 DW DMSG5 ;'PARITY='
272A 2B26 05580 DW DMSG23 ;','
272C B025 05590 DW DMSG6 ;'BREAK='
05600 ;
272E C525 05610 DW DMSG7 ;'output: '
2730 CB25 05620 DW DMSG8 ;'DTR='
2732 2B26 05630 DW DMSG23 ;','
2734 D725 05640 DW DMSG9 ;'RTS='
05650 ;
2736 EC25 05660 DW DMSG10 ;'input: '
2738 F225 05670 DW DMSG11 ;'RI='
273A 2B26 05680 DW DMSG23 ;','
273C F925 05690 DW DMSG12 ;'DSR='
273E 2B26 05700 DW DMSG23 ;','
2740 0026 05710 DW DMSG13 ;'CD='
2742 2B26 05720 DW DMSG23 ;','
2744 0726 05730 DW DMSG14 ;'CTS='

```

```

2746 2526      05740      DW      DMSG22      ;_cr
2748 00000     05750      DW      0           ;Terminator
                05760 ;
                05770 ;      Parameter pointer table
                05780 ;
274A 00000     05790 BREAKP DW      0
274C 00000     05800 QPARM  DW      0
274E 00000     05810 BAUDP  DW      0
2750 00000     05820 WORDP  DW      0
2752 00000     05830 STOPP  DW      0
2754 00000     05840 PARITYP DW      0
2756 00000     05850 EVENP  DW      0
2758 00000     05860 ODDP   DW      0
275A 00000     05870 DTRP   DW      0
275C 00000     05880 RTSP   DW      0
275E 00000     05890 RIP    DW      0
2760 00000     05900 CDP    DW      0
2762 00000     05910 DSRP   DW      0
2764 00000     05920 CTSP   DW      0
2766 00000     05930 DFARM   DW      0
2768 00000     05940 PORTD  DW      0
276A 00000     05950 FPARM   DW      0
                05960 ;
276C 00      05970 MSG1    DB      0           ;Will be LF after 1st DSP
276D 52      05980      DB      'RS232'
                53 32 33 32
                05990      IF      @MOD2
                06000 MSG1A  DB      'A'
                06010      ENDIF
2772 20      06020      DB      ' parameters: '
                70 61 72 61 6D 65 74 65
                72 73 3A 20
277F 03      06030      DB      _ETX
                06040 ;
2780 42      06050 MSG2    DB      'Baud='
                61 75 64 3D
2785 03      06060      DB      _ETX
                06070 ;
2786 57      06080 MSG3    DB      'Word='
                6F 72 64 3D
278B 03      06090      DB      _ETX
                06100 ;
278C 53      06110 MSG4    DB      'Stop='
                74 6F 70 3D
2791 03      06120      DB      _ETX
                06130 ;
2792 50      06140 MSG5    DB      'Parity='
                61 72 69 74 79 3D
2799 03      06150      DB      _ETX
                06160 ;
279A 42      06170 MSG6    DB      'Break='
                72 65 61 6B 3D
27A0 03      06180      DB      _ETX
                06190 ;
27A1 0A      06200 MSG7    DB      _LF
27A2 4F      06210      DB      'Output control: '
                75 74 70 75 74 20 63 6F
                6E 74 72 6F 6C 3A 20
27B2 03      06220      DB      _ETX
                06230 ;
27B3 44      06240 MSG8    DB      'DTR='

```

```

54 52 3D
27B7 03      06250      DB      _ETX
                06260 ;
27B8 52      06270 MSG9      DB      'RTS='
54 53 3D
27BC 03      06280      DB      _ETX
                06290 ;
27BD 0A      06300 MSG10     DB      LF
27BE 49      06310      DB      'Input control: '
        6E 70 75 74 20 63 6F 6E
        74 72 6F 6C 3A 20
27CD 03      06320      DB      _ETX
                06330 ;
27CE 52      06340 MSG11     DB      'RI='
        49 3D
27D1 03      06350      DB      _ETX
                06360 ;
27D2 44      06370 MSG12     DB      'DSR='
        53 52 3D
27D6 03      06380      DB      _ETX
                06390 ;
27D7 43      06400 MSG13     DB      'CD='
        44 3D
27DA 03      06410      DB      _ETX
                06420 ;
27DB 43      06430 MSG14     DB      'CTS='
        54 53 3D
27DF 03      06440      DB      _ETX
                06450 ;
27E0 4F      06460 MSG15     DB      'ON'
        4E
27E2 03      06470      DB      _ETX
                06480 ;
27E3 4F      06490 MSG16     DB      'OFF'
        46 46
27E6 03      06500      DB      _ETX
                06510 ;
27E7 4F      06520 MSG17     DB      'ODD'
        44 44
27EA 03      06530      DB      _ETX
                06540 ;
27EB 45      06550 MSG18     DB      'EVEN'
        56 45 4E
27EF 03      06560      DB      _ETX
                06570 ;
27F0 49      06580 MSG19     DB      'IGNORE'
        47 4E 4F 52 45
27F6 03      06590      DB      _ETX
                06600 ;
27F7 30      06610 MSG20     DB      '00000'
        30 30 30 30
27FC 03      06620      DB      _ETX
                06630 ;
27FD 58      06640 MSG21     DB      'X',_APOS
        27
27FF 30      06650 MSG21A    DB      '00',_APOS
        30 27
2802 03      06660      DB      _ETX
                06670 ;
2803 0D      06680 MSG22     DB      _CR
                06690 ;

```

2804	2C	06700	MSG23	DB	' , '
	20				
2806	03	06710		DB	_ETX
		06720 ;			
		06730 ;			Parameter evaluation table
		06740 ;			
2807	80	06750	PRMTBL\$	DB	80H ;Extended param
		06760 ;			
2808	C5	06770	BRTYP	DB	5!FLAG!NUM
2809	42	06780		DB	'BREAK'
	52 45 41	4B			
280E	00	06790	BRRESP	DB	0
280F	4A27	06800		DW	BREAKP
		06810 ;			
2811	55	06820		DB	5!ABB!FLAG
2812	51	06830		DB	'QUERY'
	55 45 52	59			
2817	00	06840	QRESP	DB	0
2818	4C27	06850		DW	QPARM
		06860 ;			
281A	94	06870	BTYP	DB	4!ABB!NUM
281B	42	06880		DB	'BAUD'
	41 55 44				
281F	00	06890	BRESP	DB	0
2820	4E27	06900		DW	BAUDP
		06910 ;			
2822	94	06920	WTYP	DB	4!ABB!NUM
2823	57	06930		DB	'WORD'
	4F 52 44				
2827	00	06940	WRESP	DB	0
2828	5027	06950		DW	WORDP
		06960 ;			
282A	94	06970	STYP	DB	4!ABB!NUM
282B	53	06980		DB	'STOP'
	54 4F 50				
282F	00	06990	SRESP	DB	0
2830	5227	07000		DW	STOPP
		07010 ;			
2832	76	07020	PTYP	DB	6!ABB!FLAG!STR
2833	50	07030		DB	'PARITY'
	41 52 49	54 59			
2839	00	07040	PRESP	DB	0
283A	5427	07050		DW	PARITYP
		07060 ;			
283C	54	07070	ETYP	DB	4!ABB!FLAG
283D	45	07080		DB	'EVEN'
	56 45 4E				
2841	00	07090	ERESP	DB	0
2842	5627	07100		DW	EVENP
		07110 ;			
2844	53	07120	OTYP	DB	3!ABB!FLAG
2845	4F	07130		DB	'ODD'
	44 44				
2848	00	07140	ORESP	DB	0
2849	5827	07150		DW	ODDP
		07160 ;			
284B	43	07170	DTYP	DB	3!FLAG
284C	44	07180		DB	'DTR'
	54 52				
284F	00	07190	DTRESP	DB	0
2850	5A27	07200		DW	DTRP

```

07210 ;
2852 43 07220 RTYP DB 3!FLAG
2853 52 07230 DB 'RTS'
54 53
2856 00 07240 RTRESP DB 0
2857 5C27 07250 DW RTSP
07260 ;
2859 42 07270 RITYP DB 2!FLAG
285A 52 07280 DB 'RI'
49
285C 00 07290 RIRESP DB 0
285D 5E27 07300 DW RIP
07310 ;
285F 42 07320 CDTYP DB 2!FLAG
2860 43 07330 DB 'CD'
44
2862 00 07340 CDRESP DB 0
2863 6027 07350 DW CDP
07360 ;
2865 43 07370 DSTYP DB 3!FLAG
2866 44 07380 DB 'DSR'
53 52
2869 00 07390 DSRESP DB 0
286A 6227 07400 DW DSRP
07410 ;
286C 43 07420 CTTYP DB 3!FLAG
286D 43 07430 DB 'CTS'
54 53
2870 00 07440 CTRESP DB 0
2871 6427 07450 DW CTSP
07460 ;
2873 57 07470 DB 7!ABB!FLAG
2874 44 07480 DB 'DEFAULT'
45 46 41 55 4C 54
287B 00 07490 DB 0
287C 6627 07500 DW DF Parm
07510 ;
07520 IF @MOD2
07530 DB 4!NUM
07540 DB 'PORT'
07550 PORTP DB 0
07560 DW PORTD
07570 ENDIF
07580 ;
287E 00 07590 DB 0 ;List terminator
07600 ;
07610 ; Evaluate and setup user parameters
07620 ;
07630 SETPARAM
287F 213524 07640 LD HL,PRMERR ;Param error exit vector
2882 E5 07650 PUSH HL ;Leave for quick exits
07660 ;
07670 ; Evaluate BAUD
07680 ;
2883 3A1F28 07690 SETBAUD LD A,(BRESP) ;Get baud response
2886 B7 07700 OR A ;Anything?
2887 2807 07710 JR Z,SETWORD ;Nope, go
2889 CDC229 07720 CALL CKBAUD ;Get baud setting
288C C0 07730 RET NZ ;Go if error
288D DD7706 07740 LD (IX+BAUDRT),A ; else save new baud
07750 ;

```



```

07760 ;      Evaluate WORD
07770 ;
2890 DD4E05 07780 SETWORD LD      C,(IX+UCIMAGE) ;P/u current setings
2893 3A2728 07790 LD      A,(WRESP) ;Get word response
2896 B7 07800 OR      A ;Anything?
2897 281C 07810 JR      Z,SETSTOP ;Nope, continue
2899 ED5B5027 07820 LD      DE,(WORDP) ;Get parameter
289D 14 07830 INC      D ;Check msb
289E 15 07840 DEC      D ;D<>0?
289F C0 07850 RET      NZ ;>256?
28A0 7B 07860 LD      A,E ;Get lsb
28A1 D605 07870 SUB      5 ;Adjust 0 relative
28A3 D8 07880 RET      C ;Go if out of range
28A4 FE04 07890 CP      4 ;5-8?
28A6 D0 07900 RET      NC ;Out of range
28A7 B7 07910 OR      A ;Clear carry, set PV
28A8 EAAD28 07920 JP      PE,$+5 ;Go if 5/8
28AB EE03 07930 XOR      3 ;Change 6=>7=>6
28AD 0F 07940 RRCA ;Align to bits 6/5
28AE 0F 07950 RRCA
28AF 0F 07960 RRCA
28B0 0660 07970 LD      B,01100000B ;For bit setup
28B2 CDAB29 07980 CALL    SETBITS ;Setup in C register
07990 ;
08000 ;      Evaluate STOP
08010 ;
28B5 3A2F28 08020 SETSTOP LD      A,(SRESP) ;Get response byte
28B8 B7 08030 OR      A ;Anything?
28B9 2817 08040 JR      Z,SETPAR ;Nope, continue
28BB ED5B5227 08050 LD      DE,(STOPP) ;Get input
28BF 14 08060 INC      D ;Check for out of range
28C0 15 08070 DEC      D ;D<>0?
28C1 C0 08080 RET      NZ ;>256
28C2 7B 08090 LD      A,E ;Get lsb
28C3 B7 08100 OR      A ;0?
28C4 C8 08110 RET      Z ;Invalid if not
28C5 FE03 08120 CP      3 ;1/2?
28C7 D0 08130 RET      NC ;Invalid
28C8 3D 08140 DEC      A ;Change 1|2 to 0|1
28C9 07 08150 RLCA ;Align result
28CA 07 08160 RLCA
28CB 07 08170 RLCA
28CC 07 08180 RLCA
28CD 0610 08190 LD      B,00010000B ;Init mask
28CF CDAB29 08200 CALL    SETBITS ;Setup in C register
08210 ;
08220 ;      Evaluate PARITY
08230 ;
28D2 3A3928 08240 SETPAR LD      A,(PRESP) ;Get parity response
28D5 B7 08250 OR      A ;Any input?
28D6 2825 08260 JR      Z,SETPEO ;Nope, check EVEN|ODD
08270 ;
08280 ;      Check if response was string or switch
08290 ;
28D8 ED5B5427 08300 LD      DE,(PARITYP) ;Get user pointer
28DC CB77 08310 BIT      6,A ;Switch?
28DE 2015 08320 JR      NZ,SETPOF ;Yes, set ON/OFF
08330 ;
08340 ;      Response was string, check 'E'|'O'
08350 ;
28E0 1A 08360 LD      A,(DE) ;Get input

```

```

28E1 CDB929 08370 CALL UCASE ;Make upper case
28E4 FE4F 08380 CP '0' ;Odd?
28E6 2807 08390 JR Z,SETPO ;Yes, go
28E8 FE45 08400 CP 'E' ;Even?
28EA C0 08410 RET NZ ;Neither, invalid
28EB CBF9 08420 SETPE SET 7,C ;Set EVEN parity
28ED 1802 08430 JR SETPAEO ;Continue
28EF CBB9 08440 SETPO RES 7,C ;Set ODD parity
28F1 CB99 08450 SETPAEO RES 3,C ;Set ON parity
28F3 182C 08460 JR SETBRK ;Continue
08470 ;
08480 ; Response was switch, check ON|OFF
08490 ;
28F5 7A 08500 SETPOF LD A,D ;Check input
28F6 B3 08510 OR E ;=0000?
28F7 CBD9 08520 SET 3,C ;Set OFF
28F9 2826 08530 JR Z,SETBRK ;Continue if yes
28FB CB99 08540 RES 3,C ;Set ON
08550 ;
08560 ; Check for ODD|EVEN params
08570 ;
28FD 3A4828 08580 SETPEO LD A,(ORESP) ;Get odd response
2900 B7 08590 OR A ;Anything?
2901 280C 08600 JR Z,SETPAE ;Nope, check EVEN
2903 ED5B5827 08610 LD DE,(ODDP) ;Get user input
2907 7A 08620 LD A,D ;Check for nil
2908 B3 08630 OR E ;DE=0000?
2909 CBF9 08640 SET 7,C ;ODD=off?
290B 2802 08650 JR Z,SETPAE ;Go if yes
290D CBB9 08660 RES 7,C ;ODD=on
08670 ;
290F 3A4128 08680 SETPAE LD A,(ERESP) ;Get even response
2912 B7 08690 OR A ;Anything?
2913 280C 08700 JR Z,SETBRK ;Nope, continue
2915 ED5B5627 08710 LD DE,(EVENP) ;Get param
2919 7A 08720 LD A,D ;Check for nil
291A B3 08730 OR E ;DE=0000?
291B CBB9 08740 RES 7,C ;EVEN=off?
291D 2802 08750 JR Z,SETBRK ;Go if yes
291F CBF9 08760 SET 7,C ;EVEN=on
08770 ;
08780 ; Evaluate BREAK
08790 ;
2921 3A0E28 08800 SETBRK LD A,(BRRESP) ;Get user response
2924 B7 08810 OR A ;Anything?
2925 2818 08820 JR Z,SETDTR ;Nope, continue
2927 ED5B4A27 08830 LD DE,(BREAKP) ;Get user input
292B CB77 08840 BIT 6,A ;Switch?
292D 2005 08850 JR NZ,SETBOF ;Yes, check ON|OFF
08860 ;
08870 ; Value entered, check if in range
08880 ;
292F 14 08890 INC D ;Check msb
2930 15 08900 DEC D ;D<>0?
2931 C0 08910 RET NZ ;>256?
2932 1808 08920 JR UPDBRK ;Continue
08930 ;
08940 ; Switch entered, check if on/off
08950 ;
2934 7A 08960 SETBOF LD A,D ;Get input
2935 B3 08970 OR E ;DE=0000?

```

```

2936 1E80      08980      LD      E,80H      ;Default
2938 2802      08990      JR      Z,UPDBRK    ;Go if off
293A 1E03      09000      LD      E,03H      ;Default
293C DD7307    09010  UPDBRK LD      (IX+BRK),E ;Update data
                09020 ;
                09030 ;      Evaluate DTR
                09040 ;
293F 3A4F28    09050  SETDTR LD      A,(DTRESP) ;Get user response byte
2942 B7        09060      OR      A          ;Anything?
2943 280C      09070      JR      Z,SETRTS    ;Continue if not
2945 ED5B5A27  09080      LD      DE,(DTRP)    ;Get user param
2949 7A        09090      LD      A,D        ;Check if anything
294A B3        09100      OR      E          ;DE=0000?
294B CBC9      09110      SET      1,C        ;Set OFF
294D 2802      09120      JR      Z,SETRTS    ;Go if yes
294F CB89      09130      RES      1,C        ;Set ON
                09140 ;
                09150 ;      Evaluate RTS
                09160 ;
2951 3A5628    09170  SETRTS LD      A,(RTRESP) ;Get user response byte
2954 B7        09180      OR      A          ;Anything?
2955 280C      09190      JR      Z,UPDCI     ;Continue if not
2957 ED5B5C27  09200      LD      DE,(RTSP)    ;Get user param
295B 7A        09210      LD      A,D        ;Check if 0
295C B3        09220      OR      E          ;DE=0000?
295D CBC1      09230      SET      0,C        ;Set OFF
295F 2802      09240      JR      Z,UPDCI     ;Go if yes
2961 CB81      09250      RES      0,C        ;Set ON
2963 CBD1      09260  UPDCI SET      2,C        ;Enable transmit
2965 DD7105    09270      LD      (IX+UCIMAGE),C ;Update new data
                09280 ;
                09290 ;      Evaluate RI
                09300 ;
2968 215C28    09310  SETRI  LD      HL,RIRESP ;Response byte
296B 0E01      09320      LD      C,00000001B ;Bit to set
296D CD8A29    09330      CALL   SETIT       ;Setup data
                09340 ;
                09350 ;      Evaluate DSR
                09360 ;
2970 216928    09370  SETDSR LD      HL,DSRESP ;Response byte
2973 0E04      09380      LD      C,00000100B ;Bit mask
2975 CD8A29    09390      CALL   SETIT       ;Setup data
                09400 ;
                09410 ;      Evaluate CD
                09420 ;
2978 216228    09430  SETCD  LD      HL,CDRESP ;Response byte
297B 0E02      09440      LD      C,00000010B ;Bit mask
297D CD8A29    09450      CALL   SETIT       ;Setup data
                09460 ;
                09470 ;      Evaluate CTS
                09480 ;
2980 217028    09490  SETCTS LD      HL,CTRESP ;Response byte
2983 0E08      09500      LD      C,00001000B ;Bit mask
2985 CD8A29    09510      CALL   SETIT       ;Setup data
                09520 ;
                09530 ;      Evaluation complete
                09540 ;
2988 F1        09550      POP      AF          ;Remove param error vect
2989 C9        09560      RET              ;Return from evaluate
                09570 ;
298A 34        09580  SETIT  INC      (HL)      ;Any response?

```

```

298B 35      09590      DEC      (HL)      ;(HL) = 00?
298C C8      09600      RET      Z          ;Back if no change
298D 23      09610      INC      HL         ;Point to response data
298E CDB429  09620      CALL     GETHL      ;Load HL with (HL)
2991 CDB429  09630      CALL     GETHL      ;Get user response
2994 79      09640      LD       A,C        ;Get response data ON
2995 07      09650      RLCA          ;Align to upper 4 bits
2996 07      09660      RLCA
2997 07      09670      RLCA
2998 07      09680      RLCA
2999 B1      09690      OR        C          ;Combine
299A 47      09700      LD       B,A        ;Set response data OFF
299B 2F      09710      CPL          ;Reverse bits
299C DDA604  09720      AND      (IX+MSMASK) ;Drop desired bits
299F 5F      09730      LD       E,A        ;Save new mask
29A0 7C      09740      LD       A,H        ;Check if OFF
29A1 B5      09750      OR        L          ;HL = 0000?
29A2 78      09760      LD       A,B        ;Get OFF bits
29A3 2801    09770      JR      Z,$+3      ;Go if off
29A5 79      09780      LD       A,C        ;Get ON bits
29A6 B3      09790      OR        E          ;Combine with remainder
29A7 DD7704  09800      LD       (IX+MSMASK),A ;Update mask
29AA C9      09810      RET          ;Done
          09820 ;
          09830 ;      Set bits in C from A using B as mask
          09840 ;
29AB A0      09850 SETBITS AND      B          ;Mask others
29AC F5      09860 PUSH     AF          ;Save data
29AD 78      09870 LD       A,B          ;Get mask
29AE 2F      09880 CPL          ;Reverse for mask off
29AF A1      09890 AND      C          ;Remove undesired bits
29B0 C1      09900 POP      BC          ;B = new bits
29B1 B0      09910 OR        B          ;Combine
29B2 4F      09920 LD       C,A          ;Update
29B3 C9      09930 RET          ;New bits set
          09940 ;
          09950 ;      Fetch HL from (HL)
          09960 ;
29B4 7E      09970 GETHL  LD       A,(HL)      ;Get lsb
29B5 23      09980 INC      HL         ;Bump pointer
29B6 66      09990 LD       H,(HL)      ;Get msb
29B7 6F      10000 LD       L,A          ;HL = (HL)
29B8 C9      10010 RET          ;Done
          10020 ;
          10030 ;      Convert char in A to upper case
          10040 ;
29B9 FE61    10050 UCASE  CP       'a'        ;In range?
29BB D8      10060 RET      C          ;Nope, go
29BC FE7B    10070 CP       'z'+1      ;In range?
29BE D0      10080 RET      NC         ;Nope, go
29BF E65F    10090 AND      5FH        ; else make upper case
29C1 C9      10100 RET          ;Done
          10110 ;
          10120 ;      Check for valid baud rate entered
          10130 ;
29C2 ED5B4E27 10140 CKBAUD LD      DE,(BAUDP) ;Baud parm (default 300)
29C6 21FA26   10150 LD      HL,BAUDTBL ;Point to baud table
29C9 010010   10160 LD      BC,16<8+0 ;B=count, C=position
          10170 ;
29CC 7E      10180 BLOOP  LD      A,(HL)      ;Fetch LSB baud
29CD 23      10190 INC      HL         ;Bump table

```

```

29CE BB      10200      CP      E      ;Lsb match?
29CF 2004    10210      JR      NZ,NOMATB ;Nope, go next entry
29D1 7E      10220      LD      A,(HL)    ;Get MSB baud
29D2 BA      10230      CP      D      ;Msb match?
29D3 2805    10240      JR      Z,MATCHB   ;Yes, baud found
          10250 ;
29D5 23      10260 NOMATB INC      HL      ;Bump to next entry
29D6 0C      10270      INC      C      ;Bump position count
29D7 10F3    10280      DJNZ     BLOOP    ;Go for table length
29D9 C9      10290      RET      ;Param error
          10300 ;
29DA 79      10310 MATCHB LD      A,C      ;Pick up baud rate code
29DB 07      10320      RLCA           ;Align to high bits
29DC 07      10330      RLCA
29DD 07      10340      RLCA
29DE 07      10350      RLCA
29DF B1      10360      OR      C      ;Use for xmit and rcv
29E0 BF      10370      CP      A      ;Z=good value
29E1 C9      10380      RET      ;Return with BAUD setting
          10390 ;
          10400 ;      Nil params entered, display current settings
          10410 ;
29E2          10420 SHOW EQU      $
          10430      IF      @MOD2
          10440      LD      A,-1      ;Set NO flag
          10450      LD      (SHOWF),A ;Save flag
          10460      LD      DE,MDNAME1 ;CL1
          10470      LD      A,'A'     ;Comm A
          10480      LD      (MSG1A),A ;To text
          10490      CALL     SHOW1    ;Display
          10500      LD      DE,MDNAME2 ;CL2
          10510      LD      A,'B'     ;Comm B
          10520      LD      (MSG1A),A ;To text
          10530      CALL     SHOW1    ;Display
          10540      LD      A,-1      ;Get flag
          10550 SHOWF EQU      $-1
          10560      OR      A      ;Any found?
          10570      JP      NZ,BADMOD ;Not installed
          10580      RET      ; else OK
          10590 ;
29E2          10600 SHOW1 @@GTMOD      ;Locate module
          00037      LD      A,83
          00038      RST      40
          10610      RET      NZ      ;Module not found
          10620      XOR      A      ;Set module found
          10630      LD      (SHOWF),A ;Init flag
          10640      PUSH     DE      ;Pass to IX
          10650      POP      IX     ;IX => module
          10660      ENDIF
          10670 ;
29E2 C33126  10680      JP      DSPALL    ;Display all settings
          10690 ;
          10700 ;      Mini param block for 'query' evaluation
          10710 ;
29E5 80      10720 PTBL2 DB      80H      ;Extended param
          10730 ;
29E6 C1      10740 FTYP DB      FLAG!NUM!1
29E7 46      10750      DB      'F'
29E8 00      10760 FRESP DB      0
29E9 6A27    10770      DW      FPARM
          10780 ;

```

```
29EB 00      10790      DB      0
              10800 ;
              10810 ;      Text string passed to @PARAM for query eval
              10820 ;
29EC 28      10830 FSTR    DB      '(F='
      46 3D
29EF          10840 INBUF   EQU      $              ;Keyboard input buffer
              10850 ;
2400          10860      END      BEGIN
```

\$ABORT	2411	\$DSP	2416	\$DSPLY	241C
\$EXIT	2407	@@1	0000	@@2	0000
@@3	0000	@@4	0000	@MOD2	0000
@MOD4	FFFF	ABB	0010	ASKBAUD	246E
ASKBRK	24E1	ASKCD	250E	ASKCTS	2517
ASKDSR	2505	ASKDTR	24EA	ASKPAR	24BF
ASKRI	24FC	ASKRTS	24F3	ASKSTOP	24A0
ASKWORD	2484	BADBAUD	247E	BADDCB	26E1
BADMOD	242B	BADPAR	24BB	BADSTOP	24B5
BADWORD	249A	BAUDP	274E	BAUDRT	0006
BAUDTBL	26FA	BEGIN	2400	BLOOP	29CC
BREAKP	274A	BRESP	281F	BRK	0007
BRRESP	280E	BR TYP	2808	BTYP	281A
CDP	2760	CDRESP	2862	CDTYP	285F
CKBAUD	29C2	CKPARM	2520	CKRSP	2663
CIKESP	2870	CISP	2764	CTTYP	286C
DEFAULT	26CC	DEFBA	0055	DEFBR	0003
DE FMS	0000	DEFUC	00A5	DFPARM	2766
DMSG0	260C	DMSG0X	2622	DMSG1	252F
DMSG10	25EC	DMSG11	25F2	DMSG12	25F9
DMSG13	2600	DMSG14	2607	DMSG2	2538
DMSG22	2625	DMSG23	262B	DMSG3	2566
DMSG4	257F	DMSG5	2591	DMSG6	25B0
DMSG7	25C5	DMSG8	25CB	DMSG89	25E1
DMSG9	25D7	DSPALL	2631	DSPDONE	264D
DSPRET	2647	DSPSET	2637	DSPTBL	271A
DSRESP	2869	DSRP	2762	DSTYP	2865
DTRESP	284F	DTRP	275A	DTYP	284B
ERESP	2841	ERROR\$	2427	ETYP	283C
EVENP	2756	FLAG	0040	F Parm	276A
FRESP	29E8	FSTR	29EC	FTYP	29E6
GETHL	29B4	GETIT	2650	GETRSP	2674
GETRSPR	267A	INBUF	29EF	IOERR	2420
MATCHB	29DA	MDNAME	26DD	MSG1	276C
MSG10	27BD	MSG11	27CE	MSG12	27D2
MSG13	27D7	MSG14	27DB	MSG15	27E0
MSG16	27E3	MSG17	27E7	MSG18	27EB
MSG19	27F0	MSG2	2780	MSG20	27F7
MSG21	27FD	MSG21A	27FF	MSG22	2803
MSG23	2804	MSG3	2786	MSG4	278C
MSG5	2792	MSG6	279A	MSG7	27A1
MSG8	27B3	MSG9	27B8	MSMASK	0004
MVUP	2666	NOMATB	29D5	NUM	0080
ODDP	2758	OFFSET	0004	ORESP	2848
OTYP	2844	PARITYP	2754	PAR ERR	002C
PGRM	243A	PORTD	2768	PORTF	245C
PRESP	2839	PRMERR	2435	PRMTBL\$	2807
PROMPT	26F7	PTBL2	29E5	PTYP	2832
QPARM	274C	QRESP	2817	RIP	275E
RIRESP	285C	RITYP	2859	RTRESP	2856
RTSP	275C	RTYP	2852	SAVESP	240A
SETBAUD	2883	SETBITS	29AB	SETBOF	2934
SETBRK	2921	SETCD	2978	SETCTS	2980
SETDSR	2970	SETDTR	293F	SETIT	298A
SETPAE	290F	SETPAEO	28F1	SETPAR	28D2
SETPARAM	287F	SETPE	28EB	SETPEO	28FD
SETPO	28EF	SETPOF	28F5	SETRI	2968
SETRTS	2951	SETSTOP	28B5	SETWORD	2890
SHOW	29E2	SHOW1	29E2	SPLP	2448
SRESP	282F	STOPP	2752	STR	0020

STRIP1	255E STYP	282A TTYP	26A0
UCASE	29B9 UCIMAGE	0005 UP	266A
UPDBRK	293C UPDCI	2963 WORDP	2750
WRESP	2827 WTYP	2822 APOS	0027
_CR	000D ETX	0003 _INIT	0002
_LF	000A @ABORT	C2EB @@ADTSK	C37E
@BANK	C896 @BKSP	C576 @BREAK	C8AC
@CHNIO	C2D6 @CKBRKC	C8FA @CKDRV	C3D2
@CKEOF	C58B @CKTSK	C369 @CLOSE	C561
@CLS	C8E4 @CMNDI	C315 @CMNDR	C32A
@CTL	C13A @DATE	C2AC @DCSTAT	C411
@DEBUG	C354 @DECHEX	C816 @DIRRD	C783
@DIRWR	C798 @DIV16	C801 @DIV8	C7EC
@DODIR	C3E7 @DSP	C0FE @DSPLY	C19E
@ERROR	C33F @EXIT	C300 @FEXT	C6F0
@FLAGS	C880 @FNAME	C705 @FSPEC	C6DB
@GATRD	C76E @GATWR	C7AD @GET	C112
@GTDCB	C72F @GTDCT	C71A @GTMOD	C744
@HDFMT	C4B9 @HEX16	C855 @HEX8	C840
@HEXDEC	C82B @HIGH\$	C86A @INIT	C537
@KBD	C176 @KEY	C0EA @KEYIN	C18A
@KLTSK	C3BD @LOAD	C6B1 @LOC	C5A0
@LOF	C5B5 @LOGER	C1D5 @LOGOT	C1EA
@MSG	C221 @MUL16	C7D7 @MUL8	C7C2
@OPEN	C54C @PARAM	C297 @PAUSE	C282
@PEOF	C5CA @POSN	C5DF @PRINT	C236
@PRT	C14E @PUT	C126 @RAMDIR	C3FC
@RDSEC	C48F @RDSSC	C759 @READ	C5F4
@REMOV	C522 @RENAM	C50D @REW	C609
@RMTSK	C393 @RPTSK	C3A8 @RREAD	C61E
@RSLCT	C47A @RSTOR	C43B @RUN	C6C6
@RWRTIT	C633 @SEEK	C465 @SEEKSC	C648
@SKIP	C65D @SLCT	C426 @STEPI	C450
@TIME	C2C1 @VCTL	C26D @VER	C672
@VRSEC	C4A4 @WEOF	C687 @WHERE	C162
@WRITE	C69C @WRSEC	C4CE @WRSSC	C4E3
@WRTRK	C4F8		

2400 is the transfer address

00000 Total errors

NOTES:

Command: SETKI

Library: SYS8/SYS

ISAM # : B3H

```

00100 ;LBSETKI/ASM - Set Keyboard Parameters
00000 00110 TITLE <SETKI - LS-DOS 6.2>
00120 ;
0004 00130 OFFSET EQU 4 ;Length: end of name to data area
0006 00140 DELAY EQU OFFSET+2
0007 00150 RPEAT EQU OFFSET+3
000A 00160 DMIN EQU 10
0001 00170 RMIN EQU 1
0016 00180 DDFALT EQU 22
0002 00190 RDFALT EQU 2
0003 00200 ETX EQU 03H
000D 00210 CR EQU 0DH
000A 00220 LF EQU 0AH
002C 00230 PAR_ERR EQU 44 ;Parameter Error
00240 ;
0040 00250 FLAG EQU 01000000B
0010 00260 ABB EQU 00010000B
0080 00270 NUM EQU 10000000B
00280 ;
0000 00290 *GET SVCMAC:3 ;SVC Macro equivalents
00010 ;SVMAC/ASM - LS-DOS Version VI
00020 *LIST OFF
03900 *LIST ON
00300 ;
2400 00310 ORG 2400H
00320 ;
2400 00330 BEGIN EQU $
2400 ED730B24 00340 LD (SAVE$P+1),SP
2404 CD2E24 00350 CALL PGRM ;Exit via RET
00360 ;
00370 ; Set exit conditions
00380 ;
2407 210000 00390 $EXIT LD HL,0 ;Init to no error
240A 310000 00400 SAVESP LD SP,$-$ ;P/u original SP
240D 00410 @@CKBRKC ;Clear any <BREAK>
240D 3E6A 00001 LD A,106
240F EF 00002 RST 40
2410 C9 00420 RET
2411 21FFFF 00430 $ABORT LD HL,-1 ;Set abort code
2414 18F4 00440 JR SAVESP
00450 ;
2416 00460 $DSP @@DSP ;Display a character
2416 3E02 00003 LD A,2
2418 EF 00004 RST 40
2419 C8 00470 RET Z ;Back if good
241A 1807 00480 JR IOERR
241C 00490 $DSPLY @@DSPLY ;Display a line
00005 IFEQ 00H,1
00006 LD HL,
00007 ENDIF
241C 3E0A 00008 LD A,10
241E EF 00009 RST 40
241F C8 00500 RET Z ;Back if good
2420 21 00510 DB 21H ;Skip LD A,##
2421 3E2C 00520 PRMERR LD A,PAR_ERR ;Parameter Error
00530 ;
00540 ; I/O Error Processing
00550 ;
2423 2600 00560 IOERR LD H,0
2425 6F 00570 LD L,A ;Save error #
2426 F6C0 00580 OR 0C0H

```

```

2428 4F      00590      LD      C,A
2429      00600      @@ERROR      ;Display message
2429 3E1A    00010      LD      A,26
242B EF      00011      RST      40
242C 18DC    00610      JR      SAVESP      ;Exit
          00620 ;
          00630 ;      PGRM - Set Keyboard Parameters
          00640 ;
242E E5      00650 PGRM    PUSH    HL      ;Save cmdline ptr
242F 113925  00660      LD      DE,MDNAME ;Name of keyboard driver
2432      00670      @@GTMOD      ;Find module header
2432 3E53    00012      LD      A,83
2434 EF      00013      RST      40
2435 3E08    00680      LD      A,8      ;Device not available
2437 C22324  00690      JP      NZ,IOERR      ;Exit if not found
243A D5      00700      PUSH    DE      ;Point to next byte
243B DDE1    00710      POP     IX      ; after module name
          00720 ;
243D E1      00730      POP     HL      ;=>cmd line
243E CD1725  00740      CALL   SKSP      ;Move to non-space
2441 7E      00750      LD      A,(HL)      ;Char fm cmd line
2442 FE28    00760      CP      '('      ;Any params?
2444 CA6224  00770      JP      Z,GETNEW      ;Get/set new values
          00780 ;
          00790 ;      Display old values
          00800 ;
2447 CD5024  00810      CALL   SETMSG      ;Move old values for dsply
244A 211E25  00820      LD      HL,DMSG      ;Pt to dsply string
244D C31C24  00830      JP      $DSPLY      ;Dsply and exit
          00840 ;
          00850 ;      Set up values in string
          00860 ;
2450 DD7E06  00870 SETMSG  LD      A,(IX+DELAY) ;P/u old Wait value
2453 112725  00880      LD      DE,DDELAY      ;=>buffer to receive
2456 CD6625  00890      CALL   HEXDEC      ;Convert to ASCII decimal
2459 DD7E07  00900      LD      A,(IX+RPEAT) ;P/u old Rate value
245C 113525  00910      LD      DE,DRPEAT      ;Pt to dsply area
245F C36625  00920      JP      HEXDEC      ;Convert to decimal ASCII
          00930 ;
2462 114025  00940 GETNEW  LD      DE,PRMTBL$      ;Check user parameters
2465      00950      @@PARAM
2465 3E11    00014      LD      A,17
2467 EF      00015      RST      40
2468 C22124  00960      JP      NZ,PRMERR      ;Go on "Parm error
246B 110000  00970      LD      DE,$-$      ;P/u D parm
246C      00980 DF Parm EQU    $-2
246E 7A      00990      LD      A,D      ;Default setting wanted?
246F B3      01000      OR      E
2470 2808    01010      JR      Z,SETSTR      ;Keep existing if not
2472 DD360616 01020      LD      (IX+DELAY),DDFALT      ;Stuff defaults
2476 DD360702 01030      LD      (IX+RPEAT),RDFALT
247A CD5024  01040 SETSTR  CALL   SETMSG      ;Put numbers in string
247D 110000  01050 QCHK   LD      DE,$-$
247E      01060 QPARM  EQU    $-2      ;Query parm used?
2480 7A      01070      LD      A,D
2481 B3      01080      OR      E
2482 2834    01090      JR      Z,CKPARM      ;Go if not
          01100 ;
2484 3A5925  01110      LD      A,(WRESP) ;P/u Wait response byte
2487 E680    01120      AND     NUM      ;If a "W" number given,
2489 2013    01130      JR      NZ,CKR      ; then ask for R only

```

```

248B 211E25 01140 GETD LD HL,DMSG ;Pt to Wait prompt msg
248E CDDE24 01150 CALL SHOW ;Ask for W time
2491 CDAA25 01160 CALL GETIT ;Get it (in DE)
2494 2808 01170 JR Z,CKR ;Don't change if CR only
2496 CDA125 01180 CALL DCHECK ;Check if good value
2499 20F0 01190 JR NZ,GETD ;Don't change if not
249B DD7706 01200 LD (IX+DELAY),A ;Store new Wait
01210 ;
249E 3A5125 01220 CKR LD A,(RRESP) ;P/u Rate response byte
24A1 E680 01230 AND NUM ; and go if Rate
24A3 2013 01240 JR NZ,CKPARAM ; entered on cmdline
24A5 212C25 01250 LD HL,RMSG ;Pt to Rate prompt msg
24A8 CDDE24 01260 CALL SHOW
24AB CDAA25 01270 CALL GETIT ;Get new Rate
24AE 2808 01280 JR Z,CKPARAM ;Don't change if CR only
24B0 CD9D25 01290 CALL RCHECK ;Check range & don't
24B3 20E9 01300 JR NZ,CKR ; change if out of range
24B5 DD7707 01310 LD (IX+RPEAT),A ;Store new Rate
01320 ;
01330 ; Check entries given on cmd line
01340 ;
24B8 3A5925 01350 CKPARAM LD A,(WRESP)
24BB E680 01360 AND NUM ;W parm response
24BD 280C 01370 JR Z,CKRP ;Go if not entered
24BF 110000 01380 LD DE,$-$ ;P/u Wait value
24C0 01390 WPARAM EQU $-2
24C2 CDA125 01400 CALL DCHECK ;Check value
24C5 C22124 01410 JP NZ,PRMERR ;Go if bad
24C8 DD7706 01420 LD (IX+DELAY),A ; else store it
01430 ;
24CB 3A5125 01440 CKRP LD A,(RRESP) ;Rate response
24CE E680 01450 AND NUM
24D0 C8 01460 RET Z ;Done if none
24D1 110000 01470 LD DE,$-$ ;P/u Rate value
24D2 01480 RPARAM EQU $-2
24D4 CD9D25 01490 CALL RCHECK ;Check value
24D7 C22124 01500 JP NZ,PRMERR ; and go if bad
24DA DD7707 01510 LD (IX+RPEAT),A ; else store it
24DD C9 01520 RET ;Done
01530 ;
24DE 060C 01540 SHOW LD B,12 ;Col posn
24E0 4E 01550 SH2 LD C,(HL) ;Get char in C
24E1 CD1624 01560 CALL $DSP ;Print byte fm string
24E4 05 01570 DEC B ;Dec chars to print
24E5 23 01580 INC HL ;Bump string ptr
24E6 3E3D 01590 LD A,'='
24E8 B9 01600 CP C
24E9 20F5 01610 JR NZ,SH2 ;Display up to =
24EB 0E20 01620 LD C,' ' ;Then space
24ED CD1624 01630 CALL $DSP
24F0 CD1725 01640 CALL SKSP ;Move to number
24F3 0E7B 01650 LD C,'{'
24F5 CD1624 01660 CALL $DSP ;Dsply opening brace
24F8 4E 01670 SH3 LD C,(HL) ;P/u character
24F9 3E2F 01680 LD A,'0'-1 ;Check C for numeric value
24FB B9 01690 CP C
24FC 3007 01700 JR NC,SH4 ;Go if not
24FE CD1624 01710 CALL $DSP ; else dsply,
2501 05 01720 DEC B ; dec chars remaining,
2502 23 01730 INC HL ; pt to next char in string
2503 18F3 01740 JR SH3 ; and loop

```

```

2505 0E7D      01750 SH4      LD      C,'}'      ;Display closing brace
2507 CD1624    01760          CALL    $DSP
250A 0E20      01770          LD      C,' '
250C CD1624    01780 SH5      CALL    $DSP      ;Tab remaining distance
250F 10FB      01790          DJNZ    SH5
2511 213D25    01800          LD      HL,ENDSTR      ;Print "?"
2514 C31C24    01810          JP      $DSPLY
                01820 ;
2517 3E20      01830 SKSP     LD      A,' '      ;Bypass all leading spaces
2519 BE        01840 SKP2     CP      (HL)
251A C0        01850          RET      NZ
251B 23        01860          INC     HL
251C 18FB      01870          JR      SKP2
                01880 ;
                01890 ;
                01900 ;
251E 57        01910 DMSG     DB      'Wait  = '
        61 69 74 20 20 20 3D 20
2527 20        01920 DDELAY    DB      ' , '
        20 20 2C 20
252C 52        01930 RMSG     DB      'Rate  = '
        61 74 65 20 20 20 3D 20
2535 20        01940 DRPEAT    DB      ' ',CR
        20 20 0D
2539 24        01950 MDNAME    DB      '$KI',ETX
        4B 49 03
253D 3F        01960 ENDSTR    DB      '? ',ETX
        20 03
                01970 ;
2540 80        01980 PRMTBL$   DB      80H
                01990 ;
2541 57        02000          DB      FLAG!ABB!7
2542 44        02010          DB      'DEFAULT'
        45 46 41 55 4C 54
2549 00        02020          DB      0
254A 6C24      02030          DW      DF PARM
                02040 ;
254C 94        02050          DB      ABB!NUM!4
254D 52        02060          DB      'RATE'      ;Repeat key rate
        41 54 45
2551 00        02070 RRESP     DB      0
2552 D224      02080          DW      RPARM
                02090 ;
2554 94        02100          DB      ABB!NUM!4
2555 57        02110          DB      'WAIT'      ;Delay before repeat
        41 49 54
2559 00        02120 WRESP     DB      0
255A C024      02130          DW      WPARM
                02140 ;
255C 55        02150          DB      ABB!FLAG!5
255D 51        02160          DB      'QUERY'
        55 45 52 59
2562 00        02170          DB      0
2563 7E24      02180          DW      QPARM
                02190 ;
2565 00        02200          NOP      ;Note end of parm table
                02210 ;
                02220 ;
                02230 ;
                02240 ;
                02250 ;

```

HEXDEC - Convert Hex Number to Decimal ASCII
A => 8-bit Hex Number to Convert
DE => Destination of ASCII characters

```

02260 ;
02270 ;
2566 C5 02280 HEXDEC PUSH BC ;Save regs
2567 E5 02290 PUSH HL
2568 F5 02300 PUSH AF
02310 ;
02320 ; Xfer number to HL
02330 ;
2569 2600 02340 LD H,0 ;Set HL = #
256B 6F 02350 LD L,A
02360 ;
256C 3E20 02370 LD A,' ' ;Character if leading 0
256E 016400 02380 LD BC,100 ;Set 10's power
2571 CD8225 02390 CALL CVD1 ;Convert to ASCII
2574 010A00 02400 LD BC,10
2577 CD8225 02410 CALL CVD1
257A 7D 02420 LD A,L ;Get remainder
257B C630 02430 ADD A,'0' ;Make ASCII and
257D 12 02440 LD (DE),A ; stuff in buffer
02450 ;
257E F1 02460 POP AF ;Recover #
257F E1 02470 POP HL ;And other regs
2580 C1 02480 POP BC
2581 C9 02490 RET
02500 ;
2582 D5 02510 CVD1 PUSH DE ;Save user buffer
2583 5F 02520 LD E,A ;Save pad character
2584 16FF 02530 LD D,0FFH ;Init digit count to -1
2586 AF 02540 XOR A
2587 14 02550 CVD2 INC D ;Inc digit count and
2588 ED42 02560 SBC HL,BC ; sub 10' power untill
258A 30FB 02570 JR NC,CVD2 ; underflow
258C 09 02580 ADD HL,BC ;Add back last sub
258D 7B 02590 LD A,E ;Recover pad char
258E 42 02600 LD B,D ;Count to B
258F D1 02610 POP DE ;Recover buffer ptr
2590 12 02620 LD (DE),A ;Tempt store pad char
2591 04 02630 INC B ;See if char a 0
2592 05 02640 DEC B
2593 2806 02650 JR Z,CVD3 ;Go if so
2595 78 02660 LD A,B ; else make digit ASCII
2596 C630 02670 ADD A,'0'
2598 12 02680 LD (DE),A ; and put in buffer
2599 3E30 02690 LD A,'0' ;Change the pad char
259B 13 02700 CVD3 INC DE ;Bump buffer
259C C9 02710 RET
02720 ;
259D 1601 02730 RCHECK LD D,RMIN ;Rate minimum value
259F 1802 02740 JR CHECK
25A1 160A 02750 DCHECK LD D,DMIN ;Wait minimum value
25A3 7B 02760 CHECK LD A,E ;Get number
25A4 E67F 02770 AND 7FH ;Keep positive
25A6 BA 02780 CP D ;Lowest allowed
25A7 D8 02790 RET C ;Too low
25A8 BF 02800 CP A ;Set Z if good value
25A9 C9 02810 RET
02820 ;
25AA 21D725 02830 GETIT LD HL,INBUF ;Key buffer
25AD 010003 02840 LD BC,3<8 ;3 chars max
25B0 02850 @@KEYIN
25B0 3E09 02860 LD A,9

```

```

25B2 EF      00017      RST      40
25B3 DA1124  02860      JP       C,$ABORT      ;Quit if Break
           02870 ;
           02880 ;      DECHEX - Decimal ASCII to Hex
           02890 ;      DE <= returns the Hex num
           02900 ;      HL => points to the start of Dec asc string
           02910 ;      A <= # of characters converted
           02920 ;
           02930 ;
25B6 0600    02940 DECHX LD      B,0      ;Init counter to 0
25B8 50      02950      LD      D,B      ;Init ret valu to 0
25B9 58      02960      LD      E,B
25BA 7E      02970 CVDEC LD      A,(HL)   ;P/u a character
25BB D630    02980      SUB     30H      ;Make binary
25BD FE0A    02990      CP      10
25BF 3013    03000      JR      NC,DONECON ;Leave when non-decimal found
25C1 E5      03010      PUSH    HL
25C2 62      03020      LD      H,D
25C3 6B      03030      LD      L,E      ;Prev. total to HL
25C4 29      03040      ADD     HL,HL     ;X2
25C5 29      03050      ADD     HL,HL     ;X4
25C6 19      03060      ADD     HL,DE     ;X5
25C7 29      03070      ADD     HL,HL     ;X10
25C8 EB      03080      EX      DE,HL    ;Result back to DE
25C9 83      03090      ADD     A,E      ;Add in newest digit
25CA 5F      03100      LD      E,A
25CB 3E00    03110      LD      A,0
25CD 8A      03120      ADC     A,D
25CE 57      03130      LD      D,A
25CF E1      03140      POP     HL      ;Get buffer posn
25D0 23      03150      INC     HL
25D1 04      03160      INC     B      ;Inc chars found
25D2 18E6    03170      JR      CVDEC    ; and continue next
           03180 ;
25D4 78      03190 DONECON LD     A,B
25D5 B7      03200      OR      A
25D6 C9      03210      RET
25D7         03220 INBUF EQU     $
           03230 ;
2400         03240      END     BEGIN

```


\$ABORT	2411 \$DSP	2416 \$DSPLY	241C
\$EXIT	2407 @@1	0000 @@2	0000
@@3	0000 @@4	0000 @MOD2	0000
@MOD4	FFFF ABB	0010 BEGIN	2400
CHECK	25A3 CKPARM	24B8 CKR	249E
CKRP	24CB CR	000D CVD1	2582
CVD2	2587 CVD3	259B CVDEC	25BA
DCHECK	25A1 DDELAY	2527 DDFALT	0016
DECHEX	25B6 DELAY	0006 DFPARM	246C
DMIN	000A DMSG	251E DONECON	25D4
DRPEAT	2535 ENDSTR	253D ETX	0003
FLAG	0040 GETD	248B GETIT	25AA
GETNEW	2462 HEXDEC	2566 INBUF	25D7
IOERR	2423 LF	000A MDNAME	2539
NUM	0080 OFFSET	0004 PAR_ERR	002C
PGRM	242E PRMERR	2421 PRMTBL\$	2540
QCHK	247D QPARM	247E RCHECK	259D
RDFALT	0002 RMIN	0001 RMSG	252C
RPARM	24D2 RPEAT	0007 RRESP	2551
SAVESP	240A SETMSG	2450 SETSTR	247A
SH2	24E0 SH3	24F8 SH4	2505
SH5	250C SHOW	24DE SKP2	2519
SKSP	2517 WPARM	24C0 WRESP	2559
@@ABORT	8733 @@ADTSK	87C6 @@BANK	8CDE
@@BKSP	89BE @@BREAK	8CF4 @@CHNIO	871E
@@CKBRKC	8D42 @@CKDRV	881A @@CKEOF	89D3
@@CKTSK	87B1 @@CLOSE	89A9 @@CLS	8D2C
@@CMNDI	875D @@CMNDR	8772 @@CTL	8582
@@DATE	86F4 @@DCSTAT	8859 @@DEBUG	879C
@@DECHEX	8C5E @@DIRRD	8BCB @@DIRWR	8BE0
@@DIV16	8C49 @@DIV8	8C34 @@DODIR	882F
@@DSP	8546 @@DSPLY	85E6 @@ERROR	8787
@@EXIT	8748 @@FEXT	8B38 @@FLAGS	8CC8
@@FNAME	8B4D @@FSPEC	8B23 @@GATRD	8BB6
@@GATWR	8BF5 @@GET	855A @@GTDGB	8B77
@@GTDCT	8B62 @@GTMOD	8B8C @@HDFMT	8901
@@HEX16	8C9D @@HEX8	8C88 @@HEXDEC	8C73
@@HIGH\$	8CB2 @@INIT	897F @@KBD	85BE
@@KEY	8532 @@KEYIN	85D2 @@KLTSK	8805
@@LOAD	8AF9 @@LOC	89E8 @@LOF	89FD
@@LOGGER	861D @@LOGOT	8632 @@MSG	8669
@@MUL16	8C1F @@MUL8	8C0A @@OPEN	8994
@@PARAM	86DF @@PAUSE	86CA @@PEOF	8A12
@@POSN	8A27 @@PRINT	867E @@PRT	8596
@@PUT	856E @@RAMDIR	8844 @@RDSEC	88D7
@@RDSSC	8BA1 @@READ	8A3C @@REMOV	896A
@@RENAM	8955 @@REW	8A51 @@RMTSK	87DB
@@RPTSK	87F0 @@RREAD	8A66 @@RSLCT	88C2
@@RSTOR	8883 @@RUN	8B0E @@RWRT	8A7B
@@SEEK	88AD @@SEEKSC	8A90 @@SKIP	8AA5
@@SLCT	886E @@STEPI	8898 @@TIME	8709
@@VDCTL	86B5 @@VER	8ABA @@VRSEC	88EC
@@WEOF	8ACF @@WHERE	85AA @@WRITE	8AE4
@@WRSEC	8916 @@WRSSC	892B @@WRTRK	8940

2400 is the transfer address
 00000 Total errors

NOTES:

NOTES:

Command: SP00L

Library: SYS8/SYS

ISAM # : A2H

```

00100 ;LBSPPOOL/ASM - Spool command
0000 00110 TITLE <SPOOL - LS-DOS 6.2>
00120 ;
0000 00130 *GET LBSPPOOLA:3
00010 ;LBSPPOOLA/ASM - SPOOL setup
00020 ;
002C 00030 PAR_ERR EQU 44 ;Parameter Error
0003 00040 ETX EQU 3
000D 00050 CR EQU 13
0028 00060 RST28 EQU 28H
2300 00070 LOWBUF$ EQU 2300H ;Low memory disk I/O buff
001F 00080 KITSK@ EQU 31 ;FLGTAB+31
00090 ;
0000 00100 *GET SVCMAC:3 ;SVC Macro equivalents
00110 ;SVCMAC/ASM - LS-DOS Version VI
00120 *LIST OFF
04000 *LIST ON
04020 *LIST OFF ;Get LDOS60/EQU
05330 *LIST ON
05340 ;
2400 05350 ORG 2400H
05360 ;
05370 ; Save stack & call Spool routine
05380 ;
2400 ED730B24 05390 SPOOL LD (SAVE$P+1),SP ;Save SP
2404 CD5924 05400 CALL SPOOL1 ;Call SPOOL Code
2407 210000 05410 EXIT LD HL,0 ;Successful exit
05420 ;
05430 ; P/u stack & clear any pending <BREAK>
05440 ;
240A 310000 05450 SAVESP LD SP,$-$ ;P/u original stack
240D 05460 @CKBRKC ;Clear any <BREAK>
240D 3E6A 00001 LD A,106
240F EF 00002 RST 40
2410 C9 05470 RET ;Ret to DOS
05480 ;
05490 ; Informative Message Exit
05500 ;
2411 217D29 05510 CLEAR LD HL,CLEAR$
2414 DD 05520 DB 0DDH
2415 21A829 05530 RESUMD LD HL,RESUMD$
2418 DD 05540 DB 0DDH
2419 219229 05550 PAUSED LD HL,PAUSED$
241C DD 05560 DB 0DDH
241D 213A2A 05570 SPLACT LD HL,SPLACT$
2420 DD 05580 DB 0DDH
2421 21D529 05590 SPLONM LD HL,SPLONM$
2424 DD 05600 DB 0DDH
2425 21BF29 05610 SPLOFM LD HL,SPLOFM$
2428 DD 05620 DB 0DDH
2429 21EF29 05630 SPLOPM LD HL,SPLOPM$
242C DD 05640 DB 0DDH
242D 21112A 05650 PRMBGM LD HL,PRMBGM$
2430 05660 @LOGOT
00003 IFEQ 00H,1
00004 LD HL,
00005 ENDIF
2430 3E0C 00006 LD A,12
2432 EF 00007 RST 40
2433 18D2 05670 JR EXIT
05680 ;

```

```

05690 ;      I/O Error Processor
05700 ;
2435 3E2C 05710 PRMERR LD      A,PAR_ERR      ;Parameter Error
2437 6F 05720 IOERR LD      L,A              ;Set HL = Error #
2438 2600 05730 LD      H,0
243A F6C0 05740 OR      0C0H              ;Set short, return
243C 4F 05750 LD      C,A              ;Error code to C
243D 05760 @@ERROR              ;Dsply error
243D 3E1A 00008 LD      A,26
243F EF 00009 RST      40
2440 18C8 05770 JR      SAVESP              ;Exit
05780 ;
05790 ;      Internal Error Message Handler
05800 ;
2442 21C72A 05810 BADDCB LD      HL,BADDCB$
2445 DD 05820 DB      0DDH
2446 216029 05830 NOFIND LD      HL,NOFIND$
2449 DD 05840 DB      0DDH
244A 21972A 05850 INUSE LD      HL,INUSE$
244D DD 05860 DB      0DDH
244E 21AD2A 05870 CANT LD      HL,CANT$
2451 05880 @@LOGOT
00010 IFEQ 00H,1
00011 LD      HL,
00012 ENDIF
2451 3E0C 00013 LD      A,12
2453 EF 00014 RST      40
2454 21FFFF 05890 ERREXIT LD      HL,-1      ;Set abort code
2457 18B1 05900 JR      SAVESP              ;Exit
05910 ;
05920 ;      SPOOL1 - Set up system to Spool Output
05930 ;
05940 SPOOL1
2459 E5 05950 PUSH     HL              ;Save INBUF$ pointer
245A 05960 @@FLAGS              ;Get flag table pointer
245A 3E65 00015 LD      A,101
245C EF 00016 RST      40
245D DD211D2D 05970 LD      IX,PRMSAV      ;Point to parm save area
2461 11142D 05980 LD      DE,SPOOL$      ;If already resident,
2464 05990 @@GTMOD              ; don't need to check
2464 3E53 00017 LD      A,83
2466 EF 00018 RST      40
2467 2032 06000 JR      NZ,SPL1      ; if memory available
2469 ED53A428 06010 LD      (SVD CB),DE      ;Save DCB pointer
246D 22B228 06020 LD      (SVBGN),HL      ;Save module begin
2470 210400 06030 LD      HL,4      ;Get parm save pointer
2473 19 06040 ADD      HL,DE
2474 22DD28 06050 LD      (HIPARM),HL      ;Save for use by clear
2477 DDE5 06060 PUSH     IX      ;Also put address in DE
2479 D1 06070 POP      DE      ;Shift saved parms into
247A 010B00 06080 LD      BC,11      ; spool driver front end
247D EDB0 06090 LDIR
247F DD5E04 06100 LD      E,(IX+4)      ;P/u previous MPARM
2482 DD5605 06110 LD      D,(IX+5)
2485 ED538D26 06120 LD      (MPARM+1),DE      ;Stuff as new default
2489 DD5E06 06130 LD      E,(IX+6)      ;P/u previous DPARM
248C DD5607 06140 LD      D,(IX+7)
248F ED531B26 06150 LD      (DPARM+1),DE      ;Stuff as new default
2493 DD7E0A 06160 LD      A,(IX+10)      ;P/u old bank #
2496 E67F 06170 AND      07FH      ;Strip hi bit
2498 32B625 06180 LD      (BPARM+1),A      ; & save as default

```

```

249B E1      06190 SPL1  POP    HL          ;Get INBUF$ pointer
              06200 ;
              06210 ;      Fetch device spec to spool
              06220 ;

249C 11E22A  06230 SPL2  LD      DE,SPLDCB      ;Fetch device spec
249F         06240      @@FSPEC      ; to spool
249F 3E4E    00019      LD      A,78
24A1 EF      00020      RST     40
24A2 E5      06250      PUSH   HL          ;Save inbuf$ ptr
24A3 201E    06260      JR      NZ,USEPR    ;Use *PR if no entry
24A5 1A      06270      LD      A,(DE)      ;If a device spec not
24A6 FE2A    06280      CP      '*'          ; found, maybe it was
24A8 2824    06290      JR      Z,SPL2A      ; the spool file spec
              06300 ;
              06310 ;      File spec fetched - Move to spool FCB
              06320 ;

24AA EB      06330      EX      DE,HL          ;DCB to HL
24AB 11332F  06340      LD      DE,SPLFCB      ;Point to spool FCB
24AE D5      06350      PUSH   DE
24AF 012000  06360      LD      BC,32
24B2 EDB0    06370      LDIR                     ;Move it in
24B4 21092B  06380      LD      HL,PR$        ;Move in default spec
24B7 11E22A  06390      LD      DE,SPLDCB      ; for spool device
24BA 010400  06400      LD      BC,4
24BD EDB0    06410      LDIR
24BF AF      06420      XOR     A
24C0 D1      06430      POP     DE          ;Point to SPLFCB
24C1 181B    06440      JR      SPL2B          ;Go - we have spec
24C3 21092B  06450 USEPR  LD      HL,PR$        ;Move in default spec
24C6 11E22A  06460      LD      DE,SPLDCB
24C9 010400  06470      LD      BC,4
24CC EDB0    06480      LDIR
              06490 ;
              06500 ;      Fetch file spec to use as disk spool buffer
              06510 ;

24CE E1      06520 SPL2A  POP     HL          ;Get inbuf$ ptr
24CF ED5BE32A 06530      LD      DE,(SPLDCB+1) ;Stuff device name for
24D3 ED53022B 06540      LD      (DSKSPC),DE    ; default spool file name
24D7 11332F  06550      LD      DE,SPLFCB      ;Fetch spool filespec
24DA         06560      @@FSPEC
24DA 3E4E    00021      LD      A,78
24DC EF      00022      RST     40
24DD E5      06570      PUSH   HL          ;Save separator char
24DE F5      06580 SPL2B  PUSH   AF          ; & line pointer
24DF 21022B  06590      LD      HL,DSKSPC      ;Default to XX/SPL
24E2 2803    06600      JR      Z,SPL2C
24E4         06610      @@FSPEC      ; if user entered none
24E4 3E4E    00023      LD      A,78
24E6 EF      00024      RST     40
24E7 21052B  06620 SPL2C  LD      HL,SPL2C      ;Default ext to SPL
24EA         06630      @@FEXT
24EA 3E4F    00025      LD      A,79
24EC EF      00026      RST     40
24ED F1      06640      POP     AF          ;Rcvr line pointer
24EE E1      06650      POP     HL
24EF FE3A    06660      CP      ':'          ;Drive entered?
24F1 201C    06670      JR      NZ,GETPRM
24F3 1A      06680 SPL3  LD      A,(DE)      ;Wait for ETX or EOL
24F4 FE0D    06690      CP      CR
24F6 2807    06700      JR      Z,SPL4
24F8 FE03    06710      CP      3

```

```

24FA 2803 06720 JR Z,SPL4
24FC 13 06730 INC DE
24FD 18F4 06740 JR SPL3
24FF 3E3A 06750 SPL4 LD A,':' ;Stuff colon for drive
2501 12 06760 LD (DE),A
2502 13 06770 INC DE
2503 7E 06780 LD A,(HL) ;P/u possible drive #
2504 FE0D 06790 CP CR
2506 CA3524 06800 JP Z,PRMERR ;Parm error if EOL
2509 23 06810 INC HL
250A 12 06820 LD (DE),A ;Stuff drive #
250B 13 06830 INC DE
250C 3E03 06840 LD A,3
250E 12 06850 LD (DE),A ;Stuff ETX
06860 ;
06870 ;
06880 ; Get parameter entries
250F 110D2B 06890 GETPRM LD DE,PRMTBL$ ;Get parms
2512 06900 @PARAM
2512 3E11 00027 LD A,17
2514 EF 00028 RST 40
2515 C23524 06910 JP NZ,PRMERR ;Jump on parm error
06920 ;
06930 ; Check on OFF
06940 ;
2518 110000 06950 OPARM LD DE,0 ;P/u off parm
251B 1C 06960 INC E ;On or off?
251C CA3128 06970 JP Z,SPLOFF ;Jump if OFF
06980 ;
06990 ; Check on RESUME despooling
07000 ;
251F 110000 07010 RPARAM LD DE,0 ;P/u Resume parm
2522 1C 07020 INC E ;Was it used?
2523 CA1C28 07030 JP Z,RESUME ;Go if so
07040 ;
07050 ; Check on PAUSE
07060 ;
2526 110000 07070 PPARAM LD DE,0 ;P/u Pause parm
2529 1C 07080 INC E ;Was it used?
252A CA0728 07090 JP Z,PAUSE ;Go if so
07100 ;
07110 ; Check on CLEAR
07120 ;
252D 110000 07130 CPARM LD DE,0 ;P/u Clear parm
2530 1C 07140 INC E ;Was is used?
2531 CAD428 07150 JP Z,CLEAR ;Go if so
07160 ;
07170 ; Assume request to be to install
07180 ;
2534 FDCB0246 07190 BIT 0,(IY+'C'-'A') ;Can't insert spool if
2538 C24E24 07200 JP NZ,CANT ; memory frozen
253B FDCB0346 07210 BIT 0,(IY+'D'-'A') ;Is spooler already on?
253F C22124 07220 JP NZ,SPLONM ;Quit if so
07230 ;
07240 ; If module resident, ck parm sizes
07250 ;
2542 11142D 07260 LD DE,SPOOL$
2545 07270 @GTMOD
2545 3E53 00029 LD A,83
2547 EF 00030 RST 40
2548 202B 07280 JR NZ,FNDDCB ;Go if no area in mem

```


The Source	LIBRARY Files	SPPOOL - LS-DOS 6.2	Page 00005
254A DD6E04	07290	LD L,(IX+4)	;Prev. MPARM
254D DD6605	07300	LD H,(IX+5)	
2550 ED5B8D26	07310	LD DE,(MPARM+1)	
2554 B7	07320	OR A	
2555 ED52	07330	SBC HL,DE	;Old must = new
2557 C22D24	07340	JP NZ,PRMBGM	;Abort if bad
255A DD6E06	07350	LD L,(IX+6)	;D parm
255D DD6607	07360	LD H,(IX+7)	
2560 ED5B1B26	07370	LD DE,(DPARM+1)	
2564 ED52	07380	SBC HL,DE	
2566 C22D24	07390	JP NZ,PRMBGM	;Abort if not same as old
2569 DD7E0A	07400	LD A,(IX+10)	;Bank #
256C E67F	07410	AND 07FH	;Strip hi bit
256E 2AB625	07420	LD HL,(BPARM+1)	
2571 BD	07430	CP L	
2572 C22D24	07440	JP NZ,PRMBGM	;Abort if not same as old
	07450 ;		
	07460 ;	Find device to spool	
	07470 ;		
2575 ED5BE32A	07480 FNDDCB	LD DE,(SPLDCB+1)	;P/u device name
2579	07490	@@GTDCB	;Find in device tables
2579 3E52	00031	LD A,82	
257B EF	00032	RST 40	
257C C23724	07500	JP NZ,IOERR	;Jump on not found
257F CB66	07510 CKRTE	BIT 4,(HL)	;Routed device?
2581 2807	07520	JR Z,CKLNK	;Bypass if not
2583 2C	07530 CKRTE1	INC L	;Pt to vector
2584 7E	07540	LD A,(HL)	;P/u vector DCB address
2585 2C	07550	INC L	; & test it
2586 66	07560	LD H,(HL)	
2587 6F	07570	LD L,A	
2588 18F5	07580	JR CKRTE	
258A CB6E	07590 CKLNK	BIT 5,(HL)	;If linked, get DCB
258C 20F5	07600	JR NZ,CKRTE1	
258E CB76	07610	BIT 6,(HL)	;If filtered, trace to
2590 281B	07620	JR Z,SETDVR	; its DCB in header
2592 54	07630	LD D,H	;Save to ck inactive
2593 5D	07640	LD E,L	
2594 2C	07650	INC L	
2595 7E	07660	LD A,(HL)	;Get vector to module
2596 2C	07670	INC L	
2597 66	07680	LD H,(HL)	
2598 6F	07690	LD L,A	
2599 010400	07700	LD BC,4	;HL now points to the
259C 09	07710	ADD HL,BC	; entry point. Get its
259D 4E	07720	LD C,(HL)	; DCB address by peeking
259E 0C	07730	INC C	; past the name field
259F 09	07740	ADD HL,BC	
25A0 7E	07750	LD A,(HL)	;Get low-order
25A1 23	07760	INC HL	
25A2 66	07770	LD H,(HL)	;Get hi-order
25A3 6F	07780	LD L,A	
25A4 E5	07790	PUSH HL	;If DCB is itself, then
25A5 ED52	07800	SBC HL,DE	; it's some error
25A7 E1	07810	POP HL	; else continue to
25A8 20D5	07820	JR NZ,CKRTE	; search the chain
25AA C34224	07830	JP BADDCB	
	07840 ;		
	07850 ;	Found the device	
	07860 ;		
25AD CB4E	07870 SETDVR	BIT 1,(HL)	;Device handle @PUT?

```

25AF CA4224 07880 JP Z,BADDCB ;Can't do if not
25B2 22252D 07890 LD (MODDVR),HL ;Stuff DCB table address
07900 ;
07910 ; Check BANK spec for availability
07920 ;
25B5 010000 07930 BPARM LD BC,0 ;Pick up the bank #
25B8 79 07940 LD A,C ;If bank=0, no need
25B9 B0 07950 OR B ; to test for
25BA 281B 07960 JR Z,LOCDCB ; availability
25BC FE08 07970 CP 8 ;This version supports
25BE D23524 07980 JP NC,PRMERR ; banks 0-7
25C1 CBFF 07990 SET 7,A ;Set for transfer
25C3 32612D 08000 LD (BANKX1),A ;Stuff for module A
25C6 32832D 08010 LD (BANKX2),A
25C9 32902D 08020 LD (BANKX3),A
25CC 32272D 08030 LD (SPLBNK),A
25CF 0602 08040 LD B,2 ;Test if bank 1-x
25D1 08050 @BANK ; is available
25D1 3E66 08033 LD A,102
25D3 EF 08034 RST 40
25D4 C24A24 08060 JP NZ,INUSE ;Quit if not available
08070 ;
08080 ; Locate a spare DCB for the spool module
08090 ;
25D7 110000 08100 LOCDCB LD DE,0
25DA 08110 @GTDCB
25DA 3E52 08035 LD A,82
25DC EF 08036 RST 40
25DD 3E21 08120 LD A,33 ;Init "No device space..."
25DF C23724 08130 JP NZ,IOERR ;Go if no spare DCBs
25E2 22192D 08140 LD (S0DCB),HL ;Save pointer
25E5 22B62D 08150 LD (S0DCB1),HL
25E8 E5 08160 PUSH HL ;Save DCB field pointer
25E9 11532F 08170 LD DE,'/S' ;Let's find a link name
25EC 14 08180 NAMLP INC D ;Bump "2nd" character
25ED 08190 @GTDCB ;If we find this name
25ED 3E52 08037 LD A,82
25EF EF 08038 RST 40
25F0 28FA 08200 JR Z,NAMLP ; look for another
25F2 E1 08210 POP HL ;Get DCB pointer
25F3 CBDE 08220 SET 3,(HL) ;NIL in case of error
25F5 010600 08230 LD BC,6
25F8 09 08240 ADD HL,BC ;Point to name field
25F9 73 08250 LD (HL),E ; & stuff in the
25FA 2C 08260 INC L ; selected spool name
25FB 72 08270 LD (HL),D
08280 ;
08290 ; Create a disk file for spooling
08300 ;
25FC ED5B1B26 08310 LD DE,(DPARM+1) ;P/u disk size
2600 DD7306 08320 LD (IX+6),E ;Save DPARM for next time
2603 DD7207 08330 LD (IX+7),D ; SPOOL is entered
2606 7A 08340 LD A,D
2607 B3 08350 OR E
2608 284E 08360 JR Z,PUHIGH ;No file if DISK=0
260A 11332F 08370 LD DE,SPLFCB ;Init the spool file
260D 21002C 08380 LD HL,BUFFER
2610 FDCB12C6 08390 SET 0,(IY+'S'-'A') ;Init file open bit
2614 08400 @INIT
2614 3E3A 08039 LD A,58
2616 EF 08040 RST 40

```

```

2617 C23724 08410 JP NZ,IOERR ;Jump on init error
261A 010400 08420 DPARM LD BC,4 ;D parm (default=4K)
261D 78 08430 LD A,B ;Parm error if > 4095K
261E E6F0 08440 AND 0F0H
2620 C23524 08450 PRMERRA JP NZ,PRMERR ;Go if too big a file
2623 CB21 08460 SLA C ;Multiply K by 4
2625 CB10 08470 RL B ; to calculate sectors
2627 CB21 08480 SLA C
2629 CB10 08490 RL B
262B ED43FF26 08500 LD (IPLDSK1+1),BC ;Stuff disk K
262F 0B 08510 DEC BC ;Adjust for 0 offset
2630 08520 @@POSN ;Position to end
2630 3E42 00041 LD A,66
2632 EF 00042 RST 40
2633 08530 @@WRITE ;Write a junk sector
2633 3E4B 00043 LD A,75
2635 EF 00044 RST 40
2636 C23724 08540 JP NZ,IOERR ;Jump on write error
2639 08550 @@REW ;Restore file to 0
2639 3E44 00045 LD A,68
263B EF 00046 RST 40
263C 08560 @@WEOF ;End of file mark
263C 3E4A 00047 LD A,74
263E EF 00048 RST 40
263F ED4B392F 08570 LD BC,(SPLFCB+6) ;P/u DEC & drive
2643 08580 @@DIRRD ;Read its dir record
2643 3E57 00049 LD A,87
2645 EF 00050 RST 40
2646 C23724 08590 JP NZ,IOERR ;Quit on read error
2649 23 08600 INC HL ;Point to DIR+1
264A CBFE 08610 SET 7,(HL) ;Turn on CREATE bit
264C 08620 @@DIRWR ;Write dir back
264C 3E58 00051 LD A,88
264E EF 00052 RST 40
264F C23724 08630 JP NZ,IOERR ;Jump on error
2652 210023 08640 LD HL,LOWBUF$ ;Revise I/O buffer
2655 22362F 08650 LD (SPLFCB+3),HL ; in file's FCB
08660 ;
08670 ; Get current HIGH$
08680 ;
2658 210000 08690 PUHIGH LD HL,0 ;Set HLB to zero to
265B 45 08700 LD B,L ; fetch HIGH$
265C 08710 @@HIGH$
265C 3E64 00053 LD A,100
265E EF 00054 RST 40
265F 22BC26 08720 LD (GBUF1+1),HL ;Save for later
2662 22BD27 08730 LD (OLDHI+1),HL
08740 ;
08750 ; If bank RAM, recalculate MPARM
08760 ;
2665 3AB625 08770 LD A,(BPARM+1) ;Alternate banks requested?
2668 B7 08780 OR A
2669 2821 08790 JR Z,MPARM ;Go if not
08800 ;
08810 ; Recalculate MEM parameter to use the
08820 ; Maximum space in bank RAM. Formula is
08830 ; [((32768 - 512 for module) - ((DPARM+1)*16))]
08840 ; Divided by 260. The 260 is derived from
08850 ; 256-byte page plus 4-bytes for pointers.
08860 ; Then revise to K from pages.
08870 ;

```

The Source	LIBRARY Files	SPPOOL - LS-DOS 6.2	Page 00008
266B 2A1B26	08880	LD HL,(DPARM+1)	
266E 23	08890	INC HL	;Bump by 1
266F 29	08900	ADD HL,HL	;Times 16
2670 29	08910	ADD HL,HL	
2671 29	08920	ADD HL,HL	
2672 29	08930	ADD HL,HL	
2673 4D	08940	LD C,L	;Xfer to BC for subtract
2674 44	08950	LD B,H	
2675 21007E	08960	LD HL,32768-512	;RAM space - module space
2678 AF	08970	XOR A	;The remainder is for
2679 ED42	08980	SBC HL,BC	; MEM pages and ptrs
267B CB3C	08990	SRL H	;A divide by 260 is
267D CB1D	09000	RR L	; accomplished by
267F 0E82	09010	LD C,260/2	; div 2 and div 130
2681	09020	@@DIV16	
2681 3E5E	00055	LD A,94	
2683 EF	00056	RST 40	
2684 4D	09030	LD C,L	
2685 44	09040	LD B,H	
2686 CB39	09050	SRL C	;Calculate
2688 CB39	09060	SRL C	; K = pages / 4
268A 1815	09070	JR CALCPTR	
	09080 ;		
	09090 ;		
	09100 ;		
	09110 ;		
268C 010100	09110 MPARM	LD BC,1	;MEM parm - default 1K
268F 78	09120	LD A,B	;Test entry - must be
2690 B7	09130	OR A	< 33K
2691 208D	09140	JR NZ,PRMERRA	;Hi-order must = 0
2693 B9	09150	CP C	;MEM = 0 ?
2694 CA3524	09160	JP Z,PRMERR	;Yes - Parameter Error
2697 7C	09170	LD A,H	;Get hi-order HIGH\$
2698 D681	09180	SUB 81H	;Can't go below 8000H
269A 1F	09190	RRA	;Divide by 2
269B CB3F	09200	SRL A	; & again to calc K
269D 91	09210	SUB C	;Reduce by K of req
269E DA3524	09220	JP C,PRMERR	;Error if goes < 8000H
26A1 DD7104	09230 CALCPTR	LD (IX+4),C	;Stuff MEM parm into
26A4 DD7005	09240	LD (IX+5),B	; save area
	09250 ;		
	09260 ;		
	09270 ;		
26A7 2A1B26	09280	LD HL,(DPARM+1)	;P/u disk K
26AA 09	09290	ADD HL,BC	;Calculate pointer
26AB 29	09300	ADD HL,HL	; buffer space required
26AC 29	09310	ADD HL,HL	; (D+M)*16 + 4 extra
26AD 23	09320	INC HL	; used to detect the end
26AE 29	09330	ADD HL,HL	
26AF 29	09340	ADD HL,HL	
	09350 ;		
	09360 ;		
	09370 ;		
	09380 ;		
26B0 E5	09380	PUSH HL	
26B1 DD6E00	09390	LD L,(IX+0)	;P/u saved HIGH\$ entry
26B4 DD6601	09400	LD H,(IX+1)	
26B7 7D	09410	LD A,L	;If <> 0, then SPOOL
26B8 B4	09420	OR H	; is resident
26B9 2009	09430	JR NZ,GBUF2	
26BB 210000	09440 GBUF1	LD HL,\$-\$;P/u current high
26BE DD7500	09450	LD (IX+0),L	;Save the value
26C1 DD7401	09460	LD (IX+1),H	

```

26C4 223629 009470 GBUF2 LD (SVEND+1),HL ;Stuff highest byte used
009480 ;
009490 ; If bank RAM used, reset to X'FFFF'
009500 ;

26C7 23 009510 INC HL ;Pt to lower 256
26C8 2E00 009520 LD L,0 ; byte block boundary
26CA 3AB625 009530 LD A,(BPARAM+1) ;Bank specified?
26CD B7 009540 OR A
26CE 2801 009550 JR Z,$+3 ;Jump if not
26D0 65 009560 LD H,L ;Set to FFFF+1 = 0
26D1 4F 009570 LD C,A ;Reside bank for
26D2 0600 009580 LD B,0 ; module B
26D4 009590 @BANK
26D4 3E66 000057 LD A,102
26D6 EF 000058 RST 40
26D7 C23724 009600 JP NZ,IOERR ;Quit on bank error
26DA DD7E04 009610 LD A,(IX+4) ;P/u MEM parm
26DD 07 009620 RLCA ;MEM * 4 = # of
26DE 07 009630 RLCA ; 256-byte blocks
26DF 47 009640 LD B,A ;Go back that number
26E0 ED44 009650 NEG ; of blocks needed
26E2 84 009660 ADD A,H ;HL now reserves buffer
26E3 67 009670 LD H,A ; blocks for MEM
26E4 22532F 009680 LD (PUTNXT$),HL ;Stuff pointers to begin
26E7 22552F 009690 LD (GETNXT$),HL ; of memory blocks
26EA 4C 009700 LD C,H
26EB D1 009710 POP DE ;Get (D+M)*16+4
26EC AF 009720 XOR A ;Backup the # of D+M
26ED ED52 009730 SBC HL,DE ; sectors
009740 ;
009750 ; Initialize the memory pointers
009760 ;

26EF E5 009770 PUSH HL ;Save pointer
26F0 3600 009780 IPLMEM LD (HL),0 ;Init the 4-byte fields
26F2 23 009790 INC HL ; for each of the disk &
26F3 3600 009800 LD (HL),0 ; memory block pointers
26F5 23 009810 INC HL
26F6 71 009820 LD (HL),C ;Stuff memory hi-order
26F7 23 009830 INC HL
26F8 3600 009840 LD (HL),0
26FA 23 009850 INC HL
26FB 0C 009860 INC C ;Bump memory hi-order
26FC 10F2 009870 DJNZ IPLMEM
009880 ;
009890 ; Initialize the disk pointers
009900 ;

26FE 110000 009910 IPLDSK1 LD DE,0 ;P/u # of disk sectors
2701 7A 009920 LD A,D
2702 B3 009930 OR E
2703 2815 009940 JR Z,IPLDSK3 ;Bypass if none
2705 010000 009950 LD BC,0 ;Set up a disk pointer
2708 3600 009960 IPLDSK2 LD (HL),0 ; for each disk block
270A 23 009970 INC HL
270B 3600 009980 LD (HL),0
270D 23 009990 INC HL
270E 71 100000 LD (HL),C ;Stuff block number
270F 23 100010 INC HL
2710 70 100020 LD (HL),B
2711 CBFE 100030 SET 7,(HL) ;Set bit 7 to indicate
2713 23 100040 INC HL ; this pntr is for disk
2714 03 100050 INC BC ;Inc block number

```

```

2715 1B      10060      DEC      DE
2716 7A      10070      LD      A,D          ;Test if finished
2717 B3      10080      OR      E
2718 20EE    10090      JR      NZ,IPLDSK2    ;Loop if not
271A 3600    10100 IPLDSK3 LD      (HL),0      ;One more used to
271C 23      10110      INC     HL          ; indicate the end
271D 3600    10120      LD      (HL),0      ; of the pointer fields
271F 23      10130      INC     HL
2720 36FF    10140      LD      (HL),0FFH
2722 23      10150      INC     HL
2723 36FF    10160      LD      (HL),0FFH
2725 E1      10170      POP     HL          ;Rcvr start of pointers
2726 E5      10180      PUSH    HL
2727 22572F  10190      LD      (MAPPTR$),HL ; & stuff for later use
272A 22592F  10200      LD      (PUTBLK$),HL
272D 225B2F  10210      LD      (GETBLK$),HL
2730 7C      10220      LD      A,H          ;Stuff 1st MEM block ptr
2731 75      10230      LD      (HL),L      ; with table addr
2732 23      10240      INC     HL
2733 77      10250      LD      (HL),A
          10260 ;
          10270 ;
          10280 ;      Modify driver for spooling device vector address
2734 FD7E1F  10290      LD      A,(IY+KITSK0) ;Transfer device hook
2737 FD6E20  10300      LD      L,(IY+KITSK0+1)
273A FD6621  10310      LD      H,(IY+KITSK0+2)
273D 32982D  10320      LD      (GETBLK0),A
2740 22992D  10330      LD      (GETBLK0+1),HL
2743 FDE5    10340      PUSH    IY          ;Set up address
2745 D1      10350      POP     DE          ; in despooler
2746 210F00  10360      LD      HL,'P'-'A'    ;PFLAG use to
2749 19      10370      ADD     HL,DE        ; determine if spooler
274A 226D2D  10380      LD      (DFLAG),HL    ; is Paused
274D 226F2E  10390      LD      (DFLAG1),HL
2750 C1      10400      POP     BC          ;Start of MEM block tbl
2751 0B      10410      DEC     BC          ;Get last byte to use
2752 C5      10420      PUSH    BC
2753 115E2F  10430      LD      DE,DVRBEND    ;Point to module B
2756 215F2F  10440      LD      HL,RELTABB    ;Point to table
2759 CD3029  10450      CALL    RELO        ;Relocate absolute refs
          10460 ;
          10470 ;
          10480 ;      If bank 0, then wipe out module A
275C 3AB625  10490      LD      A,(BPARAM+1) ;Alternate bank used?
275F B7      10500      OR      A
2760 2019    10510      JR      NZ,MOVMOB    ;Skip next if so
2762 21002D  10520      LD      HL,FIX1      ;Shift in JPs for the
2765 115C2D  10530      LD      DE,PUTCHAR    ; bank transfer
2768 010500  10540      LD      BC,5         ; linkage
276B EDB0    10550      LDIR
276D 117F2D  10560      LD      DE,FIX2UP
2770 0E05    10570      LD      C,5
2772 EDB0    10580      LDIR
2774 118C2D  10590      LD      DE,GETBLK
2777 0E05    10600      LD      C,5
2779 EDB0    10610      LDIR
277B D1      10620 MOVMOB POP     DE          ;Get last byte to use
277C 215E2F  10630      LD      HL,DVRBEND    ;Last byte of module
277F 01A901  10640      LD      BC,DVRBEND-DVRBBGN+1
2782 EDB8    10650      LDDR
2784 D5      10660      PUSH    DE          ;New last byte

```

```

10670 ;
10680 ;      Switch to bank 0 always
10690 ;
2785 010000 10700 LD      BC,0      ;Set function 0, bank 0
2788      10710 @BANK
2788 3E66    00059 LD      A,102
278A EF     00060 RST      40
278B C23724 10720 JP      NZ,IOERR    ;Quit on bank error
278E C1      10730 POP      BC
      10740 ;
      10750 ;      If module B is in bank x, switch HIGH for A
      10760 ;
278F 3A8625 10770 LD      A,(BPARM+1) ;P/u alternate bank resp.
2792 B7      10780 OR      A
2793 280D    10790 JR      Z,STORHI    ;Go if none used
2795 4F      10800 LD      C,A      ;Reserve the bank
2796 0603    10810 LD      B,3      ;Set function # 3
2798      10820 @BANK
2798 3E66    00061 LD      A,102
279A EF     00062 RST      40
279B C23724 10830 JP      NZ,IOERR    ;Quit on bank error
279E ED4B3629 10840 LD      BC,(SVEND+1) ;P/u bank-A HIGH$
27A2 C5      10850 STORHI PUSH     BC
      10860 ;
      10870 ;      Relocate absolute addresses in module A
      10880 ;
27A3 119A2D 10890 LD      DE,DVRAEND    ;Point to last used in A
27A6 219B2D 10900 LD      HL,RELTABA ;Point to relo table
27A9 CD3029 10910 CALL     RELO      ;Relocate absolute refs
27AC D1      10920 POP      DE      ;Get last byte to use
27AD 219A2D 10930 LD      HL,DVRAEND ;Last byte of module
27B0 018C00 10940 LD      BC,DVRAEND-DVRABGN+1
27B3 EDB8    10950 LDDR
27B5 211100 10960 LD      HL,PRMSAV+2-DVRABGN+1
27B8 19      10970 ADD      HL,DE      ;Data area in himem
27B9 73      10980 LD      (HL),E
27BA 23      10990 INC      HL
27BB 72      11000 LD      (HL),D      ;Save high$ there
      11010 ;
      11020 ;      Stuff new HIGH$
      11030 ;
27BC 210000 11040 OLDHI LD      HL,$-$    ;P/u old HIGH$
27BF ED52    11050 SBC      HL,DE      ;Check if old HIGH$ is
27C1 3805    11060 JR      C,NOSET    ; already lower
27C3 EB      11070 EX      DE,HL      ;New HIGH$ to HL
27C4      11080 @HIGH$ ;Stuff new HIGH$
27C4 3E64    00063 LD      A,100
27C6 EF     00064 RST      40
27C7 EB      11090 EX      DE,HL
27C8 13      11100 NOSET INC      DE      ;Point to module start
27C9 2A192D 11110 LD      HL,(S0DCB) ;P/u module DCB pointer
27CC 3646    11120 LD      (HL),46H ; & set to filter
27CE 23      11130 INC      HL      ;Pt to DCB vector pos
27CF 73      11140 LD      (HL),E    ;Put entry point of
27D0 23      11150 INC      HL      ; module into DCB
27D1 72      11160 LD      (HL),D
27D2 116A2D 11170 LD      DE,DSPLTCB ;Calculate entry point
27D3      11180 RX05 EQU      $-2
27D5 0E09    11190 LD      C,9      ;This will be task 9
27D7      11200 @ADTSK
27D7 3E1D    00065 LD      A,29

```

The Source	LIBRARY Files	SPOOL - LS-DOS 6.2	Page 00012
27D9 EF	00066	RST 40	
27DA 218C2D	11210	LD HL,GETBLK	;Now update device hook
27DB	11220 RX06	EQU \$-2	
	11230 ;		
	11240 ;		
	11250 ;	Interface to KITSK - No need to DI/EI	
	11260 ;	As KITSK not used by interrupts	
27DD FD7421	11270	LD (IY+KITSK@+2),H	;Move in device
27E0 FD7520	11280	LD (IY+KITSK@+1),L	; vector and a
27E3 3EC3	11290	LD A,0C3H	; JP opcode
27E5 FD771F	11300	LD (IY+KITSK@),A	
27E8 FDCB03C6	11310	SET 0,(IY+'D'-'A')	;Turn on device flag bit
27EC FDCB0FFE	11320	SET 7,(IY+'P'-'A')	;Show not paused
	11330 ;		
	11340 ;		
	11350 ;	Filter the specified device driver	
27F0 2A252D	11360	LD HL,(MODDVR)	;Ptr to device DCB
27F3 ED5B192D	11370	LD DE,(S0DCB)	;Ptr to spooler DCB
27F7 0603	11380	LD B,3	
27F9 F3	11390	DI	;Interrupts off during swap
27FA 4E	11400 SWAP	LD C,(HL)	;Swap first 3 bytes
27FB 1A	11410	LD A,(DE)	
27FC 77	11420	LD (HL),A	; of the DCBs
27FD 79	11430	LD A,C	
27FE 12	11440	LD (DE),A	
27FF 2C	11450	INC L	;Bump DCB pointers
2800 1C	11460	INC E	
2801 10F7	11470	DJNZ SWAP	;Loop 3 times
2803 FB	11480	EI	
2804 C31D24	11490	JP SPLACT	;Set complete
	11500 ;		
	11510 ;		
	11520 ;	Routine to PAUSE despooling	
2807 FDCB0346	11530 PAUSE	BIT 0,(IY+'D'-'A')	;Spool resident?
280B CA2524	11540	JP Z,SPL0FM	;Quit if not
280E FDCB0F7E	11550	BIT 7,(IY+'P'-'A')	;Ck first if active
2812 FDCB0FBE	11560	RES 7,(IY+'P'-'A')	; then turn off
2816 CA2924	11570	JP Z,SPL0PM	;Quit if already paused
2819 C31924	11580	JP PAUSED	
	11590 ;		
	11600 ;		
	11610 ;	Routine to RESUME despooling	
281C FDCB0346	11620 RESUME	BIT 0,(IY+'D'-'A')	;Ck first if resident
2820 CA2524	11630	JP Z,SPL0FM	;Quit if not
2823 FDCB0F7E	11640	BIT 7,(IY+'P'-'A')	;See if despooling now
2827 FDCB0FFE	11650	SET 7,(IY+'P'-'A')	
282B C22124	11660	JP NZ,SPL0NM	;Can't resume if so
282E C31524	11670	JP RESUMD	
	11680 ;		
	11690 ;		
	11700 ;	Routine to turn off the spooler	
2831 FDCB0346	11710 SPLOFF	BIT 0,(IY+'D'-'A')	;Test if off already
2835 CA2524	11720	JP Z,SPL0FM	;Go if off now
2838 11142D	11730	LD DE,SPOOL\$;Find out where spooler
283B	11740	@GTMOD	
283B 3E53	00067	LD A,83	
283D EF	00068	RST 40	
283E C24624	11750	JP NZ,NOFIND	;Quit if not there
2841 FDCB0386	11760	RES 0,(IY+'D'-'A')	; else turn off
2845 FDCB0FBE	11770	RES 7,(IY+'P'-'A')	;Show paused also
2849 018900	11780	LD BC,GETBLK0-DVRABGN	


```

284C 09      11790      ADD    HL,BC          ;Point to KITSK@ swap
284D 7E      11800      LD      A,(HL)
284E FD771F  11810      LD      (IY+KITSK@),A    ;Restuff KIH00K
2851 23      11820      INC     HL
2852 7E      11830      LD      A,(HL)
2853 FD7520  11840      LD      (IY+KITSK@+1),L
2856 23      11850      INC     HL
2857 7E      11860      LD      A,(HL)
2858 FD7721  11870      LD      (IY+KITSK@+2),A
285B 0E09    11880      LD      C,9              ;Remove task 9 from
285D         11890      @@RMTSK                  ; the task processor
285D 3E1E    00069      LD      A,30
285F EF      00070      RST      40
                11900      ;
                11910      ;
                11920      ;
                Attempt to reclaim memory space
2860 210000  11930      LD      HL,0              ;Get current HIGH$
2863 45      11940      LD      B,L
2864         11950      @@HIGH$
2864 3E64    00071      LD      A,100
2866 EF      00072      RST      40
2867 DD5E02  11960      LD      E,(IX+2)          ;Get HIGH$ after spool
286A DD5603  11970      LD      D,(IX+3)          ; was installed
286D AF      11980      XOR      A
286E ED52    11990      SBC      HL,DE            ;Is it the same now?
2870 DD5E00  12000      LD      E,(IX+0)          ;Get HIGH before res
2873 DD7700  12010      LD      (IX+0),A          ; & clear data
2876 DD5601  12020      LD      D,(IX+1)
2879 DD7701  12030      LD      (IX+1),A
287C 21552A  12040      LD      HL,NOMEM$         ;Init "Can't reclaim...
287F 2007    12050      JR      NZ,LOGMEM         ;Go if HIGH$ different
2881 EB      12060      EX       DE,HL            ; else release memory
2882         12070      @@HIGH$                  ; by resetting HIGH$
2882 3E64    00073      LD      A,100
2884 EF      00074      RST      40
2885 21702A  12080      LD      HL,GOTMEM$        ;"Memory released
2888         12090      LOGMEM @@LOGOT
                00075      IFEQ    000H,1
                00076      LD      HL,
                00077      ENDIF
2888 3E0C    00078      LD      A,12
288A EF      00079      RST      40
                12100      ;
                12110      ;
                12120      ;
                De-use any RAM bank
288B DD7E0A  12130      LD      A,(IX+10)         ;P/u bank used
288E E607    12140      AND      7              ;Strip 3-7
2890 2811    12150      JR      Z,RESDCB          ;Jump if only bank 0
2892 4F      12160      LD      C,A              ;Save bank in C
2893 F630    12170      OR      '0'             ;Convert to ASCII for dsply
2895 328C2A  12180      LD      (FREBNK$+5),A    ;Stuff in message
2898 0601    12190      LD      B,1              ;Function to reset BUR$
289A         12200      @@BANK                    ;Free the bank
289A 3E66    00080      LD      A,102
289C EF      00081      RST      40
289D 21872A  12210      LD      HL,FREBNK$        ;Init"Bank released
28A0         12220      @@LOGOT
                00082      IFEQ    000H,1
                00083      LD      HL,
                00084      ENDIF
28A0 3E0C    00085      LD      A,12

```

```

28A2 EF      00086      RST      40
              12230 ;
              12240 ;      RESET the device
              12250 ;
28A3 210000    12260 RESDCB  LD      HL,$-$
28A4          12270 SVDCB   EQU      $-2      ;Point to SPOOL DCB ptr
28A6 5E        12280      LD      E,(HL)      ; & get the DCB ptr
28A7 23        12290      INC      HL          ; into reg DE
28A8 56        12300      LD      D,(HL)
              12310 ;
              12320 ;      Check if DCB is inactive [user RESET *PR]
              12330 ;
28A9 D5        12340      PUSH     DE          ;Save DCB pointer
28AA 1C        12350      INC      E
28AB 1A        12360      LD      A,(DE)      ;Get DCB vector contents
28AC 4F        12370      LD      C,A        ; into reg BC
28AD 1C        12380      INC      E
28AE 1A        12390      LD      A,(DE)
28AF 47        12400      LD      B,A
28B0 D1        12410      POP      DE          ;Recover DCB pointer
28B1 210000    12420      LD      HL,$-$      ;P/u start of module
28B2          12430 SVBGN   EQU      $-2
28B4 AF        12440      XOR      A          ;If vector points to
28B5 ED42      12450      SBC      HL,BC      ; start of module,
28B7 C8        12460      RET      Z          ; we are finished
28B8 DD6E08    12470      LD      L,(IX+8)    ;P/u DCB being spooled
28BB DD6609    12480      LD      H,(IX+9)
28BE EB        12490      EX       DE,HL      ;SPOOL to HL,
28BF 0603      12500      LD      B,3        ; device to DE
28C1 F3        12510      DI
28C2 7E        12520 SWAP1 LD      A,(HL)    ;Undo the SPOOLer
28C3 3600      12530      LD      (HL),0     ; by swapping back the
28C5 12        12540      LD      (DE),A     ; original vector and
28C6 2C        12550      INC      L        ; zeroing the spool DCB
28C7 1C        12560      INC      E
28C8 10F8      12570      DJNZ     SWAP1
28CA 2C        12580      INC      L        ;Point to name field
28CB 2C        12590      INC      L        ; & wipe it out
28CC 2C        12600      INC      L
28CD 3600      12610      LD      (HL),0
28CF 2C        12620      INC      L
28D0 3600      12630      LD      (HL),0
28D2 FB        12640      EI
28D3 C9        12650      RET              ;Done - return
              12660 ;
              12670 ;      Routine to CLEAR the SPOOL buffer
              12680 ;
28D4 FDCB0346  12690 CLEAR BIT      0,(IY+'D'-'A') ;Spool active?
28D8 CA2524    12700      JP       Z,SPLOFM   ;Go if not active
28DB DD210000  12710      LD      IX,$-$      ;Parms in himem
28DD          12720 HIPARM EQU      $-2
28DF DD6E0B    12730      LD      L,(IX+11)   ;P/u SPLFCB pointer
28E2 DD660C    12740      LD      H,(IX+12)
28E5 DD4E0A    12750      LD      C,(IX+10)   ;P/u RAM bank used
28E8 CBB9      12760      RES      7,C        ;Set on xfer to bank
28EA 0600      12770      LD      B,0        ;Function to load bank
28EC F3        12780      DI              ;Interrupts off now
28ED          12790      @@BANK
28ED 3E66      00087      LD      A,102
28EF EF        00088      RST      40
28F0 C5        12800      PUSH     BC          ;Save old bank

```

```

28F1 E5      12810      PUSH    HL      ;Xfer FCB pointer
28F2 DDE1    12820      POP     IX      ; to IX
28F4 DD6E24  12830      LD      L,(IX+24H) ;Get LINK map pointer
28F7 DD6625  12840      LD      H,(IX+25H)
28FA E5      12850      PUSH    HL
28FB 3600    12860 CLEAR1 LD      (HL),0      ;Reset every link
28FD 23      12870      INC     HL
28FE 3600    12880      LD      (HL),0
2900 23      12890      INC     HL      ;Bypass page pointer
2901 23      12900      INC     HL
2902 7E      12910      LD      A,(HL)      ;Ck for TABEND
2903 23      12920      INC     HL
2904 3C      12930      INC     A      ;End if it was X'FF'
2905 20F4    12940      JR      NZ,CLEAR1
2907 E1      12950      POP     HL      ;Point to LINK start
2908 DD7526  12960      LD      (IX+26H),L ;Reset PUTBLK$
290B DD7427  12970      LD      (IX+27H),H
290E DD7528  12980      LD      (IX+28H),L ;Reset GETBLK$
2911 DD7429  12990      LD      (IX+29H),H
2914 7C      13000      LD      A,H      ;Show 1st LINK position
2915 75      13010      LD      (HL),L      ; is the "last" one by
2916 23      13020      INC     HL      ; loading it to point
2917 77      13030      LD      (HL),A      ; to itself
2918 23      13040      INC     HL      ;P/u the page pointer
2919 46      13050      LD      B,(HL)
291A DD7021  13060      LD      (IX+21H),B ;Stuff PUTNXT$
291D DD362000 13070      LD      (IX+20H),0
2921 DD7023  13080      LD      (IX+23H),B ;Stuff GETNXT$
2924 DD362200 13090      LD      (IX+22H),0
2928 C1      13100      POP     BC      ;Reload previous bank
2929          13110      @BANK
2929 3E66     00089      LD      A,102
292B EF      00090      RST     40
292C FB      13120      EI
292D C31124  13130      JP      CLEARD
          13140 ;
          13150 ; Relocate internal references in driver
          13160 ; HL = relocation table
          13170 ; DE = pointer to last byte of module
          13180 ; BC = pointer to last RAM byte to use
          13190 ;
2930 DDE5    13200 RELO  PUSH    IX      ;Save index reg
2932 E5      13210      PUSH    HL      ;Xfer rel tab pointer
2933 DDE1    13220      POP     IX
2935 210000  13230 SVEND LD      HL,$-$      ;P/u last byte used
2938 22112D  13240      LD      (DVRABGN+2),HL ; & stuff into module
293B 60      13250      LD      H,B      ;Xfer last byte to
293C 69      13260      LD      L,C      ; use to HL
293D AF      13270      XOR     A      ;Clear carry flag
293E ED52    13280      SBC     HL,DE
2940 44      13290      LD      B,H      ;Move to BC
2941 4D      13300      LD      C,L
2942 DD7E00  13310      LD      A,(IX)      ;Get table length
2945 DD23    13320      INC     IX
2947 DD6E00  13330 RLOOP LD      L,(IX)      ;Get address to change
294A DD6601  13340      LD      H,(IX+1)
294D 5E      13350      LD      E,(HL)      ;P/U address
294E 23      13360      INC     HL
294F 56      13370      LD      D,(HL)
2950 EB      13380      EX      DE,HL      ;Offset it
2951 09      13390      ADD     HL,BC

```

```

2952 EB      13400      EX      DE,HL
2953 72      13410      LD      (HL),D      ;And put back
2954 2B      13420      DEC     HL
2955 73      13430      LD      (HL),E
2956 DD23    13440      INC     IX
2958 DD23    13450      INC     IX
295A 3D      13460      DEC     A
295B 20EA    13470      JR      NZ,RLOOP    ;Loop till done
295D DDE1    13480      POP     IX
295F C9      13490      RET
          13500 ;
2960 43      13510 NOFIND$ DB      'Can''t locate SPOOL in memory',CR
        61 6E 27 74 20 6C 6F 63
        61 74 65 20 53 50 4F 4F
        4C 20 69 6E 20 6D 65 6D
        6F 72 79 0D
297D 53      13520 CLEAR$ DB      'SPOOL buffer cleared',CR
        50 4F 4F 4C 20 62 75 66
        66 65 72 20 63 6C 65 61
        72 65 64 0D
2992 44      13530 PAUSED$ DB     'Despooling now paused',CR
        65 73 70 6F 6F 6C 69 6E
        67 20 6E 6F 77 20 70 61
        75 73 65 64 0D
29A8 44      13540 RESUM$ DB      'Despooling now resumed',CR
        65 73 70 6F 6F 6C 69 6E
        67 20 6E 6F 77 20 72 65
        73 75 6D 65 64 0D
29BF 53      13550 SPLOFM$ DB     'Spooler is not active',CR
        70 6F 6F 6C 65 72 20 69
        73 20 6E 6F 74 20 61 63
        74 69 76 65 0D
29D5 53      13560 SPLONM$ DB     'Spooler is already active',CR
        70 6F 6F 6C 65 72 20 69
        73 20 61 6C 72 65 61 64
        79 20 61 63 74 69 76 65
        0D
29EF 53      13570 SPLOPM$ DB     'Spooler already active but paused',CR
        70 6F 6F 6C 65 72 20 61
        6C 72 65 61 64 79 20 61
        63 74 69 76 65 20 62 75
        74 20 70 61 75 73 65 64
        0D
2A11 43      13580 PRMBGM$ DB     'Cannot reinstall with altered parameters',CR
        61 6E 6E 6F 74 20 72 65
        69 6E 73 74 61 6C 6C 20
        77 69 74 68 20 61 6C 74
        65 72 65 64 20 70 61 72
        61 6D 65 74 65 72 73 0D
2A3A 53      13590 SPLACT$ DB     'Spooler is now operational',CR
        70 6F 6F 6C 65 72 20 69
        73 20 6E 6F 77 20 6F 70
        65 72 61 74 69 6F 6E 61
        6C 0D
2A55 43      13600 NOMEM$ DB      'Can''t reclaim memory space',CR
        61 6E 27 74 20 72 65 63
        6C 61 69 6D 20 6D 65 6D
        6F 72 79 20 73 70 61 63
        65 0D
2A70 4D      13610 GOTMEM$ DB     'Memory space reclaimed',CR
        65 6D 6F 72 79 20 73 70

```

	61 63 65 20 72 65 63 6C		
	61 69 6D 65 64 0D		
2A87	42 13620 FREBNK\$ DB	'Bank x released',CR	
	61 6E 6B 20 78 20 72 65		
	6C 65 61 73 65 64 0D		
2A97	52 13630 INUSE\$ DB	'Requested bank in use',CR	
	65 71 75 65 73 74 65 64		
	20 62 61 6E 6B 20 69 6E		
	20 75 73 65 0D		
2AAD	4E 13640 CANT\$ DB	'No memory space available',CR	
	6F 20 6D 65 6D 6F 72 79		
	20 73 70 61 63 65 20 61		
	76 61 69 6C 61 62 6C 65		
	0D		
2AC7	44 13650 BADDCB\$ DB	'Device driver incompatible',CR	
	65 76 69 63 65 20 64 72		
	69 76 65 72 20 69 6E 63		
	6F 6D 70 61 74 69 62 6C		
	65 0D		
0020	13660 SPLDCB DS	32	
2B02	58 13670 DSKSPC DB	'XX/'	
	58 2F		
2B05	53 13680 SPLEXT DB	'SPL',3	
	50 4C 03		
2B09	2A 13690 PR\$ DB	'*PR',ETX	
	50 52 03		
	13700 ;		
2B0D	D2 13710 PRMTBL\$ DB	'R'!80H ;6.x table format	
0080	13720 VAL EQU	80H	
0040	13730 SW EQU	40H	
0020	13740 STR EQU	20H	
0010	13750 SGL EQU	10H	
2B0E	93 13760 DB	VAL!SGL!3,'MEM',0	
	4D 45 4D 00		
2B13	8D26 13770 DW	MPARM+1	
2B15	94 13780 DB	VAL!SGL!4,'DISK',0	
	44 49 53 4B 00		
2B1B	1B26 13790 DW	DPARM+1	
2B1D	43 13800 DB	SW!3,'OFF',0	
	4F 46 46 00		
2B22	1925 13810 DW	OPARM+1	
2B24	41 13820 DB	SW!1,'N',0	
	4E 00		
2B27	1925 13830 DW	OPARM+1	
2B29	94 13840 DB	VAL!SGL!4,'BANK',0	
	42 41 4E 4B 00		
2B2F	B625 13850 DW	BPARM+1	
2B31	55 13860 DB	SW!SGL!5,'PAUSE',0	
	50 41 55 53 45 00		
2B38	2725 13870 DW	PPARM+1	
2B3A	56 13880 DB	SW!SGL!6,'RESUME',0	
	52 45 53 55 4D 45 00		
2B42	2025 13890 DW	RPARM+1	
2B44	55 13900 DB	SW!SGL!5,'CLEAR',0	
	43 4C 45 41 52 00		
2B4B	2E25 13910 DW	CPARM+1	
2B4D	00 13920 NOP		
	13930 ;		
2C00	13940 ORG	\$<-8+1<8	
0100	13950 BUFFER DS	256	

```

13970 ;
2D00 13980 *GET LBSP00LB:3
13990 ;LBSP00LB/ASM - Actual spool driver
14000 ;
14010 ;
14020 ; This is the spool/despool driver - bank 0
14030 ;
2D00 CDC32D 14040 FIX1 CALL SPLR2 ;Code mods to module-A
2D01 14050 RX16 EQU $-2 ; if strictly bank-0
2D03 AF 14060 XOR A
2D04 C9 14070 RET
2D05 CD732E 14080 CALL DSPLR1
2D06 14090 RX17 EQU $-2
2D08 AF 14100 XOR A
2D09 C9 14110 RET
2D0A CDBA2E 14120 CALL GETBLK2
2D0B 14130 RX18 EQU $-2
2D0D AF 14140 XOR A
2D0E C9 14150 RET
14160 ;
2D0F 14170 DVRABGN EQU $
2D0F 1819 14180 JR DVREPT ;Branch around linkage
2D11 0000 14190 DW $-$ ;Last byte used
2D13 05 14200 DB 5
2D14 53 14210 SPOOL$ DB 'SPOOL'
50 4F 4F 4C
2D19 0000 14220 S0DCB DW 0,0 ;Space for DCB pointers
0000
2D1D 0000 14230 PRMSAV DW 0 ;HIGH$ before space alloc
2D1F 0E2D 14240 DW DVRABGN-1 ;HIGH$ after space alloc
2D21 0000 14250 DW 0 ;Previous MPARM
2D23 0000 14260 DW 0 ;Previous DPARM
2D25 0000 14270 MODDVR DW 0 ;DCB address of device
2D27 00 14280 SPLBNK DB 0 ;Bank of RAM
2D28 332F 14290 DW SPLFCB ;FCB pointer
2D28 14300 RX8 EQU $-2
14310 ;
2D2A 2830 14320 DVREPT JR Z,PUTCHAR ;Go on @PUT
2D2C 3003 14330 JR NC,CKCTL ;Go on @CTL
2D2E 3E00 14340 LD A,0 ;Show nothing on @GET
2D30 C9 14350 RET
2D31 AF 14360 CKCTL XOR A
2D32 C9 14370 RET ;Show available on CTL-0
14380 ;
14390 ; Move stack low if needed
14400 ;
2D33 E5 14410 STKCTL PUSH HL
2D34 210580 14420 LD HL,8005H
2D37 39 14430 ADD HL,SP ;Stack in what bank?
2D38 E1 14440 POP HL
2D39 D0 14450 RET NC ;Ok if stack low
2D3A F3 14460 DI
2D3B 225A2D 14470 LD (SVHL),HL
2D3C 14480 RX19 EQU $-2
2D3E E1 14490 POP HL ;Return address
2D3F ED73582D 14500 LD (SAVSTK),SP ;Save the current stack
2D41 14510 RX20 EQU $-2
2D43 316003 14520 LD SP,STACK$-20H ;Force it low
2D46 E5 14530 PUSH HL ;User ret back on
2D47 21512D 14540 LD HL,SPOLRET
2D48 14550 RX21 EQU $-2
2D4A E3 14560 EX (SP),HL ;Now, ultimate ret 1st,

```

The Source	LIBRARY Files	SPOOL - LS-DOS 6.2	Page 00019
2D4B E5	14570	PUSH HL	; and local ret 2nd
2D4C 2A5A2D	14580	LD HL,(SVHL)	;Restore HL
2D4D	14590 RX23	EQU \$-2	
2D4F FB	14600	EI	
2D50 C9	14610	RET	
	14620 ;		
	14630 SPOLRET		
2D51 F3	14640	DI	;Just in case
2D52 ED7B582D	14650	LD SP,(SAVSTK)	;Get the old stack
2D54	14660 RX22	EQU \$-2	
2D56 FB	14670	EI	
2D57 C9	14680	RET	
2D58 0000	14690 SAVSTK	DW 0	
2D5A 0000	14700 SVHL	DW 0	
	14710 ;		
	14720 ;	Character to PUT - Get bank activated	
	14730 ;		
2D5C 21B82D	14740 PUTCHAR	LD HL,SPLR1A	;Point to transfer addr
2D5D	14750 RX01	EQU \$-2	
2D5F 59	14760	LD E,C	;Xfer char to PUT
2D60 018000	14770	LD BC,0<8!0!80H	;P/u function & bank req
2D61	14780 BANKX1	EQU \$-2	
2D63 CD332D	14790	CALL STKCTL	;Move stack low?
2D64	14800 RX24	EQU \$-2	
2D66	14810	@@BANK	; & invoke bank transfer
2D66 3E66	00091	LD A,102	
2D68 EF	00092	RST 40	
2D69 C9	14820	RET	
	14830 ;		
	14840 ;	Task processor despooling routine	
	14850 ;		
2D6A 6C2D	14860 DSPLTCB	DW DSPLR	;Despool task control blk
2D6A	14870 RX02	EQU \$-2	
2D6C 3A0000	14880 DSPLR	LD A,(\$-\$)	;P/u PFLAG\$ & ck if
2D6D	14890 DFLAG	EQU \$-2	; despooling is active
2D6F 07	14900	RLCA	
2D70 D0	14910	RET	;Back if not
2D71 2A192D	14920	LD HL,(S0DCB)	;If spooler is inactive
2D72	14930 RX08	EQU \$-2	; then don't try to
2D74 2C	14940	INC L	; do any despooling
2D75 4E	14950	LD C,(HL)	;Get DCB vector
2D76 2C	14960	INC L	
2D77 46	14970	LD B,(HL)	; into BC
2D78 210F2D	14980	LD HL,DVRABGN	;P/u spooler start
2D79	14990 RX09	EQU \$-2	
2D7B AF	15000	XOR A	
2D7C ED42	15010	SBC HL,BC	;If the same, then
2D7E C8	15020	RET	; module is not active
2D7F 21642E	15030 FIX2UP	LD HL,DSPLR0	;Point to transfer addr
2D80	15040 RX03	EQU \$-2	
2D82 018000	15050	LD BC,0<8!0!80H	;P/u function & bank req
2D83	15060 BANKX2	EQU \$-2	
2D85 CD332D	15070	CALL STKCTL	;Move stack low?
2D86	15080 RX25	EQU \$-2	
2D88	15090	@@BANK	; & invoke bank transfer
2D88 3E66	00093	LD A,102	
2D8A EF	00094	RST 40	
2D8B C9	15100	RET	
	15110 ;		
	15120 ;	KI task routine to get a disk block	
	15130 ;		

The Source	LIBRARY Files	SPPOOL - LS-DOS 6.2	Page 00020
2D8C 21B02E	15140 GETBLK LD	HL,GETBLK1 ;Point to transfer addr	
2D8D	15150 RX04 EQU	\$-2	
2D8F 018000	15160 LD	BC,0<8!0!80H ;P/u function & bank req	
2D90	15170 BANKX3 EQU	\$-2	
2D92 CD332D	15180 CALL	STKCTL ;Move stack low?	
2D93	15190 RX26 EQU	\$-2	
2D95	15200 @BANK	; & invoke bank transfer	
2D95 3E66	00095 LD	A,102	
2D97 EF	00096 RST	40	
2D98 00	15210 GETBLK DB	0,0,0 ;Orig @KITSK	
00 00			
2D9A	15220 DVRAEND EQU	\$-1	
2D9B 0D	15230 RELTAB DB	TABALEN	
2D9C 6A2D	15240 DW	RX02,RX05,RX06,RX08,RX09	
D327 DB27 722D	792D		
2DA6 3C2D	15250 DW	RX19,RX20,RX21,RX22,RX23	
412D 482D 542D 4D2D			
2DB0 642D	15260 DW	RX24,RX25,RX26	
862D 932D			
000D	15270 TABALEN EQU	\$-RELTABA/2	
	15280 ;		
	15290 ;		
2DB6	15300 DVRBBGN EQU	\$	
	15310 ;		
	15320 ;	Spool/despool module in bank x	
	15330 ;		
2DB6 0000	15340 S0DCB1 DW	0	
	15350 ;		
	15360 ;	Spool a character	
	15370 ;		
2DB8 E5	15380 SPLR1A PUSH	HL ;Save new transfer addr	
2DB9 C5	15390 PUSH	BC ;Save invocation	
2DBA 4B	15400 LD	C,E ;Get char to PUT	
2DBB CDC32D	15410 CALL	SPLR2 ;Spool a char	
2DBC	15420 RX1 EQU	\$-2	
2DBE C1	15430 POP	BC	
2DBF E1	15440 POP	HL	
2DC0	15450 @BANK	;Transfer back	
2DC0 3E66	00097 LD	A,102	
2DC2 EF	00098 RST	40	
2DC3 DD21332F	15460 SPLR2 LD	IX,SPLFCB	
2DC5	15470 RX2 EQU	\$-2	
2DC7 DD6E20	15480 LD	L,(IX+20H) ;P/u next buffer position	
2DCA DD6621	15490 LD	H,(IX+21H)	
2DCD 71	15500 LD	(HL),C ;Stuff char received	
2DCE F3	15510 DI		
2DCF DD3420	15520 SPLR3 INC	(IX+20H) ;Inc lo order	
2DD2 2807	15530 JR	Z,SPLR5 ;Jump if block full	
2DD4 CD6E2E	15540 CALL	DSPLR1X ;Output to *PR if avail	
2DD5	15550 RX3 EQU	\$-2	
2DD7 FB	15560 EI		
2DD8 C3BA2E	15570 JP	GETBLK2	
2DD9	15580 RX4 EQU	\$-2	
2DOB DD6E24	15590 SPLR5 LD	L,(IX+24H) ;P/u pointer to table	
2DDE DD6625	15600 LD	H,(IX+25H)	
2DE1 7E	15610 SPLR6 LD	A,(HL) ;Find spare pointer field	
2DE2 23	15620 INC	HL	
2DE3 B6	15630 OR	(HL)	
2DE4 2805	15640 JR	Z,SPLR7 ;Jump if spare	
2DE6 23	15650 INC	HL ; else index to next	
2DE7 23	15660 INC	HL	


```

2DE8 23      15670      INC      HL
2DE9 18F6    15680      JR        SPLR6
2DEB 2B      15690 SPLR7  DEC      HL
2DEC E5      15700      PUSH     HL          ;Save pointer
2DED 23      15710      INC      HL
2DEE 23      15720      INC      HL
2DEF 46      15730      LD        B,(HL)      ;P/u memory segment byte
2DF0 23      15740      INC      HL          ; or lo-order recno
2DF1 7E      15750      LD        A,(HL)      ;End of table?
2DF2 FEFF    15760      CP        0FFH
2DF4 200B    15770      JR        NZ,SPLR10      ;Jump if still space
2DF6 E1      15780      POP       HL
2DF7 CD6E2E  15790 SPLR8  CALL     DSPLR1X      ;Output to *PR if avail
2DF8        15800 RX5    EQU      $-2
2DFA FB      15810      EI
2DFB CDBA2E  15820 SPLR9  CALL     GETBLK2      ;Disk rec to bring back?
2DFC        15830 RX6    EQU      $-2
2DFE F3      15840      DI
2DFF 18DA    15850      JR        SPLR5
2E01 CB7F    15860 SPLR10 BIT      7,A          ;Test mem or disk block
2E03 2021    15870      JR        NZ,SPLR12      ;Jump if disk pointer
          15880 ;
          15890 ;      Found a spare memory pointer
          15900 ;
2E05 E1      15910      POP       HL          ;Rcvr ptr to start
2E06 E5      15920      PUSH     HL          ; of spare table field
2E07 75      15930      LD        (HL),L      ;Place address of table
2E08 7C      15940      LD        A,H          ; position into table
2E09 23      15950      INC      HL
2E0A 77      15960      LD        (HL),A
2E0B DD7021  15970      LD        (IX+21H),B      ;Stuff hi-order free mem
2E0E DD6E26  15980      LD        L,(IX+26H)      ;P/u current table ptr
2E11 DD6627  15990      LD        H,(IX+27H)
2E14 DD752A  16000      LD        (IX+2AH),L      ;Stuff last block pointer
2E17 DD742B  16010      LD        (IX+2BH),H
2E1A C1      16020      POP       BC          ;Rcvr ptr to field start
2E1B 71      16030      LD        (HL),C      ;Set up a link to this
2E1C 23      16040      INC      HL          ; block in last block
2E1D 70      16050      LD        (HL),B
2E1E DD7126  16060      LD        (IX+26H),C      ;Reset current block ptr
2E21 DD7027  16070      LD        (IX+27H),B
2E24 FB      16080      EI
2E25 C9      16090      RET
          16100 ;
2E26        16110 SPLR12 EQU      $
2E26 FB      16120      EI
2E27 48      16130      LD        C,B          ;Lo order recno
2E28 E60F    16140      AND      0FH          ;Hi-order recno
2E2A 47      16150      LD        B,A
2E2B DDE5    16160      PUSH     IX          ;Position file to recno
2E2D D1      16170      POP       DE          ;FCB to DE
2E2E        16180      @@POSN      ;Posn to desired record
2E2E 3E42    00099      LD        A,66
2E30 EF      00100      RST        40
2E31 00      16190      DB        0,0,0      ;Reserved
          00 00
2E34 DD4E26  16200      LD        C,(IX+26H)      ;P/u current table ptr
2E37 DD4627  16210      LD        B,(IX+27H)
2E3A E1      16220      POP       HL          ;P/u ptr to this disk
2E3B E5      16230      PUSH     HL          ;Block table field
2E3C 71      16240      LD        (HL),C      ;Tell this disk block

```

The Source	LIBRARY Files	SPOOL - LS-DOS 6.2	Page 00022
2E3D 23	16250	INC HL	; field which memory
2E3E 70	16260	LD (HL),B	; block it is storing
2E3F C1	16270	POP BC	;Exchange link in last
2E40 DD6E2A	16280	LD L,(IX+2AH)	; block pointer with new
2E43 DD662B	16290	LD H,(IX+2BH)	; link since this memory
2E46 71	16300	LD (HL),C	; block is now on disk
2E47 23	16310	INC HL	
2E48 70	16320	LD (HL),B	
2E49 DD712A	16330	LD (IX+2AH),C	
2E4C DD702B	16340	LD (IX+2BH),B	
2E4F 010001	16350	LD BC,256	;Block move the page
2E52 69	16360	LD L,C	; into the low memory
2E53 DD6621	16370	LD H,(IX+21H)	; buuffer fo disk I/O
2E56 D5	16380	PUSH DE	;Save fcb pointer
2E57 110023	16390	LD DE,LOWBUF\$	
2E5A EDB0	16400	LDIR	
2E5C D1	16410	POP DE	
2E5D	16420	@@WRITE	;Write out the block
2E5D 3E4B	00101	LD A,75	
2E5F EF	00102	RST 40	
2E60 00	16430	DB 0,0,0	;Reserved
00 00			
2E63 C9	16440	RET	
	16450 ;		
2E64 E5	16460 DSPLR0	PUSH HL	;Save new transfer addr
2E65 C5	16470	PUSH BC	;Save invocation
2E66 CD732E	16480	CALL DSPLR1	;Send device a char
2E67	16490 RX7	EQU \$-2	
2E69 C1	16500	POP BC	
2E6A E1	16510	POP HL	
2E6B	16520	@@BANK	;Transfer back
2E6B 3E66	00103	LD A,102	
2E6D EF	00104	RST 40	
	16530 ;		
	16540 ;		
	16550 ;		
			Routine will output a char to PR if available
2E6E 3A0000	16560 DSPLR1X	LD A,(\$-\$)	;P/u PFLAG\$ & ck if
2E6F	16570 DFLAG1	EQU \$-2	; despooling is active
2E71 07	16580	RLCA	
2E72 D0	16590	RET NC	;Back if not active
2E73 010004	16600 DSPLR1	LD BC,4<8!0	;Init to CTL-0
2E76 DDE5	16610	PUSH IX	;Save current DCB vector
2E78 DD2AB62D	16620 PUDCB	LD IX,(S0DCB1)	;Point to printer
2E7A	16630 RX9	EQU \$-2	
2E7C	16640	@@CHNIO	;Ck *PR status
2E7C 3E14	00105	LD A,20	
2E7E EF	00106	RST 40	
2E7F DDE1	16650	POP IX	
2E81 C0	16660	RET NZ	;Ret if not avail
2E82 DD21332F	16670 PUFCB	LD IX,SPLFCB	
2E84	16680 RX10	EQU \$-2	
2E86 DD7E23	16690	LD A,(IX+23H)	;See if char in buffer
2E89 B7	16700	OR A	; to send to the printer
2E8A C8	16710	RET Z	;Ret if none
2E8B 67	16720	LD H,A	;None also if GET pointer
2E8C DDBE21	16730	CP (IX+21H)	; is same as PUT pointer
2E8F DD7E22	16740	LD A,(IX+22H)	
2E92 2004	16750	JR NZ,DSPLR2	
2E94 DDBE20	16760	CP (IX+20H)	
2E97 C8	16770	RET Z	;In & out the same
2E98 6F	16780 DSPLR2	LD L,A	;P/u the character

```

2E99 4E      16790      LD      C,(HL)      ;Xfer to output reg
2E9A DDE5    16800      PUSH     IX          ;Save this dcb vector
2E9C 0602    16810      LD      B,2         ;Set chain for PUT
2E9E DD2AB62D 16820 PUDCB1 LD      IX,(S0DCB1) ;Pt to spooled DCB
2EA0        16830 RX11   EQU      $-2
2EA2        16840      @@CHNIO              ;Output the char
2EA2 3E14    00107      LD      A,20
2EA4 EF      00108      RST      40
2EA5 DDE1    16850      POP      IX          ;Rcvr dcb
2EA7 DD3422  16860      INC      (IX+22H)    ;Bump PUT (out) ptr
2EAA C0      16870      RET      NZ          ;Ret if still more in buf
2EAB DD362300 16880      LD      (IX+23H),0    ;Block empty, turn off
2EAF C9      16890      RET                  ; hi-order & exit
          16900 ;
          16910 ;
          16920 ;      Routine to see if we can grab a block back from
          16930 ;      the disk file & insert into a spare memory block
          16940 ;
2EB0 E5      16940 GETBLK1 PUSH     HL          ;Save new transfer addr
2EB1 C5      16950      PUSH     BC          ;Save invocation
2EB2 CDBA2E  16960      CALL     GETBLK2      ;Spool a char
2EB3        16970 RX12   EQU      $-2
2EB5 C1      16980      POP      BC
2EB6 E1      16990      POP      HL
2EB7        17000      @@BANK              ;Transfer back
2EB7 3E66    00109      LD      A,102
2EB9 EF      00110      RST      40
2EBA F5      17010 GETBLK2 PUSH     AF
2EBB DDE5    17020      PUSH     IX          ;Save in case DSPLR1
2EBD 3A562F  17030 GETBLK3 LD      A,(SPLFCB+23H) ;Jump if block being
2EBE        17040 RX13   EQU      $-2
2EC0 B7      17050      OR      A           ; printed is not empty
2EC1 202D    17060      JR      NZ,GETBLK6
          17070 ;
          17080 ;      The block's been despoiled. Get 1 from D,M?
          17090 ;
2EC3 E5      17100      PUSH     HL
2EC4 D5      17110      PUSH     DE
2EC5 C5      17120      PUSH     BC
2EC6 DD21332F 17130 GETBLK4 LD      IX,SPLFCB
2EC8        17140 RX14   EQU      $-2
2ECA DD6E28  17150      LD      L,(IX+28H)    ;P/u pointer of block
2ECD DD6629  17160      LD      H,(IX+29H)    ; being printed
2ED0 E5      17170      PUSH     HL
2ED1 4E      17180      LD      C,(HL)      ;Point to POSN or hi-mem
2ED2 23      17190      INC      HL          ; byte of field this
2ED3 46      17200      LD      B,(HL)      ; field points to
2ED4 210200  17210      LD      HL,2        ;P/u the POSN or hi-order
2ED7 09      17220      ADD      HL,BC
2ED8 7E      17230      LD      A,(HL)
2ED9 23      17240      INC      HL
2EDA CB7E    17250      BIT      7,(HL)      ;Jump if it is pointing
2EDC 201B    17260      JR      NZ,GETBLK7    ; to a disk block
2EDE DD7723  17270      LD      (IX+23H),A    ;Reset hi order of where
2EE1 E1      17280      POP      HL          ; next block coming from
2EE2 3600    17290      LD      (HL),0       ;Show that this block
2EE4 23      17300      INC      HL          ; is spare
2EE5 3600    17310      LD      (HL),0
2EE7 DD7128  17320      LD      (IX+28H),C    ;Set the next block
2EEA DD7029  17330      LD      (IX+29H),B    ; to be printed
2EED C1      17340 GETBLK5 POP      BC
2EEE D1      17350      POP      DE

```

```

2EEF E1      17360      POP      HL
2EF0         17370 GETBLK6 EQU      $
2EF0 F3      17380      DI
2EF1 CD6E2E  17390      CALL     DSPLR1X      ;Check if we can despool
2EF2         17400 RX15    EQU      $-2      ; a character while
2EF4 FB      17410      EI              ; we are in here
2EF5 DDE1    17420      POP      IX
2EF7 F1      17430      POP      AF
2EF8 C9      17440      RET
              17450 ;
              17460 ;      Next block to print is on disk, get it
              17470 ;
2EF9 4F      17480 GETBLK7 LD      C,A      ;Set up position
2EFA 7E      17490      LD      A,(HL)
2EFB E60F    17500      AND      0FH
2EFD 47      17510      LD      B,A
2EFE DDE5    17520      PUSH     IX      ;Set FCB & position
2F00 D1      17530      POP      DE      ; the spool file
2F01         17540      @@POSN
2F01 3E42    00111      LD      A,66
2F03 EF      00112      RST      40
2F04 00      17550      DB      0,0,0      ;Reserved
              00 00
2F07 E1      17560      POP      HL      ;Rcvr table ptr of
2F08 E5      17570      PUSH     HL      ; block just finished
2F09 23      17580      INC      HL
2F0A 23      17590      INC      HL      ;P/u its buffer location
2F0B         17600      @@READ      ;Read block from file
2F0B 3E43    00113      LD      A,67
2F0D EF      00114      RST      40
2F0E 00      17610      DB      0,0,0      ;Reserved
              00 00
2F11 010001  17620      LD      BC,256      ;Block move from the
2F14 56      17630      LD      D,(HL)      ; low memory buffer
2F15 59      17640      LD      E,C      ; to RAM page
2F16 210023  17650      LD      HL,LOWBUF$
2F19 EDB0    17660      LDIR
2F1B E1      17670      POP      HL      ;Tbl ptr of block fin'd
2F1C E5      17680      PUSH     HL
2F1D 7E      17690      LD      A,(HL)      ;P/u field address of
2F1E 23      17700      INC      HL      ; the next table field
2F1F 66      17710      LD      H,(HL)      ; that gets printed
2F20 6F      17720      LD      L,A
2F21 4E      17730      LD      C,(HL)      ;P/u field address of the
2F22 3600    17740      LD      (HL),0      ; block after that to
2F24 23      17750      INC      HL      ; print & show this one
2F25 46      17760      LD      B,(HL)      ; as empty (this was the
2F26 3600    17770      LD      (HL),0      ; disk one just vacated)
2F28 E1      17780      POP      HL
2F29 71      17790      LD      (HL),C      ;Reset current printing
2F2A 23      17800      INC      HL      ; block & point to
2F2B 70      17810      LD      (HL),B      ; new next block
2F2C 23      17820      INC      HL
2F2D 46      17830      LD      B,(HL)
2F2E DD7023  17840      LD      (IX+23H),B      ;Show where new printing
2F31 18BA    17850      JR      GETBLK5      ; block is & exit
              17860 ;
              17870 ;      File control block area for spool file
              17880 ;
0020         17890 SPLFCB DS      32
              17900 ;

```

```

17910 ;      Pointer to the memory location where the next
17920 ;      character received from *XX can be put
17930 ;
0002 17940 PUTNXT$ DS      2
17950 ;
17960 ;      Pointer to where the next character to print
17970 ;      is going to come from
17980 ;
0002 17990 GETNXT$ DS      2
18000 ;
18010 ;      Pointer to start of block map table
18020 ;      table configuration is:
18030 ;      TAB+0/1 -> store the TAB address if the field
18040 ;      is currently in use for receiving. If a
18050 ;      block is not current, it stores the
18060 ;      address of the next TAB to print. If
18070 ;      a disk field, then stores the TAB address
18080 ;      of the previous block (link back)
18090 ;
18100 ;      TAB+2 -> for a memory block, stores the hi-order
18110 ;      byte of where the memory block starts. If
18120 ;      a disk byte, stores the lo-order POSN #.
18130 ;
18140 ;      TAB+3 -> s/b 0 for a memory block or if a disk
18150 ;      block, it contains the hi-order POSN
18160 ;      vector with bit 7 set.
18170 ;
0002 18180 MAPPTR$ DS      2
18190 ;
18200 ;      Pointer to the beginning of the memory table
18210 ;      block pointer for the block that is receiving
18220 ;      characters from *PR calls (the current block)
18230 ;
0002 18240 PUTBLK$ DS      2
18250 ;
18260 ;      Pointer to the memory table field for the block
18270 ;      that is printing characters
18280 ;
0002 18290 GETBLK$ DS      2
18300 ;
18310 ;      Points to the table position of the previous
18320 ;      block. This is used to reset the previous link
18330 ;
2F5D 0000 18340          DW      0
2F5E      18350 DVRBEND EQU    $-1
2F5F 15    18360 RELTABB DB     TABBLN
2F60 5D2D 18370          DW     RX01,RX03,RX04
      802D 8D2D
2F66 BC2D 18380          DW     RX1,RX2,RX3,RX4,RX5,RX6,RX7,RX8
      C52D D52D D92D F82D FC2D 672E 282D
2F76 7A2E 18390          DW     RX9,RX10,RX11,RX12,RX13,RX14,RX15
      842E A02E B32E BE2E C82E F22E
2F84 012D 18400          DW     RX16,RX17,RX18
      062D 0B2D
0015      18410 TABBLN EQU    $-RELTABB/2
      18420 ;
      00140 ;
2F8A      00150          SUBTTL <>
2400      00160          END    SPOOL

```

@\$SYS	08F0 @01	0000 @02	0000
@03	0000 @04	0000 @BANK	0877
@BYTEIO	1300 @CHNIO	0689 @CKBRKC	0553
@CLS	0545 @CTL	0623 @DATE	07A8
@DIV16	06E3 @DSP	0642 @DSPLY	052D
@FRENCH	0000 @GERMAN	0000 @GET	0638
@HEX16	07BD @HEX8	07C2 @HEXDEC	06F6
@HZ50	0000 @INTL	0000 @JCL	0630
@KBD	0635 @KEY	0628 @KEYIN	0585
@KITSK	0089 @LOGGER	0503 @LOGOT	0500
@MOD2	0000 @MOD4	FFFF @MSG	0530
@MUL16	06C9 @OPREG	0084 @PRINT	0528
@PRT	063D @PUT	0645 @RSTNMI	0FE9
@RSTREG	0680 @TIME	078D @USA	FFFF
@VDCTL	0B99 @VDCTL3	0D38 @ VDCTL	0D42
ADDR_2_ROWCOL	0DF1 BADDGB	2442 BADDGB\$	2AC7
BANKX1	2D61 BANKX2	2D83 BANKX3	2D90
BAR\$	0201 BOOTST\$	439D BPARM	25B5
BUFFER	2C00 BUR\$	0200 CALCPTR	26A1
CANT	244E CANT\$	2AAD CASHK\$	0A7B
CFLAG\$	006C CKCTL	2D31 CKLNK	258A
CKRTE	257F CKRTE1	2583 CLEAR	28D4
CLEAR1	28FB CLEAR0	2411 CLEAR0\$	297D
CORE\$	0300 CPARM	252D CR	0000
CRTBGN\$	F800 DATE\$	0033 DAYTBL\$	04C7
DCBKL\$	0031 DCT\$	0470 DFLAG	2D6D
DFLAG\$	006D DFLAG1	2E6F 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
DPARM	261A DSKSPC	2B02 DSKTYP\$	04C0
DSPLR	2D6C DSPLR0	2E64 DSPLR1	2E73
DSPLR1X	2E6E DSPLR2	2E98 DSPLTCB	2D6A
DTPMT\$	04C2 DVRABGN	2D0F DVRAEND	2D9A
DVRBBGN	2DB6 DVRBEND	2F5E DVREND\$	0FF4
DVREPT	2D2A DVRHI\$	0206 ENADIS DO RAM	0817
ERREXIT	2454 ETX	0003 EXIT	2407
FDDINT\$	000E FIX1	2D00 FIX2UP	2D7F
FLGTAB\$	006A FNDDCB	2575 FREBNK\$	2A87
GBUF1	26BB GBUF2	26C4 GETBLK	2D8C
GETBLK\$	2F5B GETBLK0	2D98 GETBLK1	2EB0
GETBLK2	2EBA GETBLK3	2EBD GETBLK4	2EC6
GETBLK5	2EED GETBLK6	2EF0 GETBLK7	2EF9
GETNXT\$	2F55 GETPRM	250F GET @ ROWCOL	0DAE
GOTMEM\$	2A70 HERTZ\$	0750 HIGH\$	040E
HIPARM	28DD IFLAG\$	0072 INBUF\$	0420
INTVC\$	003E INUSE	244A INUSE\$	2A97
IOERR	2437 IPLDSK1	26FE IPLDSK2	2708
IPLDSK3	271A IPLMEM	26F0 JCLCB\$	0203
JLDCB\$	0230 KCK0	07D6 KFLAG\$	0074
KIDATA\$	08FC KIDCB\$	0208 KITSK0	001F
LBANK\$	0202 LOCDCB	25D7 LOGMEM	2888
LOWBUF\$	2300 MAPPTR\$	2F57 MAXDAY\$	0401
MODDVR	2D25 MODOUT\$	0076 MONTBL\$	04DC
MOVMOVB	277B MPARM	268C NAMLP	25EC
NFLAG\$	0077 NOF IND	2446 NOF IND\$	2960
NOMEM\$	2A55 NOSET	27C8 OLDHI	27BC

OPARM	2518	OPREG\$	0078	OPREG_SV_AREA	086E
OPREG_SV_PTR	0835	PAKNAM\$	0410	PAR_ERR	002C
PAUSE	2807	PAUSE@	0382	PAUSED	2419
PAUSED\$	2992	PCSAVE\$	07AF	PDRV\$	001B
PPARM	2526	PR\$	2B09	PRDCB\$	0218
PRMBGM	242D	PRMBGM\$	2A11	PRMERR	2435
PRMERRA	2620	PRMSAV	2D1D	PRMTBL\$	2B0D
PUDCB	2E78	PUDCB1	2E9E	PUFCB	2E82
PUHIGH	2658	PUTA@DE	0DCD	PUTBLK\$	2F59
PUTCHAR	2D5C	PUTNXT\$	2F53	PUT @	0DCA
PUT @ ROWCOL	0DC6	RELO	2930	RELTABA	2D9B
RELTABB	2F5F	RESDCB	28A3	RESUMD	2415
RESUMD\$	29A8	RESUME	281C	RFLAG\$	007B
RLOOP	2947	ROWCOL_2_ADDR	0DD0	RPARM	251F
RST28	0028	RSTOR\$	04C4	RX01	2D5D
RX02	2D6A	RX03	2D80	RX04	2D8D
RX05	27D3	RX06	27DB	RX08	2D72
RX09	2D79	RX1	2DBC	RX10	2E84
RX11	2EA0	RX12	2EB3	RX13	2EBE
RX14	2EC8	RX15	2EF2	RX16	2D01
RX17	2D06	RX18	2D0B	RX19	2D3C
RX2	2DC5	RX20	2D41	RX21	2D48
RX22	2D54	RX23	2D4D	RX24	2D64
RX25	2D86	RX26	2D93	RX3	2DD5
RX4	2DD9	RX5	2DF8	RX6	2DFC
RX7	2E67	RX8	2D28	RX9	2E7A
S0DCB	2D19	S0DCB1	2DB6	S1DCB\$	0238
SAVESP	240A	SAVSTK	2D58	SETDVR	25AD
SET_SCROLL	0CF3	SFLAG\$	007C	SGL	0010
SIDCB\$	0220	SODCB\$	0228	SPL1	249B
SPL2	249C	SPL2A	24CE	SPL2B	24DE
SPL2C	24E7	SPL3	24F3	SPL4	24FF
SPLACT	241D	SPLACT\$	2A3A	SPLBNK	2D27
SPLDCB	2AE2	SPLXT	2B05	SPLFCB	2F33
SPLOFF	2831	SPLOFM	2425	SPLOFM\$	29BF
SPLONM	2421	SPLONM\$	29D5	SPLOPM	2429
SPLOPM\$	29EF	SPLR10	2E01	SPLR12	2E26
SPLR1A	2DB8	SPLR2	2DC3	SPLR3	2DCF
SPLR5	2DDB	SPLR6	2DE1	SPLR7	2DEB
SPLR8	2DF7	SPLR9	2DFB	SPOLRET	2D51
SPOOL	2400	SPOOL\$	2D14	SPOOL1	2459
STACK\$	0380	START\$	0000	STKCTL	2D33
STORHI	27A2	STR	0020	SVBGN	28B2
SVDCB	28A4	SVEND	2935	SVHL	2D5A
SW	0040	SWAP	27FA	SWAP1	28C2
TABALEN	000D	TABBLEN	0015	TIMES\$	002D
TIMER\$	002C	TIMSL\$	002B	TIMTSK\$	0713
TMPMT\$	04C3	TRACE_INT	07B1	TYPHK\$	0A8F
TYPTSK\$	0B26	USEPR	24C3	VAL	0080
VFLAG\$	007F	ZERO\$	0401	@@ABORT	6C50
@@ADTSK	6CE3	@@BANK	71FB	@@BKSP	6EDB
@@BREAK	7211	@@CHNIO	6C3B	@@CKBRKC	725F
@@CKDRV	6D37	@@CKEOF	6EF0	@@CKTSK	6CCE
@@CLOSE	6EC6	@@CLS	7249	@@CMNDI	6C7A
@@CMNDR	6C8F	@@CTL	6A9F	@@DATE	6C11
@@DCSTAT	6D76	@@DEBUG	6CB9	@@DECHEX	717B
@@DIRR0	70E8	@@DIRWR	70FD	@@DIV16	7166
@@DIV8	7151	@@DODIR	6D4C	@@DSP	6A63
@@DSPLY	6B03	@@ERROR	6CA4	@@EXIT	6C65

@@FEXT	7055 @@FLAGS	71E5 @@FNAME	706A
@@FSPEC	7040 @@GATRD	70D3 @@GATWR	7112
@@GET	6A77 @@GTDCB	7094 @@GTDCT	707F
@@GTMOD	70A9 @@HDFMT	6E1E @@HEX16	71BA
@@HEX8	71A5 @@HEXDEC	7190 @@HIGH\$	71CF
@@INIT	6E9C @@KBD	6ADB @@KEY	6A4F
@@KEYIN	6AEF @@KLTSK	6D22 @@LOAD	7016
@@LOC	6F05 @@LOF	6F1A @@LOGGER	6B3A
@@LOGOT	6B4F @@MSG	6B86 @@MUL16	713C
@@MUL8	7127 @@OPEN	6EB1 @@PARAM	6BFC
@@PAUSE	6BE7 @@PEOF	6F2F @@POSN	6F44
@@PRINT	6B9B @@PRT	6AB3 @@PUT	6A8B
@@RAMDIR	6D61 @@RDSEC	6DF4 @@RDSSC	70BE
@@READ	6F59 @@REMOV	6E87 @@RENAM	6E72
@@REW	6F6E @@RMTSK	6CF8 @@RPTSK	6D0D
@@RREAD	6F83 @@RSLCT	6DDF @@RSTOR	6DA0
@@RUN	702B @@RWRIT	6F98 @@SEEK	6DCA
@@SEEKSC	6FAD @@SKIP	6FC2 @@SLCT	6D8B
@@STEPI	6DB5 @@TIME	6C26 @@VDCTL	6BD2
@@VER	6FD7 @@VRSEC	6E09 @@WEOF	6FEC
@@WHERE	6AC7 @@WRITE	7001 @@WRSEC	6E33
@@WRSSC	6E48 @@WTRK	6E5D	

2400 is the transfer address
 00000 Total errors

NOTES:

Command: SYSGEN

Library: SYS8/SYS

ISAM # : 1CH

```

00100 ;LBSYSGEN/ASM - SYSGEN Command
0000 00110 TITLE <SYSGEN - LS-DOS 6.2>
00120 ;
002C 00130 PAR_ERR EQU 44 ;Parameter Error
000D 00140 CR EQU 13
4300 00150 MOD3BUF EQU 4300H
0028 00160 RST28 EQU 28H
00170 ;
0000 00180 *GET SVCMAC:3 ;SVC Macro equivalents
00010 ;SVCMAC/ASM - LS-DOS Version VI
00020 *LIST OFF
03900 *LIST ON
00190 *LIST OFF ;Get SYS0/EQU
00210 *LIST ON
00220 ;
2400 00230 ORG 2400H
00240 ;
2400 00250 SYSGEN @@CKBRKC ;Break key down?
2400 3E6A 00001 LD A,106
2402 EF 00002 RST 40
2403 2804 00260 JR Z,BEGINA ;Ok if not
2405 21FFFF 00270 LD HL,-1 ; else abort
2408 C9 00280 RET
00290 ;
00300 ; Save stack & call the sysgen routine
00310 ;
2409 ED731424 00320 BEGINA LD (SAVE$P+1),SP ;Save the stack pointer
240D CD4624 00330 CALL SYSGEN1 ;Do the SYSGEN
2410 210000 00340 EXIT LD HL,0 ;Set for no error
2413 310000 00350 SAVESP LD SP,$-$ ;P/u original SP
2416 00360 @@CKBRKC ;Clear pending <BREAK>
2416 3E6A 00003 LD A,106
2418 EF 00004 RST 40
2419 C9 00370 RET ;Back to DOS
00380 ;
00390 ; I/O Error Handler
00400 ;
241A 3E2C 00410 PRMERR LD A,PAR_ERR ;Parameter Error
241C 6F 00420 IOERR LD L,A ;Error # to HL
241D 2600 00430 LD H,0
241F F6C0 00440 OR C00H ;Abbrev, return
2421 4F 00450 LD C,A
2422 00460 @@ERROR ;Display error
2422 3E1A 00005 LD A,26
2424 EF 00006 RST 40
2425 18EC 00470 JR SAVESP ;Exit
00480 ;
00490 ; Internal Error Message Handler
00500 ;
2427 213D27 00510 NOMDSK LD HL,NOMDSK$ ;"Memdisk active
242A DD 00520 DB 0DDH
242B 215B27 00530 NORTES LD HL,NORTES$ ;"File routes active
242E DD 00540 DB 0DDH
242F 219126 00550 NOTFND LD HL,NOTFND$ ;"No config found
2432 DD 00560 DB 0DDH
2433 212127 00570 NOSPL LD HL,NOSPL$ ;"Spooler active
2436 DD 00580 DB 0DDH
2437 21D626 00590 NOINDO LD HL,NOINDO$ ;"JCL active
243A DD 00600 DB 0DDH
243B 21FD26 00610 NOCANDO LD HL,NOCANDO$ ;"Sysgen inhibit set
243E CD0005 00620 CALL @LOGOT

```

```

2441 21FFFF 00630 LD HL,-1 ;Set abort code
2444 18CD 00640 JR SAVESP ;Exit
00650 ;
00660 ; SYSGEN1 - Save the state of the system
00670 ;
2446 117F27 00680 SYSGEN1 LD DE,PRMTBL$ ;Check for user parameters
2449 00690 @@PARAM
2449 3E11 00007 LD A,17
244B EF 00008 RST 40
244C C21A24 00700 JP NZ,PRMERR ;Quit on parm error
244F 00710 @@FLAGS
244F 3E65 00009 LD A,101
2451 EF 00010 RST 40
00720 ;
00730 ; Routine to process DRIVE parameter
00740 ;
2452 010000 00750 DRIVE LD BC,0 ;Drive parm response
2455 79 00760 LD A,C
2456 FE08 00770 CP 8 ;Drive in range?
2458 D21A24 00780 JP NC,PRMERR ;Go if > 7
245B F630 00790 OR '0' ;Make it ASCII
245D 322A26 00800 LD (CFGSPEC),A ; & stuff drive spec
2460 327026 00810 LD (ATRSPEC),A
2463 CDEB25 00820 CALL NPARM ;Ck on yes or no entry
2466 CAB825 00830 JP Z,CFGOFF ;Jump if SYSGEN=OFF
2469 FDCB126E 00840 BIT 5,(IY+'S'-'A') ;Can't sysgen during DO
246D C23724 00850 JP NZ,NOINDO
2470 FDCB0346 00860 BIT 0,(IY+'D'-'A') ;Can't sysgen if spooler
2474 C23324 00870 JP NZ,NOSPL ; is active
2477 FDCB0366 00880 BIT 4,(IY+'D'-'A') ;Can't sysgen if memdisk
247B C22724 00890 JP NZ,NOMDSK ; is active
247E FDCB026E 00900 BIT 5,(IY+'C'-'A') ;CK user sysgen inhibit
2482 C23B24 00910 JP NZ,NOCANDO ;Quit if set
2485 CDF525 00920 CALL CKRTES ;Ck if any route to file
2488 C22B24 00930 JP NZ,NORTES ;Can't if so
00940 ;
00950 ; Initialize a CONFIG file on disk 0, or user spec'ed drive
00960 ;
248B 111B26 00970 LD DE,GENDCB ;"CONFIG/SYS.CCC
248E 210029 00980 LD HL,GENBUF ;Config I/O buffer
2491 0600 00990 LD B,0 ;Set LRL=256
2493 01000 @@INIT ;Create config file
2493 3E3A 00011 LD A,58
2495 EF 00012 RST 40
2496 C21C24 01010 JP NZ,IOERR ;Quit on init error
01020 ;
01030 ; Generate the file header block
01040 ;
2499 3E05 01050 LD A,5 ;Put file header
249B CD1326 01060 CALL PUTOUT
249E 3E06 01070 LD A,6 ;Put header length
24A0 CD1326 01080 CALL PUTOUT
24A3 0606 01090 LD B,6 ;Put header name
24A5 213B26 01100 LD HL,CFGNAM$ ;"CONFIG"
24A8 7E 01110 WRNAM LD A,(HL) ;Write name to
24A9 23 01120 INC HL ; config file header
24AA CD1326 01130 CALL PUTOUT
24AD 10F9 01140 DJNZ WRNAM
01150 ;
01160 ; Generate disabling of all interrupts
01170 ;

```

```

24AF 0601      01180      LD      B,1      ;Set block length
24B1 113800    01190      LD      DE,@RST38 ;Set load address
24B4 219425    01200      LD      HL,C9FLD ;Point to block (X'C9')
24B7 CD9625    01210      CALL    WRBLCK ;Write the block
                01220 ;
                01230 ;      Dump memory from (DVRHI$) to DVREND$
                01240 ;
24BA 2A0602    01250      LD      HL,(DVRHI$) ;Point to max memory
24BD 11F40F    01260      LD      DE,DVREND$ ;Point to start
24C0 AF        01270      XOR      A
24C1 ED52      01280      SBC      HL,DE      ;How much to save
24C3 44        01290      LD      B,H      ; into BC
24C4 4D        01300      LD      C,L
24C5 C45C25    01310      CALL    NZ,DUMP ;File DVREND$ to top
                01320 ;
                01330 ;      Dump memory from HIGH$ to top-of-memory FIRST
                01340 ;
24C8 ED5B1C00 01350      LD      DE,(PHIGH$) ;Point to max memory
24CC 210000    01360      LD      HL,0      ;Set up to get current
24CF 45        01370      LD      B,L      ; HIGH$
24D0          01380      @@HIGH$
24D0 3E64      00013      LD      A,100
24D2 EF        00014      RST      40
24D3 EB        01390      EX      DE,HL      ;HIGH$ to DE,
24D4 AF        01400      XOR      A      ; PHIGH to HL
24D5 ED52      01410      SBC      HL,DE      ;How much to save?
24D7 44        01420      LD      B,H
24D8 4D        01430      LD      C,L
24D9 13        01440      INC      DE      ;First byte in use
24DA C45C25    01450      CALL    NZ,DUMP ;File HIGH$+1 to top
                01460 ;
                01470 ;      Generate all data from address table
                01480 ;
24DD 21A427    01490      LD      HL,ADRTBL$ ;Point to grab table
24E0 4E        01500      LD      C,(HL) ;P/u lo order length
24E1 23        01510      INC      HL
24E2 46        01520      LD      B,(HL) ;P/u hi order length
24E3 23        01530      INC      HL
24E4 78        01540      LD      A,B
24E5 B1        01550      OR      C
24E6 280B      01560      JR      Z,GENDCT ;Exit on zero length
24E8 5E        01570      LD      E,(HL) ;P/u lo order address
24E9 23        01580      INC      HL
24EA 56        01590      LD      D,(HL) ;P/u hi order address
24EB 23        01600      INC      HL
24EC E5        01610      PUSH    HL ;Save table pointer
24ED CD5C25    01620      CALL    DUMP ;P/u the code & file it
24F0 E1        01630      POP     HL ;Restore table pointer
24F1 18ED      01640      JR      WRTBL
                01650 ;
                01660 ;      Generate the DCT$ (offset)
                01670 ;
24F3 0650      01680      LD      B,80 ;Table is 80 bytes
24F5 110043    01690      LD      DE,MOD3BUF ;Place to stuff DCT$
24F8 0E00      01700      LD      C,0
24FA          01710      @@GTDCT ;Write out the DCTs
24FA 3E51      00015      LD      A,81
24FC EF        00016      RST      40
24FD FDE5      01720      PUSH    IY ; handled by SYS0
24FF E1        01730      POP     HL
2500 CD9625    01740      CALL    WRBLCK

```

```

01750 ;
01760 ;      Generate the original interrupt vector JP
01770 ;
2503 0601 01780 INTVEC LD      B,1          ;File the interrupt
2505 113800 01790      LD      DE,0RST38      ; vector address
2508 219525 01800      LD      HL,JPINST      ; with a JP
250B CD9625 01810      CALL     WRBLCK        ;Write the block
01820 ;
01830 ;      Generate the transfer address
01840 ;
250E 3E02 01850      LD      A,2          ;Transfer address
2510 CD1326 01860      CALL     PUTOUT
2513 3E02 01870      LD      A,2
2515 CD1326 01880      CALL     PUTOUT
2518 AF 01890      XOR      A          ;Xfer address is 0000
2519 CD1326 01900      CALL     PUTOUT
251C AF 01910      XOR      A
251D CD1326 01920      CALL     PUTOUT
2520 01930      @@CLOSE          ;Close config file
2520 3E3C 00017      LD      A,60
2522 EF 00018      RST      40
2523 C21C24 01940      JP      NZ,IOERR      ;Quit on close error
01950 ;
01960 ;      Let the config sector know there's a CONFIG/SYS
01970 ;
01980 IF      @MOD2
01990 ;
02000      LD      A,(DRIVE+1)      ;Drive #
02010      LD      C,A          ;Pass drive #
02020      LD      HL,SBUFF$      ;I/O buffer
2526 02030      @@GTDCT          ;Fetch DCT
00019      LD      A,81
00020      RST      40
02040 ;
02050      LD      A,(IY+3)          ;Get DCT data
02060      AND      28H          ;Bit 5/3
02070      CP      20H          ;8" floppy?
02080      JR      NZ,SETSYS1      ;Go if not
02090      LD      A,(IY+4)          ;Get DCT data
02100      AND      50H          ;Bit 6/4
02110      CP      40H          ;DD and not alien?
02120      JR      NZ,SETSYS1      ;Go if not
02130 ;
02140      LD      D,(IY+9)          ;Get GAT address
02150      LD      E,0          ;DE=>gat
2526 02160      @@RDSEC          ;Read sector
00021      LD      A,49
00022      RST      40
02170      CP      6          ;Directory?
02180      JP      NZ,IOERR      ;Go on disk error
02190      LD      A,(SBUFF$+0CDH) ;Get gat data byte
02200      BIT      7,A          ;Data disk?
02210 SETSYS1 LD      D,0          ;Init sys info sector
02220      JR      NZ,$+3          ;Go if not 8" floppy sys
02230      INC      D          ; else sysinfo on cyl 1
02240      LD      E,2          ;Sysinfo sector
02250 ;
02260      ENDIF
02270 ;
2526 21001D 02280      LD      HL,SBUFF$      ;Use this as disk buffer
02290 ;

```

```

2529 110200 02300 IF @MOD4
02310 LD DE,0<8+2 ;Trk 0, sect 2 (SIS)
02320 ENDIF
02330 ;
252C 3A5324 02340 LD A,(DRIVE+1) ;P/u logical drive #
252F 4F 02350 LD C,A ; of drive (DRIVE)
2530 02360 @@RDSEC ;Read System Info Sector
2530 3E31 00023 LD A,49
2532 EF 00024 RST 40
2533 C21C24 02370 JP NZ,IOERR ;Quit on read error
2536 AF 02380 XOR A ;Turn on standard config
2537 32011D 02390 LD (SBUF$+1),A
253A 02400 @@WRSEC ;Write it back
253A 3E35 00025 LD A,53
253C EF 00026 RST 40
253D C21C24 02410 JP NZ,IOERR ;Quit on write error
2540 2EC0 02420 LD L,0C0H ;Pt to SYS/DATA byte
2542 B6 02430 OR (HL) ;Check if System disk
2543 21AD26 02440 LD HL,WARN$ ; and inform in not
2546 2003 02450 JR NZ,$+5
2548 02460 @@LOGOT
00027 IFEQ 00H,1
00028 LD HL,
00029 ENDIF
2548 3E0C 00030 LD A,12
254A EF 00031 RST 40
02470 ;
02480 ; Let user know it's done
02490 ;
254B 02500 @@LOGOT CFGBLT ;Log completion
00032 IFEQ 01H,1
254B 214126 00033 LD HL,CFGBLT
00034 ENDIF
254E 3E0C 00035 LD A,12
2550 EF 00036 RST 40
2551 215A26 02510 LD HL,ATRBIT ;Make CONFIG invisible
2554 3E18 02520 LD A,24 ;Init for CMNDI
2556 C32800 02530 JP RST28 ; & execute
2559 02540 @@CMNDI
2559 3E18 00037 LD A,24
255B EF 00038 RST 40
02550 ;
02560 ; Routine to DUMP core
02570 ;
255C EB 02580 DUMP EX DE,HL ;Load address to HL
255D 111B26 02590 LD DE,GENDCB ;Config/sys FCB
2560 E5 02600 DUMP1 PUSH HL
2561 C5 02610 PUSH BC ;Save block length
2562 60 02620 LD H,B ;Length of this block
2563 69 02630 LD L,C ; to HL
2564 01FE00 02640 LD BC,254 ;Write block in groups
2567 AF 02650 XOR A ;Of 254 bytes max
2568 ED42 02660 SBC HL,BC ;Sub blk len fm dump len
256A 3006 02670 JR NC,DUMP2 ;Go if write len > 254
256C C1 02680 POP BC ;Recover block len
256D 210000 02690 LD HL,0 ;Show block written
2570 1801 02700 JR DUMP3
2572 F1 02710 DUMP2 POP AF ;Remove old len from stack
2573 E3 02720 DUMP3 EX (SP),HL ;Xchg len w/address
2574 41 02730 LD B,C
2575 3E01 02740 LD A,1 ;Start of block

```

```

2577 CD1326 02750 CALL PUTOUT
257A 78 02760 LD A,B
257B C602 02770 ADD A,2 ;Add for address
257D CD1326 02780 CALL PUTOUT ;Write block length
2580 7D 02790 LD A,L
2581 CD1326 02800 CALL PUTOUT ;Write lo order address
2584 7C 02810 LD A,H
2585 CD1326 02820 CALL PUTOUT ;Write hi order address
2588 7E 02830 DUMP4 LD A,(HL)
2589 23 02840 INC HL
258A CD1326 02850 CALL PUTOUT ;Write the block
258D 10F9 02860 DJNZ DUMP4
258F C1 02870 POP BC ;Get remaining length
2590 78 02880 LD A,B
2591 B1 02890 OR C
2592 20CC 02900 JR NZ,DUMP1 ;Loop if block not done
2594 C9 02910 C9FLD RET ; else go back
02920 ;
02930 ; Routine to write an offset block
02940 ;
2595 C3 02950 JPINST DB 0C3H ;JP instruction
2596 E5 02960 WRBLCK PUSH HL ;Save real load address
2597 62 02970 LD H,D ;HL = offset address
2598 6B 02980 LD L,E
2599 11B26 02990 LD DE,GENDCB ;Config/sys FCB
259C 3E01 03000 LD A,1 ;Start of block
259E CD1326 03010 CALL PUTOUT
25A1 78 03020 LD A,B
25A2 C602 03030 ADD A,2 ;Adj for address
25A4 CD1326 03040 CALL PUTOUT ;Write block length
25A7 7D 03050 LD A,L
25A8 CD1326 03060 CALL PUTOUT ;Write lo order load
25AB 7C 03070 LD A,H
25AC CD1326 03080 CALL PUTOUT ;Write hi order load
25AF E1 03090 POP HL ;Recover real address
25B0 7E 03100 WRBLK1 LD A,(HL) ; and p/u data from there
25B1 23 03110 INC HL
25B2 CD1326 03120 CALL PUTOUT ;Write the block
25B5 10F9 03130 DJNZ WRBLK1
25B7 C9 03140 RET
03150 ;
03160 ; Perform SYSGEN OFF
03170 ;
25B8 21001D 03180 CFGOFF LD HL,SBUFF$ ;Set disk I/O buffer
03190 ;
03200 IF @MOD2
03210 ;
03220 LD A,(DRIVE+1) ;Get drive
03230 LD C,A ;Pass to C
25BB 03240 @@GTDCT ;Fetch DCT
00039 LD A,81
00040 RST 40
03250 LD D,(IY+9) ;Dir cylinder
03260 ;
03270 LD A,(IY+3) ;Get dct data
03280 AND 28H ;Bit 5/3
03290 CP 20H ;8" floppy?
03300 JR NZ,SETSYS2 ;Go if not
03310 LD A,(IY+4) ;Get data
03320 AND 50H ;Bit 6/4
03330 CP 40H ;DD not alien?

```



```

03340 JR NZ,SETSYS2 ;Go if not
03350 ;
03360 LD E,0 ;GAT table
25BB 03370 @RDSEC ;Read sector
03400 LD A,49
03420 RST 40
03430 CP 6 ;Directory read?
03440 JP NZ,IOERR ;Go if not
03450 LD A,(SBUFF$+0CDH) ;Get gat data byte
03460 BIT 7,A ;System disk?
03470 SETSYS2 LD D,0 ;Cyl 0 if not
03480 JR NZ,$+3 ;Go if not
03490 INC D ;Else on cyl 1
03500 LD E,2 ;Sysinfo sector
03510 ;
03520 ;
03530 IF @MOD4
25BB 110200 LD DE,0<8+2 ;Trk 0, sect 2 (SIS)
03540 ENDIF
03550 ;
25BE 3A5324 LD A,(DRIVE+1) ;P/u drive number
25C1 4F 03560 LD C,A ; & note
25C2 03570 @RDSEC ; that no config is
25C3 3E31 03580 LD A,49
25C4 EF 03590 RST 40
25C5 C21C24 03600 JP NZ,IOERR ; on the disk
25C8 3EC9 03610 LD A,0C9H
25CA 32011D 03620 LD (SBUFF$+1),A ;A X'C9' indicates
25CD 03630 @WRSEC ; no config
25CD 3E35 03640 LD A,53
25CF EF 03650 RST 40
25D0 C21C24 03660 JP NZ,IOERR ;Quit if write error
03670 ;
03680 ; Now that none is shown, kill the CONFIG/SYS
03690 ;
25D3 111B26 03700 LD DE,GENDCB ;Pt to FCB
25D6 0600 03710 LD B,0
25D8 03720 @OPEN ;Try to open the config
25D8 3E3B 03730 LD A,59
25DA EF 03740 RST 40
25DB C22F24 03750 JP NZ,NOTFND ;Jump if not there
25DE 03760 @REMOV ;Kill the config/sys
25DE 3E39 03770 LD A,57
25E0 EF 03780 RST 40
25E1 C21C24 03790 JP NZ,IOERR ;Quit if can't be killed
25E4 03800 @LOGOT CFGDEL$ ;"config deleted..
03810 IFEQ 01H,1
25E4 217626 03820 LD HL,CFGDEL$
03830 ENDIF
25E7 3E0C 03840 LD A,12
25E9 EF 03850 RST 40
25EA C9 03860 RET ;Done - return
03870 ;
03880 ; Parameter parsing of yes/no
03890 ;
25EB 010000 03900 NPARM LD BC,0 ;P/u "no" parm entry
25EE 78 03910 LD A,B
25EF B1 03920 OR C ;If NO, make Z
25F0 EEFF 03930 XOR 0FFH
25F2 C9 03940 RET

```

```

25F3 0000      03800 YPARM   DW      0
                03810 ;
                03820 ;      Any devices routed to files?
                03830 ;

25F5 114B49    03840 CKRTES  LD      DE,'IK'      ;Point to begin of area
25F8           03850          @@GTDCB
25F8 3E52      00056          LD      A,82
25FA EF        00057          RST     40
25FB E5        03860 CKRT0   PUSH    HL           ;Save pointer
25FC CB66      03870 CKRT1   BIT     4,(HL)       ;Routed device?
25FE 280B      03880          JR      Z,CKRT2     ;Jump if not
2600 2C        03890          INC     L           ;Bypass TYPE code
2601 7E        03900          LD      A,(HL)      ;P/u route vector
2602 2C        03910          INC     L           ; lo-order
2603 66        03920          LD      H,(HL)      ;P/u vector hi-order
2604 6F        03930          LD      L,A
2605 CB7E      03940          BIT     7,(HL)      ;Routed to a file?
2607 28F3      03950          JR      Z,CKRT1     ;Ck further route if not
2609 E1        03960          POP     HL          ; else exit with NZ
260A C9        03970          RET
                03980 ;
                03990 ;      Point to next device
                04000 ;

260B E1        04010 CKRT2   POP     HL           ;Recover DCB pointer
260C 7D        04020          LD      A,L         ;Advance to next DCB area
260D C608      04030          ADD     A,8         ;Loop through all
260F 6F        04040          LD      L,A         ; devices while checking
2610 20E9      04050          JR      NZ,CKRT0    ;Loop until table end
2612 C9        04060          RET
                04070 ;

2613 4F        04080 PUTOUT  LD      C,A         ;Byte into C
2614           04090          @@PUT             ;Do the put
2614 3E04      00058          LD      A,4
2616 EF        00059          RST     40
2617 C8        04100          RET     Z           ;Back if good
2618 C31C24    04110          JP      IOERR       ; else abort
                04120 ;

261B 43        04130 GENDCB  DB      'CONFIG/SYS.CCC:0',0
          4F 4E 46 49 47 2F 53 59
          53 2E 43 43 43 3A 30 00
262A           04140 CFGSPEC EQU    $-2
000F           04150          DS      32-$(GENDCB)
263B 43        04160 CFGNAM$ DB      'CONFIG'
          4F 4E 46 49 47
2641 55        04170 CFGBLT  DB      'User configuration built',CR
          73 65 72 20 63 6F 6E 66
          69 67 75 72 61 74 69 6F
          6E 20 62 75 69 6C 74 0D
265A 41        04180 ATRBIT  DB      'Attrib CONFIG/SYS.CCC:d (I)',CR
          74 74 72 69 62 20 43 4F
          4E 46 49 47 2F 53 59 53
          2E 43 43 43 3A 64 20 28
          49 29 0D
2670           04190 ATRSPEC EQU    $-6
2676 55        04200 CFGDEL$ DB      'User configuration deleted',CR
          73 65 72 20 63 6F 6E 66
          69 67 75 72 61 74 69 6F
          6E 20 64 65 6C 65 74 65
          64 0D
2691 4E        04210 NOTFND$ DB      'No user configuration found',CR
          6F 20 75 73 65 72 20 63

```

```

6F 6E 66 69 67 75 72 61
74 69 6F 6E 20 66 6F 75
6E 64 0D
26AD 57      04220 WARN$  DB      'Warning: Target drive contains no system',CR
61 72 6E 69 6E 67 3A 20
54 61 72 67 65 74 20 64
72 69 76 65 20 63 6F 6E
74 61 69 6E 73 20 6E 6F
20 73 79 73 74 65 6D 0D
26D6 43      04230 NOINDO$ DB      'Command invalid during <D0> processing',CR
6F 6D 6D 61 6E 64 20 69
6E 76 61 6C 69 64 20 64
75 72 69 6E 67 20 3C 44
4F 3E 20 70 72 6F 63 65
73 73 69 6E 67 0D
26FD 2A      04240 NOCANDO$ DB      '** SYSGEN inhibited at this time **',CR
2A 20 53 59 53 47 45 4E
20 69 6E 68 69 62 69 74
65 64 20 61 74 20 74 68
69 73 20 74 69 6D 65 20
2A 2A 0D
2721 43      04250 NOSPL$  DB      'Can''t while SPOOL is active',CR
61 6E 27 74 20 77 68 69
6C 65 20 53 50 4F 4F 4C
20 69 73 20 61 63 74 69
76 65 0D
273D 43      04260 NOMDSK$ DB      'Can''t while MEMDISK is active',CR
61 6E 27 74 20 77 68 69
6C 65 20 4D 45 4D 44 49
53 4B 20 69 73 20 61 63
74 69 76 65 0D
275B 43      04270 NORTES$ DB      'Can''t while route-to-file is active',CR
61 6E 27 74 20 77 68 69
6C 65 20 72 6F 75 74 65
2D 74 6F 2D 66 69 6C 65
20 69 73 20 61 63 74 69
76 65 0D
277F 80      04280 ;
      04290 PRMTBL$ DB      80H,42H,'ON',0
42 4F 4E 00
2784 F325     04300      DW      YPARM
2786 43      04310      DB      43H,'OFF',0
4F 46 46 00
278B EC25     04320      DW      NPARM+1
278D 53      04330      DB      53H,'YES',0
59 45 53 00
2792 F325     04340      DW      YPARM
2794 52      04350      DB      52H,'NO',0
4E 4F 00
2798 EC25     04360      DW      NPARM+1
279A 95      04370      DB      95H,'DRIVE',0
44 52 49 56 45 00
27A1 5324     04380      DW      DRIVE+1
27A3 00      04390      NOP
      04400 ;
      04410 ;      Table of regions to sysgen
      04420 ;
27A4 0200     04430 ADRTBL$ DW      2,HIGH$      ;Save HIGH$
      0E04
27A8 0200     04440      DW      2,LOW$      ;Lowest user address
      1E00

```

The Source	LIBRARY Files	SYSGEN - LS-DOS 6.2	Page 00010
27AC 0100 2B00	04450 DW	1,TIMSL\$;Time slice	
27B0 FA00 0602	04460 DW	250,DVRHI\$;Save primary DCBs	
27B4 0300 A419	04470 DW	3,EXTDBG\$;Ext DEBUG vector & DBGHK	
27B8 3200 3C00	04480 DW	50,INTIM\$;Table & TCBs	
27BC 2200 6A00	04490 DW	34,FLGTAB\$;Flag table & assorted	
27C0 0200 9601	04500 DW	2,75*2+SVCTAB\$;Save WRITE vector	
27C4 0300 6C1A	04510 DW	3,HKRES\$;Sys overlay hook	
27C8 0200 FE08	04520 DW	2,KIDATA\$+2	
27CC 0100 940B	04530 DW	1,DODATA\$	
27D0 0200 970B	04540 DW	2,DODATA\$+3	
27D4 1000 F001	04550 DW	16,240+SVCTAB\$;SVCs 120 - 127	
27D8 1000 0800	04560 DW	16,@RST08 ;RST zones 8 and 10	
27DC 0100 5007	04570 DW	1,HERTZ\$;Hertz rate for timer	
	04580 ;		
	04590 IF	@MOD2	
	04600 DW	3,\$CRSCHAR ;Cursor char + column siz	
	04610 ENDIF		
	04620 IF	@MOD4	
27E0 0000 0000	04630 DW	0,0	
	04640		
	04650 ;		
27E4 0000 0000	04660 DW	0,0	
27E8 0000 0000	04670 DW	0,0	
27EC 0000 0000	04680 DW	0,0	
27F0 0000 0000	04690 DW	0,0	
27F4 0000 0000	04700 DW	0,0	
27F8 0000 0000	04710 DW	0,0	
27FC 0000 0000	04720 DW	0,0	
2800 0000	04730 DW	0 ;End of table	
2802 00	04740 DC	20,0 ;Patch space	
00 00 00 00 00 00 00 00			
00 00 00 00 00 00 00 00			
00 00 00			
	04750 ;		
2900 04760	ORG	\$<-8+1<+8	
2900 04770	GENBUF EQU	\$	
	04780 ;		
2400 04790	END	SYSGEN	

\$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 @LOGGER	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 @RWRTT	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 ADRTBL\$	27A4
AFLAG\$	006A ATRBIT	265A ATRSPEC	2670
AUTO?	1FF1 BAR\$	0201 BEGINA	2409
BOOTST\$	439D BREAK?	1C60 BRKVEC\$	1C88
BUR\$	0200 C9FLD	2594 CASHK\$	0A7B
CFCB\$	00E0 CFGBLT	2641 CFGDEL\$	2676
CFGFCB\$	00E0 CFGNAM\$	263B CFGOFF	25B8
CFGSPEC	262A CFLAG\$	006C CKMOD0	1A7F
CKOPEN0	1568 CKRT0	25FB CKRT1	25FC
CKRT2	260B CKRTES	25F5 CONF IG\$	203F
CORE\$	0300 CR	000D CRTBGN\$	F800
CYL GRN	16AE D@FBYT8	1A26 DATE\$	0033
DAYTBL\$	04C7 DBGSV\$	00A0 DCBKL\$	0031
DCT\$	0470 DCTBYT80	1A29 DCTFLD0	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
DRIVE	2452	DSKTYP\$	04C0	DTPMT\$	04C2
DUMP	255C	DUMP1	2560	DUMP2	2572
DUMP3	2573	DUMP4	2588	DVREND\$	0FF4
DVRHI\$	0206	EFLAG\$	006E	ENADIS DO RAM	0817
EXIT	2410	EXTDBG\$	19A4	FDDINT\$	000E
FEMSK\$	006F	FLGTAB\$	006A	GENBUF	2900
GENDCB	261B	GENDCT	24F3	GET @ ROWCOL	0DAE
HERTZ\$	0750	HIGH\$	040E	HKRES\$	1A6C
IFLAG\$	0072	INBUF\$	0420	INTIM\$	003C
INTMSK\$	003D	INTVC\$	003E	INTVEC	2503
IOERR	241C	JLCB\$	0203	JDCB\$	0024
JFCB\$	00C0	JLDCB\$	0230	JPINST	2595
JRET\$	0026	KCK@	07D6	KFLAG\$	0074
KIDATA\$	08FC	KIDCB\$	0208	LBANK\$	0202
LDRV\$	0023	LFLAG\$	0075	LNKFCB@	1566
LOW\$	001E	LSVC\$	000D	MAXCOR\$	2400
MAXDAY\$	0401	MINCOR\$	3000	MOD3BUF	4300
MODOUT\$	0076	MONTBL\$	04DC	NFLAG\$	0077
NOCANDO	243B	NOCANDO\$	26FD	NOINDO	2437
NOINDOS\$	26D6	NOMDSK	2427	NOMDSK\$	273D
NORTES	242B	NORTES\$	275B	NOSPL	2433
NOSPL\$	2721	NOTFND	242F	NOTFND\$	2691
NPARAM	25EB	OPREG\$	0078	OPREG SV_AREA	086E
OPREG SV_PTR	0835	ORARET@	14DC	OSRLS\$	003B
OSVER\$	0085	OVRLY\$	0069	PAKNAM\$	0410
PAR_ERR	002C	PAUSE@	0382	PCSAVE\$	07AF
PDRV\$	001B	PHIGH\$	001C	PRDCB\$	0218
PRMERR	241A	PRMTBL\$	277F	PUTA@DE	0DCD
PUTOUT	2613	PUT @	0DCA	PUT @ ROWCOL	0DC6
RFLAG\$	007B	ROWCOL_2_ADDR	0DD0	RST28	0028
RST38@	1BFF	RSTOR\$	04C4	RWRIT@	13A2
SIDCB\$	0238	SAVESP	2413	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	SYSERR\$	1B13
SYSGEN	2400	SYSGEN1	2446	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	WARN\$	26AD
WRBLCK	2596	WRBLK1	25B0	WRINT\$	0080
WRNAM	24A8	WRTBL	24E0	YPARM	25F3
ZERO\$	0401	ZEROA@	13A0	@@ABORT	9749
@@ADTSK	97DC	@@BANK	9CF4	@@BKSP	99D4
@@BREAK	9D0A	@@CHNIO	9734	@@CKBRKC	9D58
@@CKDRV	9830	@@CKEOF	99E9	@@CKTSK	97C7
@@CLOSE	99BF	@@CLS	9D42	@@CMNDI	9773
@@CMNDR	9788	@@CTL	9598	@@DATE	970A
@@DCSTAT	986F	@@DEBUG	97B2	@@DECHEX	9C74
@@DIRRD	9BE1	@@DIRWR	9BF6	@@DIV16	9C5F
@@DIV8	9C4A	@@DODIR	9845	@@DSP	955C
@@DSPLY	95FC	@@ERROR	979D	@@EXIT	975E
@@FEXT	9B4E	@@FLAGS	9CDE	@@FNAME	9B63
@@FSPEC	9B39	@@GATRD	9BCC	@@GATWR	9C0B
@@GET	9570	@@GTDCB	9B8D	@@GTDC	9B78

@@GTMOD	9BA2 @@HDFMT	9917 @@HEX16	9CB3
@@HEX8	9C9E @@HEXDEC	9C89 @@HIGH\$	9CC8
@@INIT	9995 @@KBD	95D4 @@KEY	9548
@@KEYIN	95E8 @@KLTSK	981B @@LOAD	9B0F
@@LOC	99FE @@LOF	9A13 @@LOGGER	9633
@@LOGOT	9648 @@MSG	967F @@MUL16	9C35
@@MUL8	9C20 @@OPEN	99AA @@PARAM	96F5
@@PAUSE	96E0 @@PEOF	9A28 @@POSN	9A3D
@@PRINT	9694 @@PRT	95AC @@PUT	9584
@@RAMDIR	985A @@RDSEC	98ED @@RDSSC	9BB7
@@READ	9A52 @@REMOV	9980 @@RENAM	996B
@@REW	9A67 @@RMTSK	97F1 @@RPTSK	9806
@@RREAD	9A7C @@RSLCT	98D8 @@RSTOR	9899
@@RUN	9B24 @@RWRT	9A91 @@SEEK	98C3
@@SEEKSC	9AA6 @@SKIP	9ABB @@SLCT	9884
@@STEPI	98AE @@TIME	971F @@VDCTL	96CB
@@VER	9AD0 @@VRSEC	9902 @@WEOF	9AE5
@@WHERE	95C0 @@WRITE	9AFA @@WRSEC	992C
@@WRSSC	9941 @@WRTRK	9956	

2400 is the transfer address

00000 Total errors

NOTES:

NOTES:

Command: SYSTEM

Library: SYS8/SYS

ISAM # : A1H

```

0000 00100 *GET LBSYSTEM:3
0001 00010 ;LBSYSTEM/ASM - SYSTEM Command
0000 00020 TITLE <SYSTEM - LS-DOS 6.2>
0003 00030 ;
00EC 00040 MODPORT EQU 0ECH
000D 00050 CR EQU 13
0006 00060 ;
0000 00070 *GET SVCMAC:3 ;SVC Macro equivalents
0008 00080 ;SVMAC/ASM - LS-DOS Version VI
0009 00090 *LIST OFF
0397 03970 *LIST ON
0399 03990 *LIST OFF ;Get SYS0/EQU
0701 07010 *LIST ON
0702 07020 ;
2400 07030 ORG 2400H
0704 07040 ;
0705 07050 SYSTEM
2400 07060 @@CKBRKC ;Check for <BREAK>
2400 3E6A 00001 LD A,106
2402 EF 00002 RST 40
2403 2825 07070 JR Z,SYSTEM1 ;Go if not
2405 21FFFF 07080 LD HL,-1 ; else abort
2408 C9 07090 RET
0710 07100 ;
2409 310000 07110 SAVESP LD SP,$-$ ;P/u original stack
240C 07120 @@CKBRKC ;Clear any <BREAK>
240C 3E6A 00003 LD A,106
240E EF 00004 RST 40
240F C9 07130 RET ;Ret to DOS
0714 07140 ;
0715 07150 ; Internal Error Message Handling
0716 07160 ;
2410 217C29 07170 NOMEM LD HL,NOMEM$ ;"No memroy available
2413 DD 07180 DB 0DDH
2414 219629 07190 PRMERR LD HL,PRMERR$ ;"Parm error
2417 07200 SETERR @@LOGOT
00005 00005 IFEQ 00H,1
00006 00006 LD HL,
00007 00007 ENDIF
2417 3E0C 00008 LD A,12
2419 EF 00009 RST 40
241A 21FFFF 07210 FRCERR LD HL,-1 ;Set abort code
241D 18EA 07220 JR SAVESP ;Exit
0723 07230 ;
0724 07240 ; I/O Error Handler
0725 07250 ;
241F 6F 07260 IOERR LD L,A ;Error # to HL
2420 2600 07270 LD H,0
2422 F6C0 07280 OR 0C0H ;Set brief to return
2424 4F 07290 LD C,A
2425 07300 @@ERROR ;Display the error
2425 3E1A 00010 LD A,26
2427 EF 00011 RST 40
2428 18DF 07310 JR SAVESP ;Exit
0732 07320 ;
0733 07330 ; SYSTEM1 - Set the O/S's parameters
0734 07340 ;
0735 07350 SYSTEM1
242A ED730A24 07360 LD (SAVESP+1),SP ;Save the stack
0737 07370 ;
242E 11A32A 07380 LD DE,PRMTBL$ ;Get parms

```

```

2431      07390      @@PARAM
2431 3E11      00012      LD      A,17
2433 EF      00013      RST      40
2434 C21424    07400      JP      NZ,PRMERR      ;Jump on parm error
2437      07410      @@FLAGS      ;Get IY = flagtab
2437 3E65      00014      LD      A,101
2439 EF      00015      RST      40
243A 010000    07420 SLOW  LD      BC,0      ;Test SLOW parm
243D 78      07430      LD      A,B
243E B1      07440      OR      C
243F 2013      07450      JR      NZ,SETSLOW    ;Jump if SLOW entered
2441 010000    07460 FPARM LD      BC,0      ;P/u FAST parm
2444 78      07470      LD      A,B
2445 B1      07480      OR      C
2446 281D      07490      JR      Z,FPEND      ;Neither SLOW nor FAST?
2448 FDCB12DE  07500      SET     3,(IY+'S'-'A') ;Set fast in flag
                07510      IF      @MOD4
244C FDCB0CF6  07520      SET     6,(IY+'M'-'A') ;Set bit 6 in port mask
                07530      ENDIF
2450 0655      07540      LD      B,55H      ;Init time slice byte
2452 1808      07550      JR      SETCLK      ;Go set it
                07560      ;
                07570      ;      Turn off FAST
                07580      ;
2454 FDCB129E  07590 SETSLOW RES     3,(IY+'S'-'A') ;Reset the FAST bit
                07600      IF      @MOD4
2458 FDCB0CB6  07610      RES     6,(IY+'M'-'A') ;Reset bit in port mask
                07620      ENDIF
245C 78      07630 SETCLK LD      A,B
245D 322B00    07640      LD      (TIMSL$),A      ;Set 55H=fast, FFH=slow
                07650      IF      @MOD4
2460 FD7E0C    07660      LD      A,(IY+'M'-'A') ;Update the port
2463 D3EC      07670      OUT     (MODPORT),A
                07680      ENDIF
2465      07690 FPEND  EQU      $
                07700      ;
                07710      ;      Trace entry
                07720      ;
2465 010100    07730 TRACE LD      BC,1      ;Init to pass by
2468 78      07740      LD      A,B
2469 B1      07750      OR      C
246A 280D      07760      JR      Z,TROFF      ;Go if Trace off
246C 3C      07770      INC     A
246D 200F      07780      JR      NZ,TREND     ;Go if not entered
                07790      IF      @MOD4
246F 11B107    07800      LD      DE,TRACE_INT ;Pt to trace SYSRES rtn
                07810      ENDIF
                07820      IF      @MOD2
                07830      LD      DE,DO_TRACE ;Address
                07840      ENDIF
2472 0E07      07850      LD      C,7      ;Init to task 7
2474      07860      @@ADTSK      ;Turn TRACE (ON)
2474 3E1D      00016      LD      A,29
2476 EF      00017      RST      40
2477 1805      07870      JR      TREND
2479 0E07      07880 TROFF LD      C,7      ;Init to task 7
247B      07890      @@RMTSK      ;Remove TRACE task
247B 3E1E      00018      LD      A,30
247D EF      00019      RST      40
247E      07900 TREND  EQU      $
                07910      ;

```

```

07920 ;      Routine to handle BREAK
07930 ;
247E 010100 07940 BREAK LD      BC,1      ;Init BREAK=ON
2481 78      07950 LD      A,B
2482 B1      07960 OR      C
2483 2809    07970 JR      Z,BRKOFF      ;Jump if BREAK=OFF
2485 3C      07980 INC     A      ;Check Break=ON
2486 200A    07990 JR      NZ,BRKEND      ;Go if no parm used
2488 FDCB12A6 08000 RES     4,(IY+'S'-'A') ;Correct system flag
248C 1804    08010 JR      BRKEND
08020 ;
08030 ;      Disable <BREAK> key
08040 ;
248E FDCB12E6 08050 BRKOFF SET     4,(IY+'S'-'A')
2492      08060 BRKEND EQU      $
08070 ;
08080 ;      Date prompt suppression during BOOT
08090 ;
2492 010100 08100 DATE LD      BC,1      ;Init to pass by
2495 78      08110 LD      A,B
2496 B1      08120 OR      C
2497 06FF    08130 LD      B,0FFH      ;Init to DATE Suppress
2499 2804    08140 JR      Z,DOFF      ;Go if suppress
249B 3C      08150 INC     A
249C 200B    08160 JR      NZ,DATEND      ;Go if not entered
249E 47      08170 LD      B,A      ;B=0 on no suppress
249F CD5F29 08180 DOFF CALL   GETCFG      ;Get track 0, sector 2
24A2 78      08190 LD      A,B      ;Get back the code byte
24A3 32C21D 08200 LD      (DTPMT$&0FFH+SBUFF$),A
24A6 CD5C29 08210 CALL   PUTCFG      ;Write it back
24A9      08220 DATEND EQU      $
08230 ;
08240 ;      Time prompt suppression during BOOT
08250 ;
24A9 010100 08260 TIME LD      BC,1      ;Init to pass by
24AC 78      08270 LD      A,B
24AD B1      08280 OR      C
24AE 06FF    08290 LD      B,0FFH      ;Init to TIME Suppress
24B0 2804    08300 JR      Z,TOFF      ;Go if suppress
24B2 3C      08310 INC     A
24B3 200B    08320 JR      NZ,TIMEND      ;Go if not entered
24B5 47      08330 LD      B,A      ;B=0 on no suppress
24B6 CD5F29 08340 TOFF CALL   GETCFG      ;Get track 0, sector 2
24B9 78      08350 LD      A,B      ;Get back the code byte
24BA 32C31D 08360 LD      (TMPMT$&0FFH+SBUFF$),A
24BD CD5C29 08370 CALL   PUTCFG      ;Write it back
24C0      08380 TIMEND EQU      $
08390 ;
08400 ;      BOOT Step rate adjust
08410 ;
24C0 01FFFF 08420 BSPARM LD      BC,-1      ;P/u parm
24C3 04      08430 INC     B      ;User entry?
24C4 281B    08440 JR      Z,BSPEND      ;Go if not entered
24C6 79      08450 LD      A,C      ;Transfer parm
24C7 FE04    08460 CP      4
24C9 3808    08470 JR      C,GUDBS      ;Ok if < 4
24CB 21832A 08480 LD      HL,BADBS$      ; else bad step rate
24CE CD1724 08490 CALL   SETERR
24D1 180E    08500 JR      BSPEND
24D3 47      08510 GUDBS LD      B,A      ;Save step
24D4 CD5F29 08520 CALL   GETCFG      ;Get System Info Sector

```

```

24D7 2E73      08530      LD      L,70H+3      ;Offset to data
24D9 7E        08540      LD      A,(HL)      ;P/u DCT+3 parms
24DA E6FC      08550      AND     0FCH      ;Strip off boot step
24DC B0        08560      OR      B          ;Merge new
24DD 77        08570      LD      (HL),A      ;Stuff back
24DE CD5C29    08580      CALL   PUTCFG      ;Write it back
24E1           08590      BSPEND  EQU      $
                08600      ;
                08610      ;      BLINK parm
                08620      ;

24E1 010100    08630      BLINK  LD      BC,1      ;P/U parm
24E4 78        08640      LD      A,B
24E5 B1        08650      OR      C
24E6 3D        08660      DEC     A          ;Specified?
24E7 2829      08670      JR      Z,BLIEND      ;Go if not
24E9 3C        08680      INC     A
24EA 2006      08690      JR      NZ,BLNKON      ;Go if BLINK=ON
24EC FDCB15F6  08700      SET     6,(IY+'V'-'A') ;Turn off blinking
                08710      IF      @MOD2
                08720      LD      A,($CRSCHAR) ;Get current char
                08730      AND     1FH      ;Keep size only
                08740      JR      SETBLK      ;Setup
                08750      ENDIF
                08760      IF      @MOD4
24F0 1820      08770      JR      BLIEND      ;Continue
                08780      ENDIF

24F2 FDCB15B6  08790      BLNKON RES     6,(IY+'V'-'A') ;Turn on blinking
24F6 FEFF      08800      CP      0FFH      ;If not just BLINK,
                08810      IF      @MOD4
24F8 2012      08820      JR      NZ,SETBLK      ; set user cursor
                08830      ENDIF
                08840      IF      @MOD2
                08850      JR      Z,LARGE      ;Go if not
                08860      BLIADJ  SUB     7          ;Force in range 0-7
                08870      JR      NC,BLIADJ      ;Go till in range
                08880      ADD     A,7          ;Add back last sub
                08890      OR      01000000B      ;Set blink enable 1/16
                08900      JR      SETBLK      ;Continue
                08910      ENDIF

24FA 010000    08920      LARGE  LD      BC,0      ; else test if large,
24FD 0C        08930      INC     C          ; small, or default
                08940      IF      @MOD2
                08950      LD      A,01000111B      ;Standard cursor
                08960      ENDIF
                08970      IF      @MOD4
24FE 3E5F      08980      LD      A,'_'      ;Standard cursor
                08990      ENDIF

2500 2002      09000      JR      NZ,SMALPRM
                09010      IF      @MOD2
                09020      LD      A,01000000B      ;To large
                09030      ENDIF
                09040      IF      @MOD4
2502 3E8F      09050      LD      A,8FH      ;To large
                09060      ENDIF

2504 010000    09070      SMALPRM LD      BC,0
2507 0C        09080      INC     C
2508 2002      09090      JR      NZ,SETBLK
                09100      IF      @MOD2
                09110      LD      A,01000101B      ;Small
                09120      ENDIF
                09130      IF      @MOD4

```

```

250A 3E88      09140      LD      A,88H      ;Small
                09150      ENDIF
250C 4F        09160 SETBLK LD      C,A      ;Xfer cursor char to C
250D 0608      09170      LD      B,8      ;Cursor update function
250F           09180      @@VDCTL
250F 3E0F      09020      LD      A,15
2511 EF        09021      RST      40
2512           09190 BLIEND EQU      $
                09200 ;
                09210 ;      RESTORE parameter
                09220 ;
2512 010100    09230 RPARM LD      BC,1      ;P/U parm
2515 78        09240      LD      A,B
2516 B1        09250      OR      C
2517 3D        09260      DEC     A      ;Specified?
2518 280F      09270      JR      Z,REND      ;Go if not
251A 3C        09280      INC     A
251B 2001      09290      JR      NZ,RSTRON      ;Go if RESTORE=on
251D 3E        09300      DB      3EH      ;Make LD A,n
251E AF        09310 RSTRON XOR     A
251F 47        09320      LD      B,A      ;Save step
2520 CD5F29    09330      CALL   GETCFG      ;Get config sector
2523 2EC4      09340      LD      L,RSTOR$&0FFH ;Pt to RESTORE flag
2525 70        09350      LD      (HL),B      ;Stuff back
2526 CD5C29    09360      CALL   PUTCFG      ;Put it back
2529           09370 REND EQU      $
                09380 ;
                09390 ;      Type ahead processing
                09400 ;
2529 010100    09410 TYPE LD      BC,1      ;P/u Type parm value
252C 78        09420      LD      A,B
252D B1        09430      OR      C
252E 3D        09440      DEC     A      ;Was Type used?
252F 280D      09450      JR      Z,TYPEND      ;Jump if TYPE not entered
2531 3C        09460      INC     A
2532 2006      09470      JR      NZ,TYPEON      ;Jump if TYPE=ON
2534 FDCB038E  09480      RES     1,(IY+'D'-'A') ;Turn TYPE off
2538 1804      09490      JR      TYPEND
253A FDCB03CE  09500 TYPEON SET     1,(IY+'D'-'A') ;Turn TYPE on
253E           09510 TYPEND EQU     $
                09520 ;
                09530 ;      Process SMOOTH
                09540 ;
253E 010100    09550 SMOOTH LD     BC,1      ;P/u Smooth parm value
2541 78        09560      LD      A,B
2542 B1        09570      OR      C
2543 3D        09580      DEC     A      ;Was Smooth used?
2544 2817      09590      JR      Z,SMEND      ;Skip if not entered
2546 3C        09600      INC     A
2547 200B      09610      JR      NZ,SMON      ;Go if Smooth=ON
                09620      IF      @MOD4
2549 FDCB039E  09630      RES     3,(IY+'D'-'A') ;Show OFF in DFLAG$
254D 3E00      09640      LD      A,00      ;Set to store NOP
254F 320E00    09650      LD      (FDDINT$),A ; put it there
                09660      ENDIF
2552 1809      09670      JR      SMEND
2554           09680 SMON EQU     $
                09690      IF      @MOD4
2554 FDCB03DE  09700      SET     3,(IY+'D'-'A') ;Show ON in DFLAG$
2558 3EF3      09710      LD      A,0F3H      ;DI opcode
255A 320E00    09720      LD      (FDDINT$),A ; put it there

```

```

09730      ENDIF
255D      09740 SMEND EQU $
          09750 ;
          09760 ; Routine to process HERTZ selection
          09770 ;
255D 010100 09780 HERTZ50 LD BC,1 ;P/u H50 parm value
2560 78      09790 LD A,B
2561 B1      09800 OR C
2562 3D      09810 DEC A ;Check if parm entered
2563 2805    09820 JR Z,HZ50END ;Go if not
2565 3E19    09830 LD A,25 ;Init 25 ints/sec
2567 325007 09840 LD (HERTZ$),A ;Put value in timer code
256A        09850 HZ50END EQU $
256A 010100 09860 HERTZ60 LD BC,1 ;P/u H60 parm value
256D 78      09870 LD A,B
256E B1      09880 OR C
256F 3D      09890 DEC A ;Check if parm entered
2570 2805    09900 JR Z,HZ60END ;Go if not
2572 3E1E    09910 LD A,30 ;Init 30 ints/sec
2574 325007 09920 LD (HERTZ$),A ; & load it to timer
2577        09930 HZ60END EQU $
          09940 ;
          09950 ; Routine to process GRAPHIC
          09960 ;
2577 010100 09970 GRAPHI LD BC,1 ;Init for bypass
257A 78      09980 LD A,B
257B B1      09990 OR C
257C 3D      10000 DEC A ;Check if parm entered
257D 280D    10010 JR Z,GRAEND ;Go if not entered
257F 3C      10020 INC A ;Test of GRAPHIC=off
2580 2806    10030 JR Z,GROFF ;Go if off
2582 FDCB03FE 10040 SET 7,(IY+'D'-'A') ; else turn on
2586 1804    10050 JR GRAEND
2588 FDCB03BE 10060 GROFF RES 7,(IY+'D'-'A') ;Turn graphic off
258C        10070 GRAEND EQU $
          10080 ;
          10090 ; Routine to process ALIVE
          10100 ;
258C 010100 10110 ALIVE LD BC,1 ;P/u Alive parm value
258F 78      10120 LD A,B
2590 B1      10130 OR C
2591 3D      10140 DEC A ;Check if entered
2592 2866    10150 JR Z,ALVEND ;Bypass if parm omitted
2594 3C      10160 INC A
2595 2007    10170 JR NZ,ALIVEON ;Jump if ALIVE=ON
          10180 IF @MOD2
          10190 LD C,6 ;Slot alive
          10200 ENDIF
          10210 IF @MOD4
2597 0E03    10220 LD C,3 ;Slot alive
          10230 ENDIF
2599        10240 @@RMTSK ;Remove it if OFF
2599 3E1E    00022 LD A,30
259B EF      00023 RST 40
259C 185C    10250 JR ALVEND
          10260 ;
259E        10270 ALIVEON EQU $
          10280 IF @MOD2
          10290 LD C,6 ;Slot alive
          10300 LD DE,DO_ALIVE ;Task address
259E        10310 @@ADTSK ;Add task

```



```

00024 LD A,29
00025 RST 40
10320 ENDIF
10330 IF @MOD4
259E CD7729 10340 CALL GOTMEM? ;Is HIGH$ frozen?
25A1 DA1024 10350 JP C,NOMEM ;Quit if so
25A4 CD6F29 10360 CALL GETHI$ ;Get address for Alive
25A7 22D325 10370 LD (ALVBGN+2),HL ;Stuff last byte used
25AA 012900 10380 LD BC,ALVEND-ALVBGN ;Get len of code
25AD AF 10390 XOR A
25AE ED42 10400 SBC HL,BC ;Calc new HIGH$
25B0 0600 10410 LD B,0
25B2 10420 @@HIGH$ ;Stuff new HIGH$
25B2 3E64 00026 LD A,100
25B4 EF 00027 RST 40
25B5 23 10430 INC HL ;Point to module start
25B6 E5 10440 PUSH HL ;Save start
25B7 0E10 10450 LD C,ALVTCB-ALVBGN+2
25B9 09 10460 ADD HL,BC ;Point to alive task
25BA 22DF25 10470 LD (ALVTCB),HL ; & stuff ALIVE TCB
25BD E3 10480 EX (SP),HL ;Rcvr start of module
25BE EB 10490 EX DE,HL ; into DE
25BF 21D125 10500 LD HL,ALVBGN
25C2 012900 10510 LD BC,ALVEND-ALVBGN
25C5 EDB0 10520 LDIR ;Move to high memory
25C7 D1 10530 POP DE ;Rcvr start of task
25C8 1B 10540 DEC DE ;Back up to TCB
25C9 1B 10550 DEC DE
25CA 0E03 10560 LD C,3 ;Add as task 3
25CC 10570 @@ADTSK
25CC 3E1D 00028 LD A,29
25CE EF 00029 RST 40
25CF 1829 10580 JR ALVEND
10590 ENDIF
10600 ;
10610 ; ALIVE high memory module
10620 ;
10630 IF @MOD4
25D1 18FE 10640 ALVBGN JR $
25D3 0000 10650 DW $-$
25D5 05 10660 DB 5,'ALIVE'
41 4C 49 56 45
25DB 0000 10670 DW 0,0
0000
25DF 0000 10680 ALVTCB DW 0
25E1 214F00 10690 LD HL,79 ;Get character at 0,79
25E4 E5 10700 PUSH HL
25E5 0601 10710 LD B,1 ;Set function 1
25E7 CD990B 10720 CALL @VDCTL ;P/u character currently
25EA FEA3 10730 CP 0A3H ; on the screen and
25EC 3E93 10740 LD A,93H ; exchange it
25EE 2802 10750 JR Z,ALIVE1
25F0 3EA3 10760 LD A,0A3H
25F2 E1 10770 ALIVE1 POP HL
25F3 0602 10780 LD B,2 ;Function to put char
25F5 4F 10790 LD C,A
25F6 CD990B 10800 CALL @VDCTL ; at row,col
25F9 C9 10810 RET
10820 ENDIF
25FA 10830 ALVEND EQU $
10840 ;

```

```

10850 ;      Routine to process SYSRES
10860 ;
25FA 01FFFF 10870 SYSRES LD      BC,-1      ;P/u Sysres parm value
25FD 04      10880      INC      B          ;Test if parm entered
25FE CA3B27 10890      JP        Z,RESEND   ;Go if not
2601 CD7729 10900      CALL     GOTMEM?    ;Is high memory available
2604 DA1024 10910      JP        C,NOMEM    ;Quit if not
2607 79      10920      LD        A,C       ;Get module to reside
2608 B7      10930      OR        A
2609 2837    10940      JR        Z,BADSYS   ;Cannot sysres 0
260B FE02    10950      CP        2         ;Check if it is SYS2
260D 2005    10960      JR        NZ,TEST6  ;Go if not
260F 217404 10970      LD        HL,DCT$+4  ;Point to SYSTEM DCT
2612 CBBE    10980      RES      7,(HL)     ; & allow CKDRV now
2614 FE06    10990 TEST6 CP        6
2616 3808    11000      JR        C,GUDSYS   ;Can sysres 1-5
2618 FE09    11010      CP        9
261A 3826    11020      JR        C,BADSYS   ;Cannot sysres 6-8
261C FE0D    11030      CP        13
261E 3022    11040      JR        NC,BADSYS  ;Nothing > 12 yet
2620 F5      11050 GUDSYS PUSH     AF       ;Save sysres req
2621 216D1A 11060      LD        HL,HKRES$+1 ;Check if the driver
2624 5E      11070      LD        E,(HL)    ; is already resident
2625 23      11080      INC      HL
2626 56      11090      LD        D,(HL)
2627 217F1A 11100      LD        HL,CKMOD0   ;Standard address
262A B7      11110      OR        A         ;Reset carry
262B ED52    11120      SBC      HL,DE      ;No driver if =
262D 2871    11130      JR        Z,PUTDVR  ;Go and install driver
262F 21E0FF 11140      LD        HL,-32     ;Find restab$ pos
2632 19      11150      ADD      HL,DE      ;Pt to vector table
2633 07      11160      RLCA
2634 85      11170      ADD      A,L
2635 6F      11180      LD        L,A
2636 3001    11190      JR        NC,$+3
2638 24      11200      INC      H          ;RESTAB$ + entry index
2639 23      11210      INC      HL         ;Pt to hi order
263A 7E      11220      LD        A,(HL)    ;P/u hi order
263B B7      11230      OR        A
263C 281B    11240      JR        Z,MOVITIN  ;Go if not there
263E 21022A 11250      LD        HL,MODRES$  ; else show already res'ed
2641 DD      11260      DB        0DDH
2642 21E429 11270 BADSYS LD        HL,BADSYS$
2645 CD1724 11280      CALL     SETERR      ;Log & Set error
2648 F1      11290      POP      AF
2649 C33B27 11300      JP        GETSEC2
264C 18FE    11310 RES1  JR        $
264E 0000    11320      DW        $-$      ;No real entry
2650 04      11330      DB        4,'SYS '  ;Last used
2655 0000    11340      DW        0,0
2658 0000    11350 RES1E EQU      $-1
2659 326900 11360 MOVITIN LD      (OVRLY$),A ;Show no overlay res
265C F1      11370      POP      AF         ;Rcvr requested SYS
265D F5      11380      PUSH     AF
265E C602    11390      ADD      A,2        ;Adjust for dec
2660 F680    11400      OR        80H       ;Set Sys request
2662 E5      11410      PUSH     HL         ;Save restab$ addr
2663 CD7F1A 11420      CALL     CKMOD0     ;Load SYSn, no exec
2666 CD6F29 11430      CALL     GETHI$     ;P/u HIGH$

```

The Source	LIBRARY Files	SYSTEM - LS-DOS 6.2	Page 00009
2669 224E26	11440	LD (RES1+2),HL	;Save in header
266C EB	11450	EX DE,HL	
266D ED4BFE23	11460	LD BC,(MAXCOR\$-2)	;P/u length of SYSx
2671 21001E	11470	LD HL,1E00H	;Pt to start of overlay
2674 09	11480	ADD HL,BC	
2675 2B	11490	DEC HL	;Point to last byte
2676 C5	11500	PUSH BC	;Save length
2677 EDB8	11510	LDDR	
2679 C1	11520	POP BC	;Rcvr length of SYSx
267A 78	11530	LD A,B	; & stuff in front
267B 12	11540	LD (DE),A	; of SYSRESeD module
267C 1B	11550	DEC DE	
267D 79	11560	LD A,C	
267E 12	11570	LD (DE),A	
267F E1	11580	POP HL	;Rcvr RESTAB ptr
2680 72	11590	LD (HL),D	;Save hi order addr
2681 2B	11600	DEC HL	
2682 73	11610	LD (HL),E	;Save low order addr
2683 F1	11620	POP AF	;Rcvr module requested
2684 FE0A	11630	CP 10	;Convert to ASCII
2686 3802	11640	JR C,\$+4	
2688 C607	11650	ADD A,7	
268A C630	11660	ADD A,'0'	
268C 215426	11670	LD HL,RES1+8	
268F 77	11680	LD (HL),A	;Stuff module name
2690 1B	11690	DEC DE	;Pt to 1st free byte
	11700		; & move in the
2691 010D00	11710	LD BC,13	; linkage protocol
2694 215826	11720	LD HL,RES1E	
2697 EDB8	11730	LDDR	
2699 EB	11740	EX DE,HL	
269A	11750	@@HIGH\$;Stuff new HIGH\$ (B=0)
269A 3E64	00030	LD A,100	
269C EF	00031	RST 40	
269D C33B27	11760	JP RESEND	
	11770 ;		
26A0 CD6F29	11780	PUTDVR CALL GETHI\$;P/u high mem
26A3 22CF26	11790	LD (RESBGN+2),HL	;Stuff last used
26A6 016E00	11800	LD BC,RESEND-RESBGN	
26A9 ED42	11810	SBC HL,BC	;Make space for driver
26AB 0600	11820	LD B,0	
26AD	11830	@@HIGH\$;Stuff new value
26AD 3E64	00032	LD A,100	
26AF EF	00033	RST 40	
26B0 E5	11840	PUSH HL	;Save new HIGH\$
26B1 110D00	11850	LD DE,13	;Pt to RESTAB\$-3
26B4 19	11860	ADD HL,DE	
26B5 220827	11870	LD (RESD1+1),HL	;Stuff ptr to RESTAB\$-3
26B8 E1	11880	POP HL	
26B9 23	11890	INC HL	;Resbgn
26BA E5	11900	PUSH HL	
26BB EB	11910	EX DE,HL	;Where it goes to DE
26BC 21CD26	11920	LD HL,RESBGN	;Where it's now
26BF EDB0	11930	LDIR	;Move up to high
26C1 E1	11940	POP HL	;Rcvr where it got to
26C2 112F00	11950	LD DE,32+RESTAB\$-RESBGN	;Index start of dvr
26C5 19	11960	ADD HL,DE	;Pt to driver entry
26C6 226D1A	11970	LD (HKRES\$+1),HL	;Hook into SYS0
26C9 F1	11980	POP AF	;Rcvr code & loop
26CA C32026	11990	JP GUDSYS	; to get the request
	12000 ;		

```

26CD 18FE       12010 RESBGN JR       $                      ;No real entry
26CF 0000       12020           DW       $-$
26D1 06       12030           DB       6,'SYSRES'
     53 59 53 52 45 53
26D8 0000       12040           DW       0,0
     0000
26DC 0100       12050 RESTAB$ DW       1,0,0,0,0,0,1,1
     0000 0000 0000 0000 0000 0100 0100
26EC 0000       12060           DW       0,0,0,0,0,0,0,0
     0000 0000 0000 0000 0000 0000 0000
     12070 ;
26FC E5       12080 RESDVR PUSH       HL
26FD F5       12090           PUSH       AF                      ;Save SYS needed
26FE 216900     12100           LD       HL,OVRLY$               ;Check if already in
2701 AE       12110           XOR       (HL)                   ; overlay area
2702 E60F     12120           AND       0FH                   ;Strip garbage
2704 2830     12130           JR       Z,NOTRES
2706 F1       12140           POP       AF
2707 21DD26     12150 RESD1 LD       HL,RESTAB$+1             ;P/u table ptr
270A F5       12160           PUSH       AF
270B E60F     12170           AND       0FH                   ;Check if in high mem
270D 07       12180           RLCA                           ;X 2
270E 85       12190           ADD       A,L
270F 6F       12200           LD       L,A
2710 3001     12210           JR       NC,$+3
2712 24       12220           INC       H
2713 7E       12230           LD       A,(HL)                 ;P/u hi order
2714 B7       12240           OR       A
2715 281F     12250           JR       Z,NOTRES               ;Go if not in high
2717 2B       12260           DEC       HL                   ;Pt to low
2718 6E       12270           LD       L,(HL)               ;P/u lo order vector
2719 67       12280           LD       H,A                   ;Xfer hi order
271A D5       12290           PUSH       DE                   ;Save these regs
271B C5       12300           PUSH       BC
271C 4E       12310           LD       C,(HL)               ;P/u module length
271D 23       12320           INC       HL                   ; into BC
271E 46       12330           LD       B,(HL)
271F 23       12340           INC       HL                   ;Point to module start
2720 11001E     12350           LD       DE,1E00H               ;Point to overlay region
2723 ED53751A   12360           LD       (HKRES$+9),DE           ;Upd TRAADR
2727 C5       12370           PUSH       BC                   ;Save the length
2728 EDB0     12380           LDIR
272A E1       12390           POP       HL                   ;Rcvr the length
272B 22FE23     12400           LD       (MAXCOR$-2),HL           ;Stuff pointer
272E C1       12410           POP       BC                   ;Restore regs
272F D1       12420           POP       DE
2730 F1       12430           POP       AF
2731 32701A     12440           LD       (HKRES$+4),A             ;Show its resident
2734 E1       12450           POP       HL                   ; in tempy loc'n
2735 C9       12460           RET
2736 F1       12470 NOTRES POP       AF
2737 E1       12480           POP       HL
2738 C37F1A     12490           JP       CKMOD@
273B       12500 RESEND EQU       $
     12510 ;
     12520 ;       Routine to process DRIVE parameter
     12530 ;
273B 3A7B27     12540 GETSEC2 LD       A,(CYLPRM+1)           ;If CYL=c entered,
273E B7       12550           OR       A                   ; get config sector
273F C45F29     12560           CALL       NZ,GETCFG
2742 01FFFF     12570 DRIVE LD       BC,-1

```

The Source	LIBRARY Files	SYSTEM - LS-DOS 6.2	Page 00011
2745 DD21701D	125800	LD IX,SBUFF\$+70H	;Pt to DCT sector fields
2749 FD217004	125900	LD IY,DCT\$;Init to do all DCT\$'s *
274D 04	126000	INC B	
274E 280B	126100	JR Z,CKEI	;Go if Drive not entered
2750 79	126200	LD A,C	
2751 FE08	126300	CP 8	;Drive in range?
2753 D21424	126400	JP NC,PRMERR	;Go if > 7
2756	126500	@@GTDCT	;Get its DCT
2756 3E51	00034	LD A,81	
2758 EF	00035	RST 40	
2759 FDE5	126600	PUSH IY	;Save the DCT address
275B	126700	CKEI EQU \$	
	126800	;	
275B FDE5	126900	ADJEI PUSH IY	;Calc IY offset into
275D E1	127000	POP HL	; DCTs to adjust IX
275E 1190FB	127100	LD DE,-DCT\$; without affecting IY
2761 19	127200	ADD HL,DE	
2762 EB	127300	EX DE,HL	
2763 DD19	127400	ADD IX,DE	
2765	127500	DRVLOOP EQU \$	
	127600	;	
	127700	;	Routine to process WP parameter
	127800	;	
2765 010100	127900	WPARAM LD BC,1	;P/u WP parm value
2768 78	128000	LD A,B	
2769 B1	128100	OR C	
276A 3D	128200	DEC A	;Check if entered
276B 280D	128300	JR Z,WPEND	; and go if not
276D 3C	128400	INC A	
276E 2806	128500	JR Z,WPOFF	;Go if OFF
2770 FDCB03FE	128600	WPON SET 7,(IY+3)	;Turn on write protect
2774 1804	128700	JR WPEND	
2776 FDCB03BE	128800	WPOFF RES 7,(IY+3)	;Turn off write protect
277A	128900	WPEND EQU \$	
	129000	;	
	129100	;	Routine to process CYL parameter
	129200	;	
277A 010000	129300	CYLPRM LD BC,0	;P/u Cyl parm value
277D 78	129400	LD A,B	
277E B1	129500	OR C	
277F 282B	129600	JR Z,CYLEND	;Go if parm not entered
2781 DDCB036E	129700	BIT 5,(IX+3)	;If drive is 8",
2785 280C	129800	JR Z,CYL1	; can't do it
2787 3A4327	129900	LD A,(DRIVE+1)	;If drive entered,
278A B7	130000	OR A	; give error msg
278B 21A629	130100	LD HL,NOCYL8\$; 'Can't CYL on 8" drive '
278E C41724	130200	CALL NZ,SETERR	;Msg if DRIVE=d
2791 1819	130300	JR DISABL	
2793 79	130400	CYL1 LD A,C	;P/u user entry
2794 3D	130500	DEC A	
2795 FE22	130600	CP 34	;Not < 35 allowed
2797 3804	130700	JR C,BADCYL	;Go if bad
2799 FE60	130800	CP 96	;Max for 5-1/4" floppy
279B 380C	130900	JR C,GUDCYL	;Good if < 96
279D 21C029	131000	BADCYL LD HL,BADCYL\$; "Cyl out of range
27A0 CD1724	131100	CALL SETERR	;Dsply and quit
27A3 AF	131200	XOR A	
27A4 327B27	131300	LD (CYLPRM+1),A	
27A7 1803	131400	JR DISABL	
27A9 DD7706	131500	GUDCYL LD (IX+6),A	;Stuff default DCT
27AC	131600	CYLEND EQU \$	

```

13170 ;
13180 ;      Routine to process DISABL parameter
13190 ;
27AC 010000 13200 DISABL LD      BC,0          ;P/u parm value
27AF 78      13210      LD      A,B
27B0 B1      13220      OR      C
27B1 280A    13230      JR      Z,ENABLE        ;Go if not DISABLE
27B3 3A4327  13240      LD      A,(DRIVE+1)     ;Don't permit disable
27B6 3C      13250      INC     A              ; if DRIVE parm not
27B7 2810    13260      JR      Z,STEP          ; entered
27B9 3EC9    13270      LD      A,0C9H         ;Init disable byte
27BB 1809    13280      JR      ENADIS
27BD 010000  13290 ENABLE LD      BC,0          ;P/u parm value
27C0 78      13300      LD      A,B
27C1 B1      13310      OR      C
27C2 2805    13320      JR      Z,STEP          ;Go if not ENABLE
27C4 3EC3    13330      LD      A,0C3H         ;Init enable byte
27C6 FD7700  13340 ENADIS LD      (IY),A       ;Stuff Ena or Dis byte
13350 ;
13360 ;      Routine to process STEP parameter
13370 ;
27C9 01FFFF  13380 STEP  LD      BC,-1         ;P/u Step parm value
27CC 04      13390      INC     B
27CD 2815    13400      JR      Z,STEPEND       ;Go if STEP not entered
27CF 79      13410      LD      A,C
27D0 FE04    13420      CP      4              ;Step must be < 4
27D2 D21424  13430      JP      NC,PRMERR       ;Quit if out of range
27D5 FDCB0466 13440      BIT     4,(IY+4)       ;Alien controller?
27D9 2039    13450      JR      NZ,WDESD       ;Bypass if alien
27DB FD7E03  13460      LD      A,(IY+3)       ;P/u DCT byte and
27DE E6FC    13470      AND     0FCH          ; mask out old step
27E0 B1      13480      OR      C              ;Or in the new value
27E1 FD7703  13490      LD      (IY+3),A       ; and put back in DCT
27E4        13500 STEPEND EQU $
13510 ;
13520 ;      Routine to process DELAY parameter
13530 ;
27E4 010100  13540 DELAY LD      BC,1          ;P/u the parm value
27E7 78      13550      LD      A,B
27E8 B1      13560      OR      C
27E9 3D      13570      DEC     A              ;Check if entered
27EA 2810    13580      JR      Z,CKDRV         ;Go if no delay entered
27EC 3C      13590      INC     A              ;Check if OFF
27ED 0600    13600      LD      B,0            ;Init delay=on
27EF 2002    13610      JR      NZ,DELAY1      ;Go if ON
27F1 0604    13620      LD      B,4            ;Init delay=off
27F3 FD7E03  13630 DELAY1 LD      A,(IY+3)     ;Update delay
27F6 E6FB    13640      AND     0FBH          ; in DCT
27F8 B0      13650      OR      B
27F9 FD7703  13660      LD      (IY+3),A
13670 ;
13680 ;      Routine to process CKDRV parameter
13690 ;
27FC 010100  13700 CKDRV LD      BC,1          ;P/u Ckdrv parm value
27FF 78      13710      LD      A,B
2800 B1      13720      OR      C
2801 3D      13730      DEC     A              ;Check if entered
2802 2810    13740      JR      Z,WDESD       ;Go if not entered
2804 3C      13750      INC     A              ;See if OFF or ON
2805 0600    13760      LD      B,0            ;Init ckdrv=on
2807 2002    13770      JR      NZ,CKDRV1     ;Go if ON

```

```

2809 0680      13780      LD      B,80H      ;Set bit 7, ckdrv off
280B FD7E04    13790 CKDRV1 LD      A,(IY+4)    ;Update ckdrv bit
280E E67F      13800      AND     7FH        ; by removing old state,
2810 B0        13810      OR      B          ; merging 0 or 80H
2811 FD7704    13820      LD      (IY+4),A      ; and putting back in DCT
2814 3A4327    13830 WDESD LD      A,(DRIVE+1) ;Drive parm used prev ?
2817 3C        13840      INC     A
2818 2014      13850      JR      NZ,PUTSEC2    ;Go if DRIVE entered
281A 110A00    13860      LD      DE,10        ; else loop thru all DCTs
281D DD19      13870      ADD     IX,DE        ;Advance to next DCT
281F FD19      13880      ADD     IY,DE
2821 FDE5      13890      PUSH    IY
2823 E1        13900      POP     HL          ;Xfer DCT start to HL
2824 7D        13910      LD      A,L          ;Ck on end of DCTs
2825 FEC0      13920      CP      80+70H      ;Offset is 70H
2827 DA6527    13930      JP      C,DRVLOOP    ;Loop until done
282A 210000    13940      LD      HL,0        ;Show no DCT given
282D E5        13950      PUSH    HL
282E 3A7B27    13960 PUTSEC2 LD     A,(CYLPRM+1) ;If CYL=c entered
2831 B7        13970      OR      A          ; put config sector
2832 C45C29    13980      CALL   NZ,PUTCFG
2833          13990      ;
2834          14000      ;
2835          14010      ;
2836          14020      ;
2837          14030      ;
2838          14040      ;
2839          14050      ;
283A CAD928    14060      ;
283D 3C        14070      ;
283E 2823      14080      ;
2840 219E2B    14090      ;
2843 E5        14100      ;
2844 1A        14110      ;
2845 FE0D      14120      ;
2847 2815      14130      ;
2849 FE2C      14140      ;
284B 2811      14150      ;
284D FE22      14160      ;
284F 280D      14170      ;
2851 FE29      14180      ;
2853 2809      14190      ;
2855 FE03      14200      ;
2857 2805      14210      ;
2859 77        14220      ;
285A 13        14230      ;
285B 23        14240      ;
285C 18E6      14250      ;
285E 360D      14260      ;
2860 E1        14270      ;
2861 1844      14280      ;
2863          14290      ;
2864          14300      ;
2865          14310      ;
2866          14320      ;
2867          14330      ;
2868          14340      ;
2869          14350      ;
286A          14360      ;
286B          14370      ;
286C          14380      ;
286D          14390      ;
286E          14400      ;
286F          14410      ;
2870          14420      ;
2871          14430      ;

```

The Source	LIBRARY Files	SYSTEM - LS-DOS 6.2	Page 00014
2872 3033	14320	JR NC,DVR2	;Go if not BREAK
	14330 ;		
	14340 ;	BREAK entered on DRIVER request	
	14350 ;	Reset all drive code table positions	
	14360 ;	if DRIVE not entered	
	14370 ;		
2874 CD5F29	14380	CALL GETCFG	;Get SIS
2877 210019	14390	LD HL,SBUFF\$+70H-DCT\$;Pt to tables "offset"
287A 3A4327	14400 DDCT1	LD A,(DRIVE+1)	;Was Drive entered?
287D 3C	14410	INC A	
287E D1	14420	POP DE	;Rcvr drive #
287F 2003	14430	JR NZ,DDCT2	
2881 117004	14440	LD DE,DCT\$;Else use all 8
2884 19	14450 DDCT2	ADD HL,DE	;Index the specific DCT
2885 7B	14460 DVRA	LD A,E	
2886 FE98	14470	CP 40+70H	
2888 3EC9	14480	LD A,0C9H	;Disable drives 4-7
288A 3002	14490	JR NC,DVRB	
288C 3EC3	14500	LD A,0C3H	
288E 77	14510 DVRB	LD (HL),A	;Stuff vector (JP/RET)
288F 2C	14520	INC L	;Pt to default DCT+3
2890 2C	14530	INC L	
2891 2C	14540	INC L	
2892 1C	14550	INC E	;Pt to resident DCT+3
2893 1C	14560	INC E	
2894 1C	14570	INC E	
2895 010700	14580	LD BC,7	;Shift defaults to
2898 EDB0	14590	LDIR	; resident DCT
289A 3A4327	14600 DVRC	LD A,(DRIVE+1)	;Test if drive entered
289D 3C	14610	INC A	
289E 203A	14620	JR NZ,SYSPRM	;Go if yes
28A0 7B	14630	LD A,E	;Ck on end of DCTs
28A1 FEC0	14640	CP 80+70H	
28A3 30E0	14650	JR NC,DVRA	;Loop until all 8
28A5 1833	14660	JR SYSPRM	
28A7 7E	14670 DVR2	LD A,(HL)	;Is first char <ENTER>?
28A8 FE0D	14680	CP CR	
28AA 282D	14690	JR Z,DVR4	;Bypass if default
28AC CD7729	14700	CALL GOTMEM?	
28AF D1	14710	POP DE	;Pop to keep stack
28B0 DA1024	14720	JP C,NOMEM	; straight if error
28B3 D5	14730	PUSH DE	
28B4 119D2B	14740	LD DE,FCB1	;Else fetch spec
28B7	14750	@@FSPEC	
28B7 3E4E	00043	LD A,78	
28B9 EF	00044	RST 40	
28BA 21A02A	14760	LD HL,DCTEXT	;Use /DCT as the
28BD	14770	@@FEXT	; default extension
28BD 3E4F	00045	LD A,79	
28BF EF	00046	RST 40	
28C0	14780	@@FLAGS	
28C0 3E65	00047	LD A,101	
28C2 EF	00048	RST 40	
28C3 FDCB12D6	14790	SET 2,(IY+'S'-'A')	;Set RUN bit
28C7	14800	@@LOAD	;Load the prog
28C7 3E4C	00049	LD A,76	
28C9 EF	00050	RST 40	
28CA D1	14810	POP DE	;Recover drive table
28CB C21F24	14820	JP NZ,IOERR	;Quit on load error
28CE D5	14830	PUSH DE	
28CF E5	14840	PUSH HL	;Save prog's traadr

The Source	LIBRARY Files	SYSTEM - LS-DOS 6.2	Page 00015
28D0 21D928	14850	LD HL,DVR4 ;Pt to our return	
28D3 E3	14860	EX (SP),HL ;Exch with his	
28D4 FDCB02DE	14870	SET 3,(IY+'C'-'A') ;Set system request	
28D8 E9	14880	JP (HL) ;Go to prog	
28D9 D1	14890	POP DE ;Rcvr drive table	
	14900 ;		
	14910 ;	System = new system drive	
	14920 ;		
28DA 01FFFF	14930	SYSPRM LD BC,-1 ;P/u parm value	
28DD 78	14940	LD A,B	
28DE C3842B	14950	JP PATCH1 ;Set HL=0, no abort if exit	
	14960 ;		
28E1 3D	14970	CONT1 DEC A	
28E2 FE08	14980	CP 8 ;Drive in range?	
28E4 D21424	14990	JP NC,PRMERR ;Go if > 7	
	15000 ;		
	15010 ;	Check on diskette in place	
	15020 ;		
28E7	15030	@@CKDRV	
28E7 3E21	00051	LD A,33	
28E9 EF	00052	RST 40	
28EA 2025	15040	JR NZ,SYSP2 ;Prompt if no diskette	
28EC	15050	@@GTDCT ;Get its DCT	
28EC 3E51	00053	LD A,81	
28EE EF	00054	RST 40	
	15060 ;		
	15070 ;	Perform minimal check for SYSTEM disk	
	15080 ;		
28EF FD5609	15090	SYSP1 LD D,(IY+9) ;P/u the dir cyl	
28F2 21001D	15100	LD HL,SBUFF\$	
28F5 5D	15110	LD E,L ;Init for GAT read	
28F6	15120	@@RDSSC ;Read the sector	
28F6 3E55	00055	LD A,85	
28F8 EF	00056	RST 40	
28F9 C21F24	15130	JP NZ,IOERR ;Quit on read error	
28FC 2ECD	15140	LD L,0CDH ;Check the SYSTEM bit	
28FE CB7E	15150	BIT 7,(HL)	
2900 200F	15160	JR NZ,SYSP2 ;Not system? Go prompt	
2902 21001D	15170	LD HL,SBUFF\$;Point to buffer	
2905 1E05	15180	LD E,5 ;Point to dir sec #5	
2907	15190	@@RDSSC ;Read the sector	
2907 3E55	00057	LD A,85	
2909 EF	00058	RST 40	
290A 7E	15200	LD A,(HL) ;P/u the byte	
290B E650	15210	AND 01010000B ;Check for a system file	
290D FE50	15220	CP 01010000B ; and not killed	
	15230	; in this slot, (SYS1)	
290F 2818	15240	JR Z,SYSP3 ;Alive and a SYS file	
	15250 ;		
	15260 ;	Diskette is not a SYSTEM diskette	
	15270 ;		
2911 79	15280	SYSP2 LD A,C ;Stuff drive # in msg	
2912 C630	15290	ADD A,'0' ;Cvrt drive to ASCII	
2914 32802A	15300	LD (NDSYS\$+32),A	
2917	15310	@@DSPLY NDSYS\$;"Need system diskette..	
	00059	IFEQ 01H,1	
2917 21602A	00060	LD HL,NDSYS\$	
	00061	ENDIF	
291A 3E0A	00062	LD A,10	
291C EF	00063	RST 40	
291D 219D2B	15320	LD HL,FCB1 ;Create response buffer	

The Source	LIBRARY Files	SYSTEM - LS-DOS 6.2	Page 00016
2920 010000	15330	LD BC,0<8	;Only BREAK or ENTER
2923	15340	@KEYIN	;Wait for entry
2923 3E09	00064	LD A,9	
2925 EF	00065	RST 40	
2926 D8	15350	RET C	;Exit on <BREAK>
2927 18B1	15360	JR SYSPRM	;Continue to check
	15370 ;		
2929 217404	15380 SYSP3	LD HL,DCT\$+4	
292C CB7E	15390	BIT 7,(HL)	;Check Ckdrv bit in DCT
292E CB8E	15400	RES 7,(HL)	;Enable it for now
2930 217004	15410	LD HL,DCT\$	
2933 119D2B	15420	LD DE,FCB1	;Use FCB space
2936 D5	15430	PUSH DE	;Save where temp
2937 E5	15440	PUSH HL	;Save where new SYSTEM
2938 010A00	15450	LD BC,10	;Save current SYS's DCT
293B EDB0	15460	LDIR	
293D FDCB04BE	15470	RES 7,(IY+4)	;Make sure OFF
2941 2804	15480	JR Z,NOSET	;Only set if it was set
	15490		; on the SYSTEM drive
2943 FDCB04FE	15500	SET 7,(IY+4)	;CKDRV inhibit drive 0
2947 FDE5	15510 NOSET	PUSH IY	;New DCT to HL
2949 E1	15520	POP HL	
294A D1	15530	POP DE	;Rcvr SYS DCT now
294B E5	15540	PUSH HL	;Save where it goes
294C 0E0A	15550	LD C,10	
294E EDB0	15560	LDIR	;New DCT into SYSTEM
2950 D1	15570	POP DE	;P/u where old SYS goes
2951 E1	15580	POP HL	;P/u temp address
2952 0E0A	15590	LD C,10	;Move old SYS to new DCT
2954 EDB0	15600	LDIR	
2956 21452A	15610	LD HL,NWSYS\$;New sys drive installed
2959 C31724	15620	JP SETERR	;Display and exit w/abort
	15630		; in case JCL active
	15640 ;		
	15650 ;		
	15660 ;		
		Routines to read/write the config sector	
295C 3E35	15670 PUTCFG	LD A,53	;@WRSEC
	15680 ;		
	15690	IF @MOD4	
295E 21	15700	DB 21H	;Ignore next with LD HL,
	15710	ENDIF	
	15720	IF @MOD2	
	15730	JR PUTGETC	;Jump over if mod2
	15740	ENDIF	
	15750 ;		
295F	15760 GETCFG	EQU \$	
	15770	IF @MOD4	
295F 3E31	15780	LD A,49	;@RDSEC
2961 110200	15790	LD DE,0<8+2	;Config sector
	15800	ENDIF	
	15810 ;		
	15820	IF @MOD2	
	15830	LD A,(DCT\$+3)	;Drive 0 dct data
	15840	AND 28H	;Bit 5/3
	15850	CP 20H	;8" floppy?
	15860	JR NZ,SETSYS1	;Go if not
	15870	LD A,(DCT\$+4)	;Get data
	15880	AND 50H	;Bit 6/4
	15890	CP 40H	;DD not alien?
	15900	JR NZ,SETSYS1	;Go if not
	15910 ;		

```

15920 LD HL,SBUFF$ ;System buffer
15930 LD A,(DCT$+9) ;Get dir cylinder
15940 LD D,A ;Pass to D
15950 LD E,0 ;Init sector 0
15960 LD C,E ;Init drive 0
2964 15970 @@RDSSC ;Read directory
00066 LD A,85
00067 RST 40
15980 JR NZ,PUTGETE ;Go on disk error
15990 LD A,(SBUFF$+0CDH) ;Get gat data byte
16000 BIT 7,A ;Is data disk?
16010 SETSYS1 LD DE,0<8+2 ;Init sysinfo sector
16020 JR NZ,$+3 ;Go if data disk
16030 INC D ;Else new sysinfo
16040 LD A,49 ;Init RDSEC
16050 ENDF
16060 ;
2964 21001D 16070 PUTGETC LD HL,SBUFF$ ;System buffer
2967 0E00 16080 LD C,0 ; in system drive
2969 EF 16090 RST 40 ;Read or Write
296A C8 16100 RET Z ;Back if good i/o
296B E1 16110 PUTGETE POP HL
296C C31F24 16120 JP IOERR ; else abort
16130 ;
16140 ; Routine to get HIGH$
16150 ;
296F 0600 16160 GETHI$ LD B,0 ;Init to get HIGH$
2971 60 16170 LD H,B
2972 68 16180 LD L,B
2973 16190 @@HIGH$ ;Get current HIGH$
2973 3E64 00068 LD A,100
2975 EF 00069 RST 40
2976 C9 16200 RET
16210 ;
16220 ; Routine to test if HIGH$ is frozen
16230 ;
2977 3A6C00 16240 GOTMEM? LD A,(CFLAG$) ;Check if memory can
297A 0F 16250 RRCA ; be altered
297B C9 16260 RET
16270 ;
297C 4E 16280 NOMEM$ DB 'No memory space available',CR
6F 20 6D 65 6D 6F 72 79
20 73 70 61 63 65 20 61
76 61 69 6C 61 62 6C 65
0D
2996 50 16290 PRMERR$ DB 'Parameter error',CR
61 72 61 6D 65 74 65 72
20 65 72 72 6F 72 0D
29A6 43 16300 NOCYL8$ DB 'CYL=c invalid on 8" drive',CR
59 4C 3D 63 20 69 6E 76
61 6C 69 64 20 6F 6E 20
38 22 20 64 72 69 76 65
0D
29C0 43 16310 BADCYL$ DB 'Cylinder count out of range <35-96>',CR
79 6C 69 6E 64 65 72 20
63 6F 75 6E 74 20 6F 75
74 20 6F 66 20 72 61 6E
67 65 20 3C 33 35 2D 39
36 3E 0D
29E4 43 16320 BADSYS$ DB 'Can''t SYSRES requested module',CR
61 6E 27 74 20 53 59 53

```

	52 45 53 20 72 65 71 75		
	65 73 74 65 64 20 6D 6F		
	64 75 6C 65 0D		
2A02	53 16330 MODRES\$ DB		'System module already resident',CR
	79 73 74 65 6D 20 6D 6F		
	64 75 6C 65 20 61 6C 72		
	65 61 64 79 20 72 65 73		
	69 64 65 6E 74 0D		
2A21	45 16340 EIPMPT DM		'Enter DCT driver <BREAK=default> : ',3'
	6E 74 65 72 20 44 43 54		
	20 64 72 69 76 65 72 20		
	3C 42 52 45 41 4B 3D 64		
	65 66 61 75 6C 74 3E 20		
	3A 20 03		
2A45	4E 16350 NWSYS\$ DB		'New SYSTEM drive installed',CR
	65 77 20 53 59 53 54 45		
	4D 20 64 72 69 76 65 20		
	69 6E 73 74 61 6C 6C 65		
	64 0D		
2A60	49 16360 NDSYS\$ DB		'Insert SYSTEM diskette in drive X ',CR
	6E 73 65 72 74 20 53 59		
	53 54 45 4D 20 64 69 73		
	6B 65 74 74 65 20 69 6E		
	20 64 72 69 76 65 20 58		
	20 0D		
2A83	42 16370 BADBS\$ DB		'Boot step out of range <0-3>',CR
	6F 6F 74 20 73 74 65 70		
	20 6F 75 74 20 6F 66 20		
	72 61 6E 67 65 20 3C 30		
	2D 33 3E 0D		
2AA0	44 16380 DCTEXT DM		'DCT'
	43 54		
	16390 ;		
2AA3	16400 PRMTBL\$ EQU		\$
2AA3	41 16410 DB		'ALIVE '
	4C 49 56 45 20		
2AA9	8D25 16420 DW		ALIVE+1
2AAB	42 16430 DB		'BSTEP '
	53 54 45 50 20		
2AB1	C124 16440 DW		BSPARM+1
2AB3	44 16450 DM		'DRIVER'
	52 49 56 45 52		
2AB9	3628 16460 DW		DVRPARAM+1
2ABB	46 16470 DB		'FAST '
	41 53 54 20 20		
2AC1	4224 16480 DW		FPARAM+1
2AC3	53 16490 DB		'SLOW '
	4C 4F 57 20 20		
2AC9	3B24 16500 DW		SLOW+1
2ACB	53 16510 DB		'SYSRES'
	59 53 52 45 53		
2AD1	FB25 16520 DW		SYSRES+1
2AD3	53 16530 DB		'SYSTEM'
	59 53 54 45 4D		
2AD9	DB28 16540 DW		SYSPRM+1
2ADB	54 16550 DB		'TYPE '
	59 50 45 20 20		
2AE1	2A25 16560 DW		TYPE+1
2AE3	57 16570 DB		'WP '
	50 20 20 20 20		
2AE9	6627 16580 DW		WPARAM+1

2AEB 42	16590	DB	'BLINK '	
4C 49 4E 4B 20				
2AF1 E224	16600	DW	BLINK+1	
2AF3 42	16610	DB	'BREAK '	
52 45 41 4B 20				
2AF9 7F24	16620	DW	BREAK+1	
2AFB 43	16630	DB	'CYL '	
59 4C 20 20 20				
2B01 7B27	16640	DW	CYLPRM+1	
2B03 44	16650	DB	'DATE '	
41 54 45 20 20				
2B09 9324	16660	DW	DATE+1	
2B0B 44	16670	DB	'DELAY '	
45 4C 41 59 20				
2B11 E527	16680	DW	DELAY+1	
2B13 43	16690	DB	'CKDRV '	
4B 44 52 56 20				
2B19 FD27	16700	DW	CKDRV+1	
2B1B 44	16710	DB	'DISABL'	
49 53 41 42 4C				
2B21 AD27	16720	DW	DISABL+1	
2B23 44	16730	DB	'DRIVE '	
52 49 56 45 20				
2B29 4327	16740	DW	DRIVE+1	
2B2B 45	16750	DB	'ENABLE'	
4E 41 42 4C 45				
2B31 BE27	16760	DW	ENABLE+1	
2B33 47	16770	DB	'GRAPHI'	
52 41 50 48 49				
2B39 7825	16780	DW	GRAPHI+1	
2B3B 4C	16790	DB	'LARGE '	
41 52 47 45 20				
2B41 FB24	16800	DW	LARGE+1	
2B43 52	16810	DB	'RESTOR'	
45 53 54 4F 52				
2B49 1325	16820	DW	RPARM+1	
2B4B 53	16830	DB	'SMALL '	
4D 41 4C 4C 20				
2B51 0525	16840	DW	SMALPRM+1	
2B53 53	16850	DB	'STEP '	
54 45 50 20 20				
2B59 CA27	16860	DW	STEP+1	
2B5B 54	16870	DB	'TIME '	
49 4D 45 20 20				
2B61 AA24	16880	DW	TIME+1	
2B63 54	16890	DB	'TRACE '	
52 41 43 45 20				
2B69 6624	16900	DW	TRACE+1	
2B6B 53	16910	DB	'SMOOTH'	
4D 4F 4F 54 48				
2B71 3F25	16920	DW	SMOOTH+1	
2B73 48	16930	DB	'HERTZ5'	
45 52 54 5A 35				
2B79 5E25	16940	DW	HERTZ50+1	
2B7B 48	16950	DB	'HERTZ6'	
45 52 54 5A 36				
2B81 6B25	16960	DW	HERTZ60+1	
2B83 00	16970	NOP		
	16980 ;			
2B84 210000	16990 PATCH1	LD	HL,0	;Set HL=0 "NO ERRORS"
2B87 B1	17000	OR	C	

The Source	LIBRARY Files	SYSTEM - LS-DOS 6.2	Page 00020
2B88 3C	17010 INC	A	;Parm entered?
2B89 C8	17020 RET	Z	;Exit if not
2B8A C3E128	17030 JP	CONT1	;Continue on now
	17040 ;		
2B8D 00	17050 DC	16,0	;Patch space
	00 00 00 00 00 00 00 00		
	00 00 00 00 00 00 00 00		
2B9D 00	17060 FCB1 DC	32,0	
	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		
	17070 ;		
2400	00110 END	SYSTEM	

\$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 @LOGGER	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
@PSN	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 ADJEI	275B
FLAG\$	006A ALIVE	258C ALIVE1	25F2
ALIVEON	259E ALVBGN	25D1 ALVEND	25FA
ALVTCB	25DF AUTO?	1FF1 BADBS\$	2A83
BADCYL	279D BADCYL\$	29C0 BADSYS	2642
BADSYS\$	29E4 BAR\$	0201 BLIEND	2512
BLINK	24E1 BLNKON	24F2 BOOTST\$	439D
BREAK	247E BREAK?	1C60 BRKEND	2492
BRKOFF	248E BRKVEC\$	1C88 BSPARM	24C0
BSPEND	24E1 BUR\$	0200 CASHK\$	0A7B
CFCB\$	00E0 CFGFCB\$	00E0 CFLAG\$	006C
CKDRV	27FC CKDRV1	280B CKEI	275B
CKMOD@	1A7F CKOPEN@	1568 CONF IG\$	203F
CONT1	28E1 CORE\$	0300 CR	000D

CRTBGN\$	F800	CYL1	2793	CYLEND	27AC
CYLPRM	277A	CYL GRN	16AE	D0FBYT8	1A26
DATE	2492	DATE\$	0033	DATEND	24A9
DAYTBL\$	04C7	DBGSV\$	00A0	DCBKL\$	0031
DCT\$	0470	DCTBYT80	1A29	DCTEXT	2AA0
DCTFLD0	1A34	DDCT1	287A	DDCT2	2884
DELAY	27E4	DELAY1	27F3	DFLAG\$	006D
DIRBUF\$	2300	DISABL	27AC	DIS DO RAM	0846
DODATA\$	0B94	DODCB\$	0210	D0FF	249F
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	DRIVE	2742	DRVLOOP	2765
DSKTYP\$	04C0	DTPMT\$	04C2	DVR2	28A7
DVR4	28D9	DVRA	2885	DVRB	288E
DVRC	289A	DVREND\$	0FF4	DVRHI\$	0206
DVRPAM	2835	EFLAG\$	006E	EIPMPT	2A21
ENABLE	27BD	ENADIS	27C6	ENADIS DO RAM	0817
EXTDBG\$	19A4	FCB1	2B9D	FDDINT\$	000E
FEMSK\$	006F	FLGTAB\$	006A	FARM	2441
FPEND	2465	FRCERR	241A	GETCFG	295F
GETHI\$	296F	GETN1	2844	GETN2	285E
GETN3	2863	GETSEC2	273B	GET @ ROWCOL	0DAE
GOTMEM?	2977	GRAEND	258C	GRAPHI	2577
GROFF	2588	GUDBS	24D3	GUDCYL	27A9
GUDSYS	2620	HERTZ\$	0750	HERTZ50	255D
HERTZ60	256A	HIGH\$	040E	HKRES\$	1A6C
HZ50END	256A	HZ60END	2577	IFLAG\$	0072
INBUF\$	0420	INTIM\$	003C	INTMSK\$	003D
INTVC\$	003E	IOERR	241F	JCLCB\$	0203
JDCB\$	0024	JFCB\$	00C0	JLDCB\$	0230
JRET\$	0026	KCK0	07D6	KFLAG\$	0074
KIDATA\$	08FC	KIDCB\$	0208	LARGE	24FA
LBANK\$	0202	LDRV\$	0023	LFLAG\$	0075
LNKFCB0	1566	LOW\$	001E	LSVC\$	000D
MAXCOR\$	2400	MAXDAYS	0401	MINCOR\$	3000
MODOUT\$	0076	MODPORT	00EC	MODRES\$	2A02
MONTBL\$	04DC	MOVITIN	2659	NDSYS\$	2A60
NFLAG\$	0077	NOCYL8\$	29A6	NOMEM	2410
NOMEM\$	297C	NOSET	2947	NOTRES	2736
NWSYS\$	2A45	OPREG\$	0078	OPREG SV AREA	086E
OPREG SV_PTR	0835	ORARET0	14DC	OSRLS\$	003B
OSVER\$	0085	OVRLY\$	0069	PAKNAM\$	0410
PATCH1	2B84	PAUSE0	0382	PCSAVE\$	07AF
PDRV\$	001B	PHIGH\$	001C	PRDCB\$	0218
PRMERR	2414	PRMERR\$	2996	PRMTBL\$	2AA3
PUTA@DE	0DCD	PUTCFG	295C	PUTDVR	26A0
PUTGETC	2964	PUTGETE	296B	PUTSEC2	282E
PUT @	0DCA	PUT @ ROWCOL	0DC6	REND	2529
RESI	264C	RESIE	2658	RESBGN	26CD
RES01	2707	RESDVR	26FC	RESEND	273B
RESTAB\$	26DC	RFLAG\$	007B	ROWCOL 2 ADDR	0DD0
RPARM	2512	RST380	1BFF	RSTOR\$	04C4
RSTRON	251E	RWRIT0	13A2	SIDCB\$	0238
SAVESP	2409	SBUFF\$	1D00	SET@EXEC	1A79
SETBLK	250C	SETCLK	245C	SETERR	2417
SETSLOW	2454	SET_SCROLL	0CF3	SFCB\$	008C
SFLAG\$	007C	SIDCB\$	0220	SLOW	243A
SMALPRM	2504	SMEND	255D	SMON	2554
SMOOTH	253E	SODCB\$	0228	SPACE4\$	2142
STACK\$	0380	START\$	0000	STEP	27C9

STEPEND	27E4	SVCRET\$	000B	SVCTAB\$	0100
SYSERR\$	1B13	SYSP1	28EF	SYSP2	2911
SYSP3	2929	SYSPRM	28DA	SYSRES	25FA
SYSTEM	2400	SYSTEM1	242A	TCB\$	004E
TEST6	2614	TFLAG\$	007D	TIME	24A9
TIME\$	002D	TIMEND	24C0	TIMER\$	002C
TIMSL\$	002B	TIMTSK\$	0713	TMPMT\$	04C3
TOFF	24B6	TRACE	2465	TRACE_INT	07B1
TREND	247E	TROFF	2479	TYPE	2529
TYPEND	253E	TYPEON	253A	TYPHK\$	0A8F
TYPTSK\$	0B26	USTOR\$	0013	VFLAG\$	007F
WDESD	2814	WPARM	2765	WPEND	277A
WPOFF	2776	WPON	2770	WRINT\$	0080
ZERO\$	0401	ZEROA0	13A0	@ABORT	6BFF
@@ADTSK	6C92	@@BANK	71AA	@@BKSP	6E8A
@@BREAK	71C0	@@CHNIO	6BEA	@@CKBRKC	720E
@@CKDRV	6CE6	@@CKEOF	6E9F	@@CKTSK	6C7D
@@CLOSE	6E75	@@CLS	71F8	@@CMNDI	6C29
@@CMNDR	6C3E	@@CTL	6A4E	@@DATE	6BC0
@@DCSTAT	6D25	@@DEBUG	6C68	@@DECHEX	712A
@@DIRRD	7097	@@DIRWR	70AC	@@DIV16	7115
@@DIV8	7100	@@DODIR	6CFB	@@DSP	6A12
@@DSPLY	6AB2	@@ERROR	6C53	@@EXIT	6C14
@@FEXT	7004	@@FLAGS	7194	@@FNAME	7019
@@FSPEC	6FEF	@@GATRD	7082	@@GATWR	70C1
@@GET	6A26	@@GTDCB	7043	@@GTDCT	702E
@@GTMOD	7058	@@HDFMT	6DCD	@@HEX16	7169
@@HEX8	7154	@@HEXDEC	713F	@@HIGH\$	717E
@@INIT	6E4B	@@KBD	6A8A	@@KEY	69FE
@@KEYIN	6A9E	@@KLTSK	6CD1	@@LOAD	6FC5
@@LOC	6EB4	@@LOF	6EC9	@@LOGGER	6AE9
@@LOGOT	6AFE	@@MSG	6B35	@@MUL16	70EB
@@MUL8	70D6	@@OPEN	6E60	@@PARAM	6BAB
@@PAUSE	6B96	@@PEOF	6EDE	@@POSN	6EF3
@@PRINT	6B4A	@@PRT	6A62	@@PUT	6A3A
@@RAMDIR	6D10	@@RDSEC	6DA3	@@RDSSC	706D
@@READ	6F08	@@REMOV	6E36	@@RENAM	6E21
@@REW	6F1D	@@RMTSK	6CA7	@@RPTSK	6CBC
@@RREAD	6F32	@@RSLCT	6D8E	@@RSTOR	6D4F
@@RUN	6FDA	@@RWIT	6F47	@@SEEK	6D79
@@SEEKSC	6F5C	@@SKIP	6F71	@@SLCT	6D3A
@@STEPI	6D64	@@TIME	6BD5	@@VDCTL	6B81
@@VER	6F86	@@VRSEC	6DB8	@@WEOF	6F9B
@@WHERE	6A76	@@WRITE	6FB0	@@WRSEC	6DE2
@@WRSSC	6DF7	@@WRTRK	6E0C		

2400 is the transfer address

00000 Total errors

NOTES:

NOTES:

NOTES:

NOTES:

NOTES:

```

00100 ;LDOS60/EQU - Equates from cross reference of Lowcore
0000 00110 TITLE <LDOS60/EQU>
00120 ;
08F0 00130 @$SYS EQU 08F0H
0000 00140 @01 DEFL 0000H
0000 00150 @02 DEFL 0000H
0000 00160 @03 DEFL 0000H
0000 00170 @04 DEFL 0000H
0877 00180 @BANK EQU 0877H
1300 00190 @BYTE IO EQU 1300H
0689 00200 @CHNIO EQU 0689H
0553 00210 @CKBRKC EQU 0553H
0545 00220 @CLS EQU 0545H
0623 00230 @CTL EQU 0623H
07A8 00240 @DATE EQU 07A8H
06E3 00250 @DIV16 EQU 06E3H
0642 00260 @DSP EQU 0642H
052D 00270 @DSPLY EQU 052DH
0000 00280 @FRENCH EQU 0000H
0000 00290 @GERMAN EQU 0000H
0638 00300 @GET EQU 0638H
07BD 00310 @HEX16 EQU 07BDH
07C2 00320 @HEX8 EQU 07C2H
06F6 00330 @HEXDEC EQU 06F6H
0000 00340 @HZ50 EQU 0000H
0000 00350 @INTL EQU 0000H
0630 00360 @JCL EQU 0630H
0635 00370 @KBD EQU 0635H
0628 00380 @KEY EQU 0628H
0585 00390 @KEYIN EQU 0585H
0089 00400 @KITSK EQU 0089H
0503 00410 @LOGGER EQU 0503H
0500 00420 @LOGOT EQU 0500H
0000 00430 @MOD2 EQU 0000H
FFFF 00440 @MOD4 EQU 0FFFFH
0530 00450 @MSG EQU 0530H
06C9 00460 @MUL16 EQU 06C9H
0084 00470 @OPREG EQU 0084H
0528 00480 @PRINT EQU 0528H
063D 00490 @PRT EQU 063DH
0645 00500 @PUT EQU 0645H
0FE9 00510 @RSTNMI EQU 0FE9H
0680 00520 @RSTREG EQU 0680H
078D 00530 @TIME EQU 078DH
FFFF 00540 @USA EQU 0FFFFH
0B99 00550 @VDCTL EQU 0B99H
0D38 00560 @VDCTL3 EQU 0D38H
0D42 00570 @_VDCTL EQU 0D42H
0DF1 00580 ADDR_2 ROWCOL EQU 0DF1H
0201 00590 BAR$ 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	DSKTP\$ 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	KCK0 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	PAUSE0 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	SIDCB\$ 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

No end statement

00000 Total errors

```

00100 ;SYS0/EQU - Equates from cross reference of Sysres
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	@LOGGER	EQU	0503H
0500	00960	@LOGOT	EQU	0500H
0000	00970	@MOD2	EQU	0000H
FFFF	00980	@MOD4	EQU	FFFFFH
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	@RPSTK	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	F800H
16AE	01790 CYL_GRN EQU	16AEH
1A26	01800 D@F8BYT8 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	0BCH
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	SIDCB\$ 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 ;SVC MAC/ASM - LS-DOS Version VI
00110 TITLE <SVC MAC - MACRO EQUIVALENTS>
00120 ;*LIST OFF
00130 ;
00140 @MOD2 EQU 0
FFFF 00150 @MOD4 EQU -1
00160 @KEY MACRO
00170 LD A,1
00180 RST 40
00190 ENDM
00200 @@DSP MACRO
00210 LD A,2
00220 RST 40
00230 ENDM
00240 @@GET MACRO
00250 LD A,3
00260 RST 40
00270 ENDM
00280 @@PUT MACRO
00290 LD A,4
00300 RST 40
00310 ENDM
00320 @@CTL MACRO
00330 LD A,5
00340 RST 40
00350 ENDM
00360 @@PRT MACRO
00370 LD A,6
00380 RST 40
00390 ENDM
00400 @@WHERE MACRO
00410 LD A,7
00420 RST 40
00430 ENDM
00440 @@KBD MACRO
00450 LD A,8
00460 RST 40
00470 ENDM
00480 @@KEYIN MACRO
00490 LD A,9
00500 RST 40
00510 ENDM
00520 @@DSPLY MACRO #MSG
00530 IFEQ %,1
00540 LD HL,#MSG
00550 ENDF
00560 LD A,10
00570 RST 40
00580 ENDM
00590 @@LOGGER MACRO
00600 LD A,11
00610 RST 40
00620 ENDM
00630 @@LOGOT MACRO #MSG
00640 IFEQ %,1
00650 LD HL,#MSG
00660 ENDF
00670 LD A,12
00680 RST 40
00690 ENDM
00700 @@MSG MACRO

```

0000	00710	LD	A,13
0000	00720	RST	40
0000	00730	ENDM	
0000	00740	@@PRINT MACRO	#MSG
0000	00750	IFEQ	%,1
0000	00760	LD	HL,#MSG
0000	00770	ENDIF	
0000	00780	LD	A,14
0000	00790	RST	40
0000	00800	ENDM	
0000	00810	@@VDCTL MACRO	
0000	00820	LD	A,15
0000	00830	RST	40
0000	00840	ENDM	
0000	00850	@@PAUSE MACRO	
0000	00860	LD	A,16
0000	00870	RST	40
0000	00880	ENDM	
0000	00890	@@PARAM MACRO	
0000	00900	LD	A,17
0000	00910	RST	40
0000	00920	ENDM	
0000	00930	@@DATE MACRO	
0000	00940	LD	A,18
0000	00950	RST	40
0000	00960	ENDM	
0000	00970	@@TIME MACRO	
0000	00980	LD	A,19
0000	00990	RST	40
0000	01000	ENDM	
0000	01010	@@CHNIO MACRO	
0000	01020	LD	A,20
0000	01030	RST	40
0000	01040	ENDM	
0000	01050	@@ABORT MACRO	
0000	01060	LD	A,21
0000	01070	RST	40
0000	01080	ENDM	
0000	01090	@@EXIT MACRO	
0000	01100	LD	A,22
0000	01110	RST	40
0000	01120	ENDM	
0000	01130	@@CMNDI MACRO	
0000	01140	LD	A,24
0000	01150	RST	40
0000	01160	ENDM	
0000	01170	@@CMNDR MACRO	
0000	01180	LD	A,25
0000	01190	RST	40
0000	01200	ENDM	
0000	01210	@@ERROR MACRO	
0000	01220	LD	A,26
0000	01230	RST	40
0000	01240	ENDM	
0000	01250	@@DEBUG MACRO	
0000	01260	LD	A,27
0000	01270	RST	40
0000	01280	ENDM	
0000	01290	@@CKTSK MACRO	
0000	01300	LD	A,28
0000	01310	RST	40

0000	01320	ENDM	
0000	01330	@@ADTSK	MACRO
0000	01340	LD	A, 29
0000	01350	RST	40
0000	01360	ENDM	
0000	01370	@@RMTSK	MACRO
0000	01380	LD	A, 30
0000	01390	RST	40
0000	01400	ENDM	
0000	01410	@@RPTSK	MACRO
0000	01420	LD	A, 31
0000	01430	RST	40
0000	01440	ENDM	
0000	01450	@@KLTSK	MACRO
0000	01460	LD	A, 32
0000	01470	RST	40
0000	01480	ENDM	
0000	01490	@@CKDRV	MACRO
0000	01500	LD	A, 33
0000	01510	RST	40
0000	01520	ENDM	
0000	01530	@@DODIR	MACRO
0000	01540	LD	A, 34
0000	01550	RST	40
0000	01560	ENDM	
0000	01570	@@RAMDIR	MACRO
0000	01580	LD	A, 35
0000	01590	RST	40
0000	01600	ENDM	
0000	01610	@@DCSTAT	MACRO
0000	01620	LD	A, 40
0000	01630	RST	40
0000	01640	ENDM	
0000	01650	@@SLCT	MACRO
0000	01660	LD	A, 41
0000	01670	RST	40
0000	01680	ENDM	
0000	01690	@@RSTOR	MACRO
0000	01700	LD	A, 44
0000	01710	RST	40
0000	01720	ENDM	
0000	01730	@@STEPI	MACRO
0000	01740	LD	A, 45
0000	01750	RST	40
0000	01760	ENDM	
0000	01770	@@SEEK	MACRO
0000	01780	LD	A, 46
0000	01790	RST	40
0000	01800	ENDM	
0000	01810	@@RSLCT	MACRO
0000	01820	LD	A, 47
0000	01830	RST	40
0000	01840	ENDM	
0000	01850	@@RDSEC	MACRO
0000	01860	LD	A, 49
0000	01870	RST	40
0000	01880	ENDM	
0000	01890	@@VRSEC	MACRO
0000	01900	LD	A, 50
0000	01910	RST	40
0000	01920	ENDM	

0000	01930	@@HDFMT	MACRO	
0000	01940		LD	A,52
0000	01950		RST	40
0000	01960		ENDM	
0000	01970	@@WRSEC	MACRO	
0000	01980		LD	A,53
0000	01990		RST	40
0000	02000		ENDM	
0000	02010	@@WRSSC	MACRO	
0000	02020		LD	A,54
0000	02030		RST	40
0000	02040		ENDM	
0000	02050	@@WRTRK	MACRO	
0000	02060		LD	A,55
0000	02070		RST	40
0000	02080		ENDM	
0000	02090	@@RENAM	MACRO	
0000	02100		LD	A,56
0000	02110		RST	40
0000	02120		ENDM	
0000	02130	@@REMOV	MACRO	
0000	02140		LD	A,57
0000	02150		RST	40
0000	02160		ENDM	
0000	02170	@@INIT	MACRO	
0000	02180		LD	A,58
0000	02190		RST	40
0000	02200		ENDM	
0000	02210	@@OPEN	MACRO	
0000	02220		LD	A,59
0000	02230		RST	40
0000	02240		ENDM	
0000	02250	@@CLOSE	MACRO	
0000	02260		LD	A,60
0000	02270		RST	40
0000	02280		ENDM	
0000	02290	@@BKSP	MACRO	
0000	02300		LD	A,61
0000	02310		RST	40
0000	02320		ENDM	
0000	02330	@@CKEOF	MACRO	
0000	02340		LD	A,62
0000	02350		RST	40
0000	02360		ENDM	
0000	02370	@@LOC	MACRO	
0000	02380		LD	A,63
0000	02390		RST	40
0000	02400		ENDM	
0000	02410	@@LOF	MACRO	
0000	02420		LD	A,64
0000	02430		RST	40
0000	02440		ENDM	
0000	02450	@@PEOF	MACRO	
0000	02460		LD	A,65
0000	02470		RST	40
0000	02480		ENDM	
0000	02490	@@POSN	MACRO	
0000	02500		LD	A,66
0000	02510		RST	40
0000	02520		ENDM	
0000	02530	@@READ	MACRO	

0000	02540	LD	A,67
0000	02550	RST	40
0000	02560	ENDM	
0000	02570 @@REW	MACRO	
0000	02580	LD	A,68
0000	02590	RST	40
0000	02600	ENDM	
0000	02610 @@RREAD	MACRO	
0000	02620	LD	A,69
0000	02630	RST	40
0000	02640	ENDM	
0000	02650 @@RWRIT	MACRO	
0000	02660	LD	A,70
0000	02670	RST	40
0000	02680	ENDM	
0000	02690 @@SEEKSC	MACRO	
0000	02700	LD	A,71
0000	02710	RST	40
0000	02720	ENDM	
0000	02730 @@SKIP	MACRO	
0000	02740	LD	A,72
0000	02750	RST	40
0000	02760	ENDM	
0000	02770 @@VER	MACRO	
0000	02780	LD	A,73
0000	02790	RST	40
0000	02800	ENDM	
0000	02810 @@WEOF	MACRO	
0000	02820	LD	A,74
0000	02830	RST	40
0000	02840	ENDM	
0000	02850 @@WRITE	MACRO	
0000	02860	LD	A,75
0000	02870	RST	40
0000	02880	ENDM	
0000	02890 @@LOAD	MACRO	
0000	02900	LD	A,76
0000	02910	RST	40
0000	02920	ENDM	
0000	02930 @@RUN	MACRO	
0000	02940	LD	A,77
0000	02950	RST	40
0000	02960	ENDM	
0000	02970 @@FSPEC	MACRO	
0000	02980	LD	A,78
0000	02990	RST	40
0000	03000	ENDM	
0000	03010 @@FEXT	MACRO	
0000	03020	LD	A,79
0000	03030	RST	40
0000	03040	ENDM	
0000	03050 @@FNAME	MACRO	
0000	03060	LD	A,80
0000	03070	RST	40
0000	03080	ENDM	
0000	03090 @@GTDCT	MACRO	
0000	03100	LD	A,81
0000	03110	RST	40
0000	03120	ENDM	
0000	03130 @@GTDCB	MACRO	
0000	03140	LD	A,82

0000	03150	RST	40
0000	03160	ENDM	
0000	03170	@@GTMOD MACRO	
0000	03180	LD	A,83
0000	03190	RST	40
0000	03200	ENDM	
0000	03210	@@RDSSC MACRO	
0000	03220	LD	A,85
0000	03230	RST	40
0000	03240	ENDM	
0000	03250	@@GATRD MACRO	
0000	03260	LD	A,86
0000	03270	RST	40
0000	03280	ENDM	
0000	03290	@@DIRRD MACRO	
0000	03300	LD	A,87
0000	03310	RST	40
0000	03320	ENDM	
0000	03330	@@DIRWR MACRO	
0000	03340	LD	A,88
0000	03350	RST	40
0000	03360	ENDM	
0000	03370	@@GATWR MACRO	
0000	03380	LD	A,89
0000	03390	RST	40
0000	03400	ENDM	
0000	03410	@@MUL8 MACRO	
0000	03420	LD	A,90
0000	03430	RST	40
0000	03440	ENDM	
0000	03450	@@MUL16 MACRO	
0000	03460	LD	A,91
0000	03470	RST	40
0000	03480	ENDM	
0000	03490	@@DIV8 MACRO	
0000	03500	LD	A,93
0000	03510	RST	40
0000	03520	ENDM	
0000	03530	@@DIV16 MACRO	
0000	03540	LD	A,94
0000	03550	RST	40
0000	03560	ENDM	
0000	03570	@@DECHEX	MACRO
0000	03580	LD	A,96
0000	03590	RST	40
0000	03600	ENDM	
0000	03610	@@HEXDEC	MACRO
0000	03620	LD	A,97
0000	03630	RST	40
0000	03640	ENDM	
0000	03650	@@HEX8 MACRO	
0000	03660	LD	A,98
0000	03670	RST	40
0000	03680	ENDM	
0000	03690	@@HEX16 MACRO	
0000	03700	LD	A,99
0000	03710	RST	40
0000	03720	ENDM	
0000	03730	@@HIGH\$ MACRO	
0000	03740	LD	A,100
0000	03750	RST	40

The Source	LIBRARY Files	SVMAC - MACRO EQUIVALENTS	Page 00007
0000	03760	ENDM	
0000	03770 @@FLAGS	MACRO	
0000	03780	LD	A,101
0000	03790	RST	40
0000	03800	ENDM	
0000	03810 @@BANK	MACRO	
0000	03820	LD	A,102
0000	03830	RST	40
0000	03840	ENDM	
0000	03850 @@BREAK	MACRO	#ADR
0000	03860	IFEQ	%,1
0000	03870	LD	HL,#ADR
0000	03880	ENDIF	
0000	03890	LD	A,103
0000	03900	RST	40
0000	03910	ENDM	
0000	03920 @@CLS	MACRO	
0000	03930	LD	A,105
0000	03940	RST	40
0000	03950	ENDM	
0000	03960 @@CKBRKC	MACRO	
0000	03970	LD	A,106
0000	03980	RST	40
0000	03990	ENDM	
0000	04000 *LIST	ON	
0000	04010	END	
00000	Total errors		

