SOURCE ACCESS NAME= DLU09.KSG.HEXBUS.DILLO.AAAA
OBJECT ACCESS NAME= DLU09.KSG.HEXBUS.DILLO.OBJREL3
LISTING ACCESS NAME= DLU09.KSG.HEXBUS.DILLO.LSTREL3
ERROR ACCESS NAME= DLU09.KSG.HEXBUS.DILLO.ERRREL3
OPTIONS= XREF,BUNLST,TUNLST,DUNLST
MACRO LIBRARY PATHNAME=

LINE	KEY	NAME
0001	A	DLU09.KSG.HEXBUS.DILLO.EQUATES =>DLU09.KSG.HEXBUS.DILLO.EQUATES
0002	B	DLU09.KSG.HEXBUS.DILLO.LINKAGE =>DLU09.KSG.HEXBUS.DILLO.LINKAGE
0003	C	DLU09.KSG.HEXBUS.DILLO.POWERUP =>DLU09.KSG.HEXBUS.DILLO.POWERUP
0004	D	DLU09.KSG.HEXBUS.DILLO.MAINENT =>DLU09.KSG.HEXBUS.DILLO.MAINENT
0005	E	DLU09.KSG.HEXBUS.DILLO.PARSER =>DLU09.KSG.HEXBUS.DILLO.PARSER
0006	F	DLU09.KSG.HEXBUS.DILLO.OPEN =>DLU09.KSG.HEXBUS.DILLO.OPEN
0007	G	DLU09.KSG.HEXBUS.DILLO.CLOSE =>DLU09.KSG.HEXBUS.DILLO.CLOSE
0008	H	DLU09.KSG.HEXBUS.DILLO.READWRIT =>DLU09.KSG.HEXBUS.DILLO.READWRIT
0009	I	DLU09.KSG.HEXBUS.DILLO.SAVELOAD =>DLU09.KSG.HEXBUS.DILLO.SAVELOAD
0010	J	DLU09.KSG.HEXBUS.DILLO.RETTYDEL =>DLU09.KSG.HEXBUS.DILLO.RETTYDEL
0011	K	DLU09.KSG.HEXBUS.DILLO.STATUS =>DLU09.KSG.HEXBUS.DILLO.STATUS
0012	L	DLU09.KSG.HEXBUS.DILLO.VDPCPU =>DLU09.KSG.HEXBUS.DILLO.VDPCPU
0013	M	DLU09.KSG.HEXBUS.DILLO.EXITS =>DLU09.KSG.HEXBUS.DILLO.EXITS
0014	N	DLU09.KSG.HEXBUS.DILLO.XMITRECV =>DLU09.KSG.HEXBUS.DILLO.XMITRECV
0015	O	DLU09.KSG.HEXBUS.DILLO.PREOUTIN =>DLU09.KSG.HEXBUS.DILLO.PREOUTIN
0016	P	DLU09.KSG.HEXBUS.DILLO.MISC =>DLU09.KSG.HEXBUS.DILLO.MISC
0017	Q	DLU09.KSG.HEXBUS.DILLO.INTRPT =>DLU09.KSG.HEXBUS.DILLO.INTRPT
0018	R	DLU09.KSG.HEXBUS.DILLO.RESET =>DLU09.KSG.HEXBUS.DILLO.RESET
0019	S	DLU09.KSG.HEXBUS.DILLO.RAWDATA =>DLU09.KSG.HEXBUS.DILLO.RAWDATA
0020	T	DLU09.KSG.HEXBUS.DILLO.CATALOG =>DLU09.KSG.HEXBUS.DILLO.CATALOG
0021	U	DLU09.KSG.HEXBUS.DILLO.SLAVE =>DLU09.KSG.HEXBUS.DILLO.SLAVE
0022	V	DLU09.KSG.HEXBUS.DILLO.DISKRT =>DLU09.KSG.HEXBUS.DILLO.DISKRT
0023	W	DLU09.KSG.HEXBUS.DILLO.DISKIO =>DLU09.KSG.HEXBUS.DILLO.DISKIO

```
 0001                    COPY DLU09.KSG.HEXBUS.DILLO.EQUATES
A0001           ***************************************************************
A0002           *
A0003           *       HEXBUS DSR for TI 99/8 Computer System
A0004           *
A0005           *       VERSION 1.2     09/16/83                              2
A0006           *
A0007           *
A0008           *       This DSR accepts the following device names:
A0009           *       HEXBUS, RS232, RS232/1, RS232/2,
A0010           *       DSK, DSK1, DSK2, DSK3, DSK4
A0011           *
A0012           *       The HEXBUS I/F (OSO) chip is used as an interface to
A0013           *       the HEXBUS.  The following describes its configura-
A0014           *       tion and use:
A0015           *
A0016           *            The OSO uses locations >5FF8 through >5FFF for
A0017           *            memory mapped I/O.  The chip automatically con-
A0018           *            trols the HSK(HANDSHAKE) and BAV(BUS_AVAILABLE)
A0019           *            lines on the HEXBUS.
A0020           *
A0021           *            STATUS REGISTER:  (read @>5FFA)
A0022           *               Bit 0: HSKWT - Set when a byte has been sent
A0023           *                      over the bus and HSK has risen again
A0024           *                      (i.e., the byte has been accepted).
A0025           *               Bit 1: HSKRD - Set when an entire byte has
A0026           *                      been recieved in the Read Data Reg.
A0027           *               Bit 2: BAVIAS - Set when BAV drops.
A0028           *               Bit 3: BAVAIS - Set when BAV rises.
A0029           *               Bit 4: SBAV - Set if BAV is active (low).
A0030           *               Bit 5: WBUSY - Set while a byte is being
A0031           *                      written to the bus.  Reset when HSKWT
A0032           *                      is set (after HSK rises again).
A0033           *               Bit 6: RBUSY - Set while a byte is being read
A0034           *                      from the bus. Reset when HSKRD is set.
A0035           *               Bit 7: SHSK - Set if HSK is active (low).
A0036           *               NOTE: Bits 0-3 are cleared by reading STATUS.
A0037           *
A0038           *           CONTROL REGISTER:  (write @>5FFA, read @>5FFC)
A0039           *               Bit 0: WEIN - Interrupt enable for completing
A0040           *                      a write to the HEXBUS. (NOT USED)
A0041           *               Bit 1: REIN - Interrupt enable for completing
A0042           *                      a read from the HEXBUS. (NOT USED)
A0043           *               Bit 2: BAVIAEN - Interrupt enable for BAV
A0044           *                      falling.  This is used to sense a
A0045           *                      SERVICE REQUEST from a device.
A0046           *               Bit 3: BAVAIEN - Interrupt enable for BAV
A0047           *                      rising.  This is used in SLAVE MODE
A0048           *                      to sense an end of message.
A0049           *               Bit 4: BAVC - Used to control BAV line.  Set
A0050           *                      this bit to drop BAV; reset to raise.
A0051           *               Bit 5: WEN - Write Enable. Enables chip to
A0052           *                      write a byte to the bus whenever a
A0053           *                      byte is written to the transmit reg.
A0054           *               Bit 6: REN - Read Enable. Enables chip to
A0055           *                      latch HSK and read a byte into the
A0056           *                      read data register.
A0057           *               Bit 7: CR7 - Not used, currently.  Will be
A0058           *                      used to manually hold HSK low in
A0059           *                      SLAVE MODE, between message and re-
```

```
A0060           *                    sponse.  Code shows that, when set,
A0061           *                    HSK should be held low.
A0062           *
A0063           *          READ DATA REGISTER: (read @>5FF8)
A0064           *             This register contains the byte received from
A0065           *             the bus.  HSK is held low until this location
A0066           *             is read. Then, HSK is automatically released.
A0067           *
A0068           *          TRANSMIT DATA REGISTER:
A0069           *                                (write @>5FF8; read @>5FFE)
A0070           *             Write to this register (while WEN=1) to write
A0071           *             a byte to the bus.
A0072           *
A0073           *          The chip automatically reads or writes LSNibble,
A0074           *          followed by MSNibble of each byte.  The DSR or
A0075           *          application must explicity send or receive the
A0076           *          LSByte, followed by the MSByte when reading or
A0077           *          writing a word at a time.
A0078           *
A0079           *     Register Usage Restrictions:
A0080           *          1) DSR proper may not use R0 - R2 while deciding
A0081           *             if it is the proper DSR.
A0082           *          2) DSR proper may use R0 through R10.
A0083           *          3) LSByte of R3 distinguishes between entry from
A0084           *             File Mgt. System or Assem. Branch Vector.
A0085           *             (00/FF)
A0086           *
A0087           *                                          Kenneth Gregg
A0088           ***************************************************************
```

```
A0090                ****************************************************************
A0091                *
A0092                *       MISCELLANEOUS EQUATES
A0093                *
A0094   8C02  VWA     EQU   >8C02              VDP write address
A0095   FBFE  VRD     EQU   >8800-VWA          VDP read data
A0096   FFFE  VWD     EQU   >8C00-VWA          VDP write data
A0097   0600  TMOUT1  EQU   >0600              20 MS delay  (PRECOM)
A0098   0300  TMOUT4  EQU   >0300              20 MS delay
A0099   0020  MEDFUL  EQU   32                 Media Full Error
A0100   00FE  SLVILL  EQU   254                Illegal in Slave Mode
A0101   0020  CHKBRK  EQU   >0020              Console routine to test for F-4
A0102                *
A0103                ****************************************************************
A0104                *
A0105                *       CIRCULAR BUFFER POINTERS (RS232 type device)
A0106                *
A0107   0000  CBUFST  EQU   >00                 Start address (word)
A0108   0002  CBUFLN  EQU   >02                 Lengtn of buffer (byte)
A0109   0003  CBUFRO  EQU   >03                 Read offset (DSR read only)
A0110   0004  CBUFOF  EQU   >04                 Write offset (byte)
A0111                *
A0112                ****************************************************************
A0113                *
A0114                *       RAM ALLOCATION INFORMATION
A0115                *       Block reserved in high RAM, mapped at >E000
A0116                *
A0117   8810  MAPPER  EQU   >8810                       Location of mapper
A0118   00FF  MEMSIZ  EQU   >00FF        255 BYTES      Number of bytes reserved
A0119   E008  RAMBEG  EQU   >E008        START ADR      Mapped RAM start address
A0120   E008  LEVEL1  EQU   RAMBEG+0     [E008-09]      Level 1 return address
A0121   E00A  LEVEL2  EQU   RAMBEG+2     [E00A-0B]      Level 2 return address
A0122   E00C  LEVEL3  EQU   RAMBEG+4     [E00C-0D]      Level 3 return address
A0123   E00E  OPENBF  EQU   RAMBEG+6     [E00E-11]      4 byte buffer for open
A0124   E012  LUN250  EQU   RAMBEG+10    [E012-28]      LUNO block 250 (DISK)
A0125   E029  LUN251  EQU   RAMBEG+33    [E029-3F]      LUNO block 251 (DISK)
A0126   E040  LUN252  EQU   RAMBEG+56    [E040-56]      LUNO block 252 (DISK)
A0127   E057  LUN253  EQU   RAMBEG+79    [E057-6D]      LUNO block 253 (DISK)
A0128   E06E  BLOKAV  EQU   RAMBEG+102   [E06E-6F]      Block available address
A0129   E070  DSKLUN  EQU   RAMBEG+104   [E070-71]      Disk file LUNO value
A0130   E072  SRVREQ  EQU   RAMBEG+106   [E072]         Service request byte
A0131   E073  SLAST   EQU   RAMBEG+107   [E073]         Slave HSK ctrl. flag
A0132                *
A0133                ****************************************************************
```

```
A0135           ****************************************************************
A0136           *
A0137           *       HEXBUS (OSO) REGISTERS AND MASKS
A0138           *
A0139  5FF8  OSOMAP EQU  >5FF8      OSO chip memory mapped address base
A0140  5FF8  HEXRDD EQU  OSOMAP+0  [R] HEXBUS Read data register
A0141  5FFA  HEXSTS EQU  OSOMAP+2  [R] HEXBUS Status register
A0142  5FFC  HEXCTR EQU  OSOMAP+4  [R] HEXBUS Control register
A0143  5FFE  HEXMTR EQU  OSOMAP+6  [R] HEXBUS Xmit data register
A0144  5FF8  HEXMTW EQU  OSOMAP+0  [W] HEXBUS Xmit data register
A0145  5FFA  HEXCTW EQU  OSOMAP+2  [W] HEXBUS Control register
A0146  8000  HSKWT  EQU  >8000          STATUS HSK write completion
A0147  4000  HSKRD  EQU  >4000          STATUS HSK read service req.
A0148  2000  BAVIAS EQU  >2000          STATUS BAV trans: inact=>act
A0149  1000  BAVAIS EQU  >1000          STATUS BAV trans: act=>inact
A0150  0800  SBAV   EQU  >0800          STATUS BAV (1=> BAV active)
A0151  0400  WBUSY  EQU  >0400          STATUS Write busy (1=>active)
A0152  0200  RBUSY  EQU  >0200          STATUS Read busy (1=>active)
A0153  0100  SHSK   EQU  >0100          STATUS HSK (1=> HSK active) .
A0154           *
A0155           ****************************************************************
A0156           *
A0157           *       SCRATCHPAD RAM ASSIGNMENT (FAC AREA)
A0158           *       CPU PAB configuration used to store user's PAB
A0159           *
A0160  004A  FAC    EQU  >4A                        Beginning of 36 byte FAC
A0161  004A  OPCODE EQU  FAC+0  [4A] CPU PAB - I/O operation code
A0162  004B  FLGSTS EQU  FAC+1  [4B] CPU PAB - Flag byte
A0163  004C  BUFADR EQU  FAC+2  [4C] CPU PAB - Data buffer address
A0164  004E  CHRCNT EQU  FAC+4  [4E] CPU PAB - Character count
A0165  0050  RECNUM EQU  FAC+6  [50] CPU PAB - Record number
A0166  0052  LRECLN EQU  FAC+8  [52] CPU PAB - Logical record length
A0167  0054  SCNOFF EQU  FAC+10 [54] CPU PAB - Screen offset
A0168  0054  RAMTST EQU  FAC+10 [54] CPU PAB - Returned PAB type flag
A0169  0055  PABLUN EQU  FAC+11 [55] CPU PAB - Returns HEX Luno to user
A0170  0056  ERSTAT EQU  FAC+12 [56] CPU PAB - Error status byte
A0171  0057  NAMLEN EQU  FAC+13 [57] CPU PAB - Length of name string
A0172  0058  DEVLEN EQU  FAC+14 [58] CPU      Length of device name
A0173  005A  PABPTR EQU  FAC+16 [5A] CPU      Ptr. of end of dev.name
A0174  005C  ALCFLG EQU  FAC+18 [5C] HEX      HEXBUS open attributes
A0175  005D  DEVCOD EQU  FAC+19 [5D] HEX PAB - Device code
A0176  005E  COMMAD EQU  FAC+20 [5E] HEX PAB - Command code
A0177  005F  LUNO   EQU  FAC+21 [5F] HEX PAB - Logical unit number
A0178  0060  RECNO  EQU  FAC+22 [60] HEX PAB - Record number
A0179  0062  BUFLEN EQU  FAC+24 [62] HEX PAB - Buffer length
A0180  0064  WDATLN EQU  FAC+26 [64] HEX PAB - Data Length (write)
A0181  0066  RDATLN EQU  FAC+28 [66] HEX      Data Length (read)
A0182  0068  RECLEN EQU  FAC+30 [68] HEX      Record length at open
A0183  006A  COMSTS EQU  FAC+32 [6A] HEX      Comm. status returned
A0184           *
A0185           ****************************************************************
```

```
 0002                    COPY DLU09.KSG.HEXBUS.DILLO.LINKAGE
B0001                ***************************************************************
B0002                *
B0003                *       DSR HEADER
B0004                *
B0005 0000   AA          BYTE >AA                Valid DSR ID byte >AA
B0006 0001   01          BYTE >01                Version Number
B0007 0002 0000          DATA >0000              Reserved
B0008 0004 0020'         DATA PWRLNK             Link to Powerup routine
B0009 0006 0000          DATA 0                  Reserved
B0010 0008 0026'         DATA DSRLNK             Link to DSR list
B0011 000A 0084'         DATA SUBLNK             Link to Subprogram list
B0012 000C 00AE'         DATA INTLNK             Link to Interrupt routine
B0013 000E 0000          DATA 0                  Reserved
B0014                *
B0015                ***************************************************************
B0016                *
B0017                *       HEXBUS OSO CONTROL BITS
B0018                *
B0019 0010 F000  CLRINT  DATA >F000 BITS 0-3 : Used to disable interrupts
B0020 0012 0F00  DISABL  DATA >0F00 BITS 4-7 : Used to disable communication
B0021 0014 2000  BAVIAE  DATA >2000 BIT    2 : BAV I=>A interrupt enable
B0022 0016 1000  BAVAIE  DATA >1000 BIT    3 : BAV A=>I interrupt enable
B0023 0018 0800  BAVC    DATA >0800 BIT    4 : BAV control (1=>active)
B0024 001A 0400  WEN     DATA >0400 BIT    5 : Write enable
B0025 001C 0200  REN     DATA >0200 BIT    6 : Read enable
B0026 001E 0100  CR7     DATA >0100 BIT    7 : unused [WILL BE USED FOR HSK]
B0027                *
B0028                ***************************************************************
B0029                *
B0030                *       POWERUP ENTRY POINT LINKAGE
B0031                *
B0032 0020 0000  PWRLNK  DATA 0,PWRENT           Link, powerup entry point
B0033 0024   00          BYTE 0
B0034 0026               EVEN
B0035                *
B0036                ***************************************************************
```

```
B0038                   ************************************************************
B0039                   *
B0040                   *       DSR DEVICE NAME LIST
B0041                   *
B0042 0026 0030' DSRLNK DATA DSR2,RS232         RS232
B0043 002A   05         BYTE 5
B0044 002B   52         TEXT 'RS232'
B0045 0030              EVEN
B0046 0030 003C' DSR2   DATA DSR3,RS2321        RS232/1
B0047 0034   07         BYTE 7
B0048 0035   52         TEXT 'RS232/1'
B0049 003C              EVEN
B0050 003C 0048' DSR3   DATA DSR4,RS2322        RS232/2
B0051 0040   07         BYTE 7
B0052 0041   52         TEXT 'RS232/2'
B0053 0048              EVEN
B0054 0048 0054' DSR4   DATA DSR5,HEXBUS        HEXBUS
B0055 004C   06         BYTE 6
B0056 004D   48         TEXT 'HEXBUS'
B0057 0054              EVEN
B0058 0054 005C' DSR5   DATA DSR6,DSK           DSK
B0059 0058   03         BYTE 3
B0060 0059   44         TEXT 'DSK'
B0061 005C              EVEN
B0062 005C 0066' DSR6   DATA DSR7,DSK1          DSK1
B0063 0060   04         BYTE 4
B0064 0061   44         TEXT 'DSK1'
B0065 0066              EVEN
B0066 0066 0070' DSR7   DATA DSR8,DSK2          DSK2
B0067 006A   04         BYTE 4
B0068 006B   44         TEXT 'DSK2'
B0069 0070              EVEN
B0070 0070 007A' DSR8   DATA DSR9,DSK3          DSK3
B0071 0074   04         BYTE 4
B0072 0075   44         TEXT 'DSK3'
B0073 007A              EVEN
B0074 007A 0000  DSR9   DATA 0,DSK4             DSK4
B0075 007E   04         BYTE 4
B0076 007F   44         TEXT 'DSK4'
B0077 0084              EVEN
B0078                   *
B0079                   ************************************************************
```

```
B0081                    ************************************************************
B0082                    *
B0083                    *       DISK SUBROUTINE LINKAGE TABLES
B0084                    *
B0085 0084 008A' SUBLNK  DATA SBR0,RWSECT  SUB 10 - Read/write sector
B0086 0088   01          BYTE 1,>10
B0087 008A 0090' SBR0    DATA SBR1,DFORM   SUB 11 - Format diskette
B0088 008E   01          BYTE 1,>11
B0089 0090 0096' SBR1    DATA SBR2,FPROT   SUB 12 - Modify file protection
B0090 0094   01          BYTE 1,>12
B0091 0096 009C' SBR2    DATA SBR3,RNAME   SUB 13 - Rename file
B0092 009A   01          BYTE 1,>13
B0093 009C 00A2' SBR3    DATA SBR4,DINP    SUB 14 - Direct file input
B0094 00A0   01          BYTE 1,>14
B0095 00A2 00A8' SBR4    DATA SBR5,DOUTP   SUB 15 - Direct file output
B0096 00A6   01          BYTE 1,>15
B0097 00A8 0000  SBR5    DATA 0,RELOC      SUB 16 - Buffer relocation routine
B0098 00AC   01          BYTE 1,>16
B0099 00AE               EVEN              Call Files not needed on 99/8
B0100                    *
B0101                    ************************************************************
B0102                    *
B0103                    *       INTERRUPT ROUTINE LINKAGE
B0104                    *
B0105 00AE 0000  INTLNK  DATA 0,INTENO     Link, Interrupt entry point
B0106 00B2   00          BYTE 0            Name length
B0107 00B4               EVEN
B0108                    *
B0109                    ************************************************************
```

```
B0111                  ***************************************************************
B0112                  *
B0113                  *       DETERMINE WHICH DEVICE WAS ACCESSED
B0114                  *       Place a local device ID number in bits
B0115                  *       8 through 15 of R10.
B0116                  *
B0117 00B4 04CA  HEXBUS CLR   R10                HEXBUS     :     code = 0 (DC----)
B0118 00B6 1053         JMP   DSRENO
B0119 00B8 020A  RS232  LI    R10,>0001          RS232      :     code = 1 (DC 20)
      00BA 0001
B0120 00BC 1050         JMP   DSRENO
B0121 00BE 020A  RS2321 LI    R10,>0001          RS232/1    :     code = 1 (DC 20)
      00C0 0001
B0122 00C2 104D         JMP   DSRENO
B0123 00C4 020A  RS2322 LI    R10,>0002          RS232/2    :     code = 2 (DC 70)
      00C6 0002
B0124 00C8 104A         JMP   DSRENO
B0125 00CA 020A  DSK    LI    R10,>0003          DSK        :     code = 3 (DC100)
      00CC 0003
B0126 00CE 1047         JMP   DSRENO
B0127 00D0 020A  DSK1   LI    R10,>0004          DSK1       :     code = 4 (DC101)
      00D2 0004
B0128 00D4 1044         JMP   DSRENO
B0129 00D6 020A  DSK2   LI    R10,>0005          DSK2       :     code = 5 (DC102)
      00D8 0005
B0130 00DA 1041         JMP   DSRENO
B0131 00DC 020A  DSK3   LI    R10,>0006          DSK3       :     code = 6 (DC103)
      00DE 0006
B0132 00E0 103E         JMP   DSRENO
B0133 00E2 020A  DSK4   LI    R10,>0007          DSK4       :     code = 7 (DC104)
      00E4 0007
B0134 00E6 103B         JMP   DSRENO                                                  1
B0135                  *
B0136                  ***************************************************************
```

```
 0003                    COPY DLU09.KSG.HEXBUS.DILLO.POWERUP
C0001                ***************************************************************
C0002                *
C0003                *       POWER-UP ROUTINE
C0004                *
C0005                *       Registers Destroyed: R0,R1,R4,R5,R6,R7,R8
C0006                *
C0007                *       Call Subroutines   : PRECOM, BYOUT
C0008                *
C0009 00E8 C1CB   PWRENT MOV  R11,R7             Save return address
C0010 00EA 04C4          CLR  R4                 R4 = 0
C0011 00EC D804          MOVB R4,@HEXCTW         Set control byte to zero
      00EE 5FFA
C0012 00F0 02A4          STWP R4                 Set up R4 as address >E0
C0013 00F2 0224          AI   R4,->E0            Adjust R4 to address >00
      00F4 FF20
C0014 00F6 04C0          CLR  R0                 R0 = 0
C0015 00F8 0201          LI   R1,MEMSIZ          Set memory size to be assigned
      00FA 00FF
C0016 00FC 2D40          XOP  0,5                Request RAM
C0017 00FE 04C0          CLR  R0                 R0 = 0
C0018 0100 D800          MOVB R0,@LUN250         Initialize LUN0 block 250
      0102 E012
C0019 0104 D800          MOVB R0,@LUN251         Initialize LUN0 block 251
      0106 E029
C0020 0108 D800          MOVB R0,@LUN252         Initialize LUN0 block 252
      010A E040
C0021 010C D800          MOVB R0,@LUN253         Initialize LUN0 block 253
      010E E057
C0022 0110 D800          MOVB R0,@SRVREQ         Initialize service request byte
      0112 E072
C0023 0114 D820          MOVB @MAPRC0,@MAPPER    Return MAPPER state
      0116 0EE4'
      0118 8810
C0024 011A 1000          NOP                     Let Mapper finish operation   2
C0025 011C 0205          LI   R5,PWREXT          R5 = timeout branch
      011E 0150'
C0026 0120 06A0          BL   @PRECOM            Set up for communication
      0122 0DD8'
C0027 0124 D060          MOVB @HEXCTR,R1         R1 = control register
      0126 5FFC
C0028 0128 F060          SOCB @WEN,R1            Set write enable bit
      012A 001A'
C0029 012C D801          MOVB R1,@HEXCTW         Tell the control register
      012E 5FFA
C0030 0130 04C0          CLR  R0                 R0 = 0
C0031 0132 06A0          BL   @BYOUT             Send out zero device code
      0134 0DFE'
C0032 0136 D020          MOVB @RESTOP,R0         Get reset command
      0138 116E'
C0033 013A 06A0          BL   @BYOUT             Send reset command
      013C 0DFE'
C0034 013E 0206          LI   R6,7               Seven zeros to send
      0140 0007
C0035 0142 06A0          BL   @SENDZ             Send 7 zeros to bus
      0144 0A9E'
C0036 0146 D1A0  RST9    MOVB @HEXSTS,R6         R6 = hexbus status byte
      0148 5FFA
C0037 014A 0246          ANDI R6,HSKWT           Is HSKWT true? (of last byte)
      014C 8000
```

```
C0038 014E 13FB          JEQ  RST9              If not, wait until it is
C0039 0150 D060  PWREXT  MOVB @HEXCTR,R1        R1 = control register
      0152 5FFC
C0040 0154 5060          SZCB @DISABL,R1        Set chip to disable
      0156 0012'
C0041 0158 D801          MOVB R1,@HEXCTW        Tell the control byte
      015A 5FFA
C0042 015C 0457          B    *R7               Return to console routine
C0043              *
C0044              ********************************************************************
```

```
 0004                    COPY DLU09.KSG.HEXBUS.DILLO.MAINENT
D0001                ***************************************************************
D0002                *
D0003                *       MAIN ENTRY TO DSR
D0004                *       DSR proper may use: R0,R1,R2,R3,R4,R5,R6,R7,R8,R9,R10
D0005                *       R10 is used as a flag register as follows:
D0006                *            Bit 0: Circular Interrupt Req., device not open
D0007                *            Bit 1: Circular Interrupt Req., device is open
D0008                *            Bit 2: Signals RS232-type device
D0009                *            Bit 3: Verify Save File flag
D0010                *            Bit 4: PAB location flag (1 = CPU, 0 = VDP)
D0011                *            Bit 5: Flags Subroutine Entry (DATA 10)
D0012                *            Bit 6: Signals DSK device name entry
D0013                *            Bit 7: Circular Interrupt level
D0014                *            LSByt: Local DSR device number (0-7)
D0015                *
D0016 015E 02A4   DSRENO STWP R4                  Set up the CPU RAM pointer
D0017 0160 0224          AI   R4,->E0             R4 = >8300
      0162 FF20
D0018 0164 2D43          XOP  3,5                 Map in >E000 block
D0019 0166 1604          JNE  DSREO1              If no error, go on
D0020 0168 C80B          MOV  R11,@LEVEL1         Hardware error return
      016A E008
D0021 016C 0460          B    @SIZERR             Exit with error 4
      016E 0AEA'
D0022 0170 C80B   DSREO1 MOV  R11,@LEVEL1         Save the return address
      0172 E008
D0023 0174 D060          MOVB @HEXCTR,R1          R1 = control register
      0176 5FFC
D0024 0178 2060          COC  @BAVIAE,R1          Is cir. intrpt. enabled?
      017A 0014'
D0025 017C 1602          JNE  DSREN1              If not, continue
D0026 017E 026A          ORI  R10,>100            If yes, set bit 7 of R10
      0180 0100
```

```
D0028                ****************************************************************
D0029                *
D0030                *       COPY USER PAB INTO SCRATCHPAD RAM
D0031                *       Decides whether PAB is in VDP or CPU.  Sets PABLOC
D0032                *       flag to 1 if in CPU.  Shuffles information to
D0033                *       convert VDP PAB to CPU PAB format.
D0034                *
D0035                *       VDP PAB FORMAT:
D0036                *
D0037                *      ¦  I/O OPCODE....  ¦  FLAG/STATUS...  ¦
D0038                *      ¦  DATA BUFFER ADDRESS.............  ¦
D0039                *      ¦  L.R.L.........  ¦  CHR. COUNT....  ¦
D0040                *      ¦  RECORD NUMBER...................  ¦
D0041                *      ¦  SCRN. OFFSET..  ¦  NAME LENGTH...  ¦
D0042                *      ¦  file descriptor, bytes 10-?       ¦
D0043                *       ---------------------------------------
D0044                *
D0045                *       CPU PAB FORMAT:
D0046                *
D0047                *      ¦  I/O OPCODE....  ¦  FLAGS.........  ¦
D0048                *      ¦  DATA BUFFER ADDRESS.............  ¦
D0049                *      ¦  CHARACTER COUNT.................  ¦
D0050                *      ¦  RECORD NUMBER...................  ¦
D0051                *      ¦  LOGICAL RECORD LENGTH...........  ¦
D0052                *      ¦  SCRN. OFFSET..  ¦  ASSGND. LUNO*.  ¦
D0053                *      ¦  ERROR STATUS..  ¦  NAME LENGTH...  ¦
D0054                *      ¦  file descriptor, bytes 14-?       ¦
D0055                *       ---------------------------------------
D0056                *
D0057                *       *NOTE:  Assigned Luno byte is cleared on entry to the
D0058                *               DSR, and set to the LUNO given to the file on
D0059                *               exit.  This information is passed to the user
D0060                *               only when the CPU PAB format is used.
D0061                *
D0062 0182 42A0      DSREN1 SZC   @H0800,R10        Clear PAB location bit in R10
      0184 0F08'
D0063 0186 C924             MOV   @ERSTAT(R4),@PABPTR(R4) Store pointer to PAB
      0188 0056
      018A 005A
D0064 018C C924             MOV   @SCNOFF(R4),@DEVLEN(R4) Store device name length
      018E 0054
      0190 0058
D0065 0192 C264             MOV   @RAMTST(R4),R9    R9 = Device length and PAB type
      0194 0054
D0066 0196 2260             COC   @H8000,R9         Is PAB in CPU RAM?
      0198 0F06'
D0067 019A 131E             JEQ   CPUPAB            If yes, process as CPU PAB
D0068 019C 06A0             BL    @PABACS           If not, set up for VDP read
      019E 09EA'
D0069 01A0 0000             DATA  >0000
D0070 01A2 0205             LI    R5,10             VDP PAB is 10 bytes
      01A4 000A
D0071 01A6 C184             MOV   R4,R6             R6 = >8300
D0072 01A8 0226             AI    R6,OPCODE         R6 = >834A
      01AA 004A
D0073 01AC DDAF      PAVMOV MOVB  @VRD(R15),*R6+    Copy 1 byte of VDP PAB
      01AE FBFE
D0074 01B0 0605             DEC   R5                One les byte to copy
D0075 01B2 16FC             JNE   PAVMOV            Continue with next byte
D0076 01B4 C164             MOV   @LRECLN(R4),R5    Position screen offset
```

```
      01B6 0052
D0077 01B8 D905          MOVB R5,@SCNOFF(R4)
      01BA 0054
D0078 01BC 06C5          SWPB R5                    Position length of F.D.
D0079 01BE D905          MOVB R5,@NAMLEN(R4)
      01C0 0057
D0080 01C2 C164          MOV  @CHRCNT(R4),R5        Position log. rec. len.
      01C4 004E
D0081 01C6 0985          SRL  R5,8
D0082 01C8 0245          ANDI R5,>00FF
      01CA 00FF
D0083 01CC C905          MOV  R5,@LRECLN(R4)
      01CE 0052
D0084 01D0 5920          SZCB @HFF,@CHRCNT(R4) Clear upper byte CHRCNT
      01D2 0F01'
      01D4 004E
D0085 01D6 100F          JMP  CLRERR                Done with VDP PAB
D0086 01D8 E2A0  CPUPAB  SOC  @H0800,R10            Set PAB location flag in R10
      01DA 0F08'
D0087 01DC 4920          SZC  @H8000,@DEVLEN(R4) Reset PAB type bit in DEVLEN
      01DE 0F06'
      01E0 0058
D0088 01E2 06A0          BL   @CPUACS               Set up for CPU read (R9)
      01E4 0A08'
D0089 01E6 0205          LI   R5,14                 CPU PAB is 14 bytes
      01E8 000E
D0090 01EA C184          MOV  R4,R6                 R6 = >8300
D0091 01EC 0226          AI   R6,OPCODE             R6 = >834A
      01EE 004A
D0092 01F0 CDB9  PACMOV  MOV  *R9+,*R6+             Copy one word of PAB
D0093 01F2 0645          DECT R5                    One less word to copy
D0094 01F4 16FD          JNE  PACMOV                Continue copying PAB
D0095 01F6 D920  CLRERR  MOVB @H00,@ERSTAT(R4) Clear error conditions        1
      01F8 0EF0'
      01FA 0056
D0096 01FC D920          MOVB @H00,@LUNO(R4)    Set LUNO value to zero       1
      01FE 0EF0'
      0200 005F
D0097                    *
D0098                    ***************************************************************
```

```
 0005                    COPY DLU09.KSG.HEXBUS.DILLO.PARSER
E0001                ***************************************************************
E0002                *
E0003                *       BEGIN PROCESSING THE COMMAND
E0004                *
E0005 0202 9920  BEG1   CB    @H80,@OPCODE(R4) Does opcode specify cir.intrpt?
      0204 0F06'
      0206 004A
E0006 0208 1605         JNE   MPABL1           If not, continue
E0007 020A 5920         SZCB  @H80,@OPCODE(R4) Clear cir.intrpt. bit in opcode
      020C 0F06'
      020E 004A
E0008 0210 026A         ORI   R10,>8000        Set bit 0 of R10 as CI flag
      0212 8000
E0009 0214 C14A  MPABL1 MOV   R10,R5           R5 = copy of R10 flags/code
E0010 0216 0A85         SLA   R5,8             MSByte R5 = local device type
E0011 0218 1305         JEQ   BEG2             If code = 0, treat as "HEXBUS"
E0012 021A 0225         AI    R5,-3*256        Is it an RS232 code?
      021C FD00
E0013 021E 113E         JLT   RSENT            If yes, treat as "RS232"
E0014 0220 0460         B     @DSKENT          If not, treat as "DSK..."
      0222 02DA'
E0015 0224 9824  BEG2   CB    @OPCODE(R4),@H10 Is opcode for TTY break?
      0226 004A
      0228 0EF9'
E0016 022A 1306         JEQ   OPOK             If yes, continue
E0017 022C 9824         CB    @OPCODE(R4),@H09 Is opcode 0 to 9?
      022E 004A
      0230 0EFC'
E0018 0232 1202         JLE   OPOK             If yes, opcode is valid
E0019 0234 0460  OPERR  B     @BADOP           If not, bad opcode error
      0236 0AE4'
E0020 0238 06A0  OPOK   BL    @PARSE           Parse the option string
      023A 0300'
E0021 023C C145         MOV   R5,R5            Is there a special operation
E0022 023E 1301         JEQ   OPOK1            If not, continue
E0023 0240 0455         B     *R5              Execute special operation
E0024 0242 D024  OPOK1  MOVB  @DEVCOD(R4),R0   R0 = hexbus device code
      0244 005D
E0025 0246 0980         SRL   R0,8             Make it a word
E0026 0248 0280         CI    R0,20            Is device code < 20?
      024A 0014
E0027 024C 1A11         JL    OPOK3            If yes, continue
E0028 024E 0280         CI    R0,27            Is 20<= device code<=27?
      0250 001B
E0029 0252 120C         JLE   TRAP1            If yes, set flag in R10
E0030 0254 0280         CI    R0,50            Is device code < 50?
      0256 0032
E0031 0258 1A0B         JL    OPOK3            If yes, continue
E0032 025A 0280         CI    R0,57            Is 50<=device code<=57?
      025C 0039
E0033 025E 1206         JLE   TRAP1            If yes, set flag in R10
E0034 0260 0280         CI    R0,70            Is device code < 70?
      0262 0046
E0035 0264 1A05         JL    OPOK3            If yes, continue
E0036 0266 0280         CI    R0,77            Is device code > 77?
      0268 004D
E0037 026A 1B02         JH    OPOK3            If yes, continue
E0038 026C E2A0  TRAP1  SOC   @H2000,R10       Set flag bit 2 in R10
      026E 0F0A'
```

```
E0039 0270 D164  OPOK3  MOVB @OPCODE(R4),R5   R5 = opcode
      0272 004A
E0040 0274 9805         CB   R5,@H10         Is opcode for TTY break?
      0276 0EF9'
E0041 0278 1602         JNE  OPOK2           If not, continue
E0042 027A 0460         B    @TTYBRK         If yes, perform TTY break
      027C 08EE'
E0043 027E 0985  OPOK2  SRL  R5,8            Make R5 a word
E0044 0280 0A15         SLA  R5,1            R5 = R5*2 for use as offset
E0045 0282 C165         MOV  @OPTBL(R5),R5   R5 = routine address
      0284 0288'
E0046 0286 0455         B    *R5             Execute proper routine
E0047              *
E0048              ********************************************************************
E0049              *
E0050              *       OPCODE PROCESS ENTRY ADDRESSES
E0051              *
E0052 0288 04FA' OPTBL  DATA OPEN            Entry point : OPEN routine
E0053 028A 0648'        DATA CLOSE           Entry point : CLOSE routine
E0054 028C 0684'        DATA READ            Entry point : READ routine
E0055 028E 06D8'        DATA WRITE           Entry point : WRITE routine
E0056 0290 08B6'        DATA REWIND          Entry point : REWIND routine
E0057 0292 0806'        DATA LOAD            Entry point : LOAD routine
E0058 0294 0728'        DATA SAVE            Entry point : SAVE routine
E0059 0296 08F6'        DATA DELETE          Entry point : DELETE routine
E0060 0298 0AE4'        DATA BADOP           Entry point : BAD OPCODE ERROR
E0061 029A 0958'        DATA STATUS          Entry point : STATUS routine
E0062              *
E0063              ********************************************************************
```

```
E0065                ********************************************************************
E0066                *
E0067                *       RS232 DEVICES ARE HANDLED HERE
E0068                *       RS232 and RS232/1 are assigned device code 20.
E0069                *       RS232/2 is given a device code 70.
E0070                *
E0071 029C 9824      RSENT   CB    @OPCODE(R4),@H10 Is opcode for TTY break?
      029E 004A
      02A0 0EF9'
E0072 02A2 1304              JEQ   RSENT1           If yes, continue
E0073 02A4 9824              CB    @OPCODE(R4),@H03 Is opcode 0 to 3?
      02A6 004A
      02A8 0F04'
E0074 02AA 1BC4              JH    OPERR            If not, error
E0075 02AC 06CA      RSENT1  SWPB  R10              MSByte of R10 = code
E0076 02AE 980A              CB    R10,@H01         Does code = 1 (RS232 or /1)?
      02B0 0EF8'
E0077 02B2 06CA              SWPB  R10              Restore R10 to original
E0078 02B4 1604              JNE   RSENT2           If not, must be RS232/2
E0079 02B6 D920              MOVB  @D20,@DEVCOD(R4) If yes, device code = 20
      02B8 0F13'
      02BA 005D
E0080 02BC 1003              JMP   RSENT3           Continue
E0081 02BE D920      RSENT2  MOVB  @D70,@DEVCOD(R4) Device code = 70 (RS232/2)
      02C0 0F12'
      02C2 005D
E0082 02C4 C1A4      RSENT3  MOV   @PABPTR(R4),R6   R6 points to end of dev. name
      02C6 005A
E0083 02C8 D1E4              MOVB  @NAMLEN(R4),R7   R7 = # chars. left in F.D.
      02CA 0057
E0084 02CC 0987              SRL   R7,8             Make it a word
E0085 02CE 61E4              S     @DEVLEN(R4),R7   Are there more characters?
      02D0 0058
E0086 02D2 1302              JEQ   RSENT4           If not, continue
E0087 02D4 0607              DEC   R7               Skip '.' after device name
E0088 02D6 0586              INC   R6               Advance address past '.'
E0089 02D8 10B4      RSENT4  JMP   OPOK1            Continue
E0090                *
E0091                ********************************************************************
```

```
E0093                ***************************************************************
E0094                *
E0095                *       HANDLES DISK DEVICES
E0096                *
E0097                *       Device codes are assigned as follows:
E0098                *            DSK       =       100
E0099                *            DSK1      =       101
E0100                *            DSK2      =       102
E0101                *            DSK3      =       103
E0102                *            DSK4      =       104
E0103                *
E0104 02DA 026A   DSKENT ORI   R10,>0200          Set flag for DSK entry
      02DC 0200
E0105 02DE 0225          AI    R5,100*256        Set device code for DSK device
      02E0 6400
E0106 02E2 D905          MOVB  R5,@DEVCOD(R4)    Store it in ALC PAB
      02E4 005D
E0107 02E6 D1E4          MOVB  @NAMLEN(R4),R7    Set R7 to length of opt. string
      02E8 0057
E0108 02EA 0987          SRL   R7,8
E0109 02EC 61E4          S     @DEVLEN(R4),R7
      02EE 0058
E0110 02F0 0607          DEC   R7
E0111 02F2 1104          JLT   OPERR1            If no filename, error
E0112 02F4 C1A4          MOV   @PABPTR(R4),R6    R6 points to next '.'
      02F6 005A
E0113 02F8 0586          INC   R6                R6 points to option string
E0114 02FA 10A3          JMP   OPOK1             Continue processing
E0115 02FC 0460   OPERR1 B     @FILERR           NO FILENAME ERROR EXIT
      02FE 0AD8'
E0116                *
E0117                ***************************************************************
```

```
E0119                ***************************************************************
E0120                *
E0121                *       FILE DESCRIPTOR PARSE ROUTINE [HEXBUS...]
E0122                *       Format: [.<special switch>].<device code>[.LU=<luno>]
E0123                *               [.<option>]
E0124                *
E0125                *       Upon Exit:  R5 = special switch entry point
E0126                *                   R6 = address in F.D. where parse stopped
E0127                *                   R7 = number of remaining F.D. bytes
E0128                *
E0129                *       Registers Destroyed:   R0,R1,R2,R3,R5,R6,R7
E0130                *
E0131                *       Calls Subroutines:     SUREAD, RDBYT2, RDBYT3, GETSW,
E0132                *                              GETNUM
E0133                *
E0134 0300 C80B   PARSE  MOV  R11,@LEVEL2       Save return address
      0302 E00A
E0135 0304 04C5          CLR  R5                Initialize spec. switch R5 = 0
E0136 0306 D905          MOVB R5,@LUNO(R4)      Set LUNO to zero
      0308 005F
E0137 030A C064          MOV  @PABPTR(R4),R1    R1 points to end of device name
      030C 005A
E0138 030E 06A0          BL   @SUREAD           Set up for read from RAM
      0310 0A56'
E0139 0312 D024          MOVB @NAMLEN(R4),R0    R0 = length of entire F.D.
      0314 0057
E0140 0316 0980          SRL  R0,8              Make R0 a word
E0141 0318 6024          S    @DEVLEN(R4),R0    Are there more chars. in F.D.?
      031A 0058
E0142 031C 131D          JEQ  BADATT            If not, error
E0143 031E 04C2          CLR  R2                Clear R2
E0144 0320 06A0          BL   @RDBYT2           Read byte from RAM into R2
      0322 0A36'
E0145 0324 0600          DEC  R0                Any more chars. in F.D.?
E0146 0326 1318          JEQ  BADATT            If not, error
E0147 0328 0282          CI   R2,'.'*256        Is that char. a '.'?
      032A 2E00
E0148 032C 1615          JNE  BADATT            If not, error
```

```
E0150                ********************************************************************
E0151                *
E0152                *       EITHER A SPECIAL SWITCH OR DEVICE CODE WILL FOLLOW
E0153                *
E0154 032E 06A0              BL    @RDBYT2             Read byte from RAM into R2
      0330 0A36'
E0155 0332 0600              DEC   R0                  One less character to parse
E0156 0334 0282              CI    R2,'Z'*256          Is the char. before 'Z'?
      0336 5A00
E0157 0338 1B0F              JH    BADATT              If not, error
E0158 033A 0282              CI    R2,'A'*256          Is the char. after 'A'?
      033C 4100
E0159 033E 1A02              JL    PARSE0              If not, continue
E0160 0340 06A0              BL    @GETSW              Look for a special switch
      0342 03B0'
E0161 0344 06A0  PARSE0      BL    @GETNUM             Look for device code (R3)
      0346 04B6'
E0162 0348 0283              CI    R3,255              Is device code > 255?
      034A 00FF
E0163 034C 1B05              JH    BADATT              If yes, error
E0164 034E 06C3              SWPB  R3                  MSByte of R3 is device code
E0165 0350 D903              MOVB  R3,@DEVCOD(R4)      Store device code in ALC PAB
      0352 005D
E0166 0354 04C2              CLR   R2                  Clear R2
E0167 0356 1002              JMP   PARSE4              Continue
E0168 0358 0460  BADATT      B     @BADAT0             ERROR EXIT
      035A 0ADE'
```

```
E0170                    ***************************************************************
E0171                    *
E0172                    *       A LUNO (.LU=<luno>) MAY FOLLOW
E0173                    *
E0174 035C 0280  PARSE4  CI    R0,3              Are there 3 chars. left?
      035E 0003
E0175 0360 111B          JLT   PARSE1             If not, stop parsing
E0176 0362 06A0          BL    @RDBYT3           Read byte from RAM into R3
      0364 0A46'
E0177 0366 06C3          SWPB  R3                MSByte  R3 = 2nd letter of sw.
E0178 0368 06A0          BL    @RDBYT3           Read byte from RAM into R3
      036A 0A46'
E0179 036C 0283          CI    R3,'UL'           Is this the LUNO switch?
      036E 554C
E0180 0370 1613          JNE   PARSE1            If not, stop parsing
E0181 0372 06A0          BL    @RDBYT2           Read byte from RAM into R2
      0374 0A36'
E0182 0376 0282          CI    R2,'='*256        Is this char. the '='?
      0378 3D00
E0183 037A 160E          JNE   PARSE1            If not, stop parsing
E0184 037C 0220          AI    R0,-3             Three less characters to parse
      037E FFFD
E0185 0380 13EB          JEQ   BADATT            If not more chars, error
E0186 0382 06A0          BL    @RDBYT2           Read byte from RAM into R2
      0384 0A36'
E0187 0386 0600          DEC   R0                One less char. to parse
E0188 0388 06A0          BL    @GETNUM           Get the Logical Unit Number(R3)
      038A 04B6'
E0189 038C 0283          CI    R3,255            Is LUNO > 255?
      038E 00FF
E0190 0390 1BE3          JH    BADATT            If yes, error
E0191 0392 06C3          SWPB  R3                Make R3 LUNO a byte
E0192 0394 D903          MOVB  R3,@LUNO(R4)      Store LUNO in ALC PAB
      0396 005F
E0193                    *
E0194                    ***************************************************************
E0195                    *
E0196                    *       EXITING PARSER ROUTINE
E0197                    *
E0198 0398 D1A4  PARSE1  MOVB  @NAMLEN(R4),R6    MSByte R6 = total length F.D.
      039A 0057
E0199 039C 0986          SRL   R6,8              Make it a word
E0200 039E 61A4          S     @DEVLEN(R4),R6    R6 = length of option string
      03A0 0058
E0201 03A2 6180          S     R0,R6             R6 = R6 - # chars. remaining
E0202 03A4 A1A4          A     @PABPTR(R4),R6    R6 = address of remaining opts.
      03A6 005A
E0203 03A8 C1C0          MOV   R0,R7             R7 = number of chars. to parse
E0204 03AA C2E0          MOV   @LEVEL2,R11       Restore return address
      03AC E00A
E0205 03AE 045B          RT                      Return to caller
E0206                    *
E0207                    ***************************************************************
```

```
E0209                ***************************************************************
E0210                *
E0211                *       GET SPECIAL SWITCH ROUTINE
E0212                *
E0213                *       Upon Exit:  R5 = switch routine entry or zero
E0214                *
E0215                *       Registers Destroyed:  R0,R2,R3,R5,R6
E0216                *
E0217                *       Calls Subroutines:    RDBYT2, RDBYT3
E0218                *
E0219 03B0 C80B   GETSW  MOV   R11,@LEVEL3      Save return address
      03B2 E00C
E0220 03B4 C0C2          MOV   R2,R3            R3 = copy of current F.D. byte
E0221 03B6 06C3          SWPB  R3               Make room for next char. in R3
E0222 03B8 06A0          BL    @RDBYT3          Read byte from RAM into R3
      03BA 0A46'
E0223 03BC 0600          DEC   R0               One less character to scan
E0224 03BE 1131          JLT   GETSWX           If no more chars., error
E0225 03C0 0205          LI    R5,SPSWTB-2      R5 = address of switch tbl.-2
      03C2 048C'
E0226 03C4 05C5   GETSW1 INCT  R5               Skip entry address in table
E0227 03C6 C1B5          MOV   *R5+,R6          Are we at end of table (R6=0)?
E0228 03C8 132C          JEQ   GETSWX           If yes, error
E0229 03CA 80C6          C     R6,R3            Have we found the switch?
E0230 03CC 16FB          JNE   GETSW1           If not, continue table search
E0231 03CE 0283          CI    R3,'RT'          Is switch 'Transfer raw data'?
      03D0 5254
E0232 03D2 1604          JNE   GETSW2           If not, continue
E0233 03D4 C000          MOV   R0,R0            More info. following 'TR'?
E0234 03D6 1625          JNE   GETSWX           If yes, error
E0235 03D8 C155          MOV   *R5,R5           R5 = switch routine address
E0236 03DA 0455          B     *R5              Execute TR switch routine now
E0237 03DC C155   GETSW2 MOV   *R5,R5           R5 = switch routine address
E0238 03DE 0283          CI    R3,'OF'          Is switch 'Format Media'?
      03E0 4F46
E0239 03E2 1612          JNE   GETSW3           If not, continue
E0240 03E4 0280          CI    R0,10            Are there 10 chars. left?
      03E6 000A
E0241 03E8 111C          JLT   GETSWX           If not, error
E0242 03EA 0202          LI    R2,5             R2 = 5 = loop counter
      03EC 0005
E0243 03EE 0206          LI    R6,FORTBL        R6 points to format string
      03F0 04A8'
E0244 03F2 06A0   FOLOOP BL    @RDBYT3          Read byte from RAM into R3
      03F4 0A46'
E0245 03F6 06C3          SWPB  R3               Make room for next byte in R3
E0246 03F8 06A0          BL    @RDBYT3          Read byte from RAM into R3
      03FA 0A46'
E0247 03FC 8D83          C     R3,*R6+          Do these 2 bytes match?
E0248 03FE 1611          JNE   GETSWX           If not, error
E0249 0400 0602          DEC   R2               One less word to compare
E0250 0402 16F7          JNE   FOLOOP           Continue until all compared
E0251 0404 0220          AI    R0,-10           Ten less chars. to parse
      0406 FFF6
E0252 0408 0640   GETSW3 DECT  R0               Are there 2 more chars. in F.D.?
E0253 040A 110B          JLT   GETSWX           If not, error
E0254 040C 04C2          CLR   R2               Clear R2
E0255 040E 06A0          BL    @RDBYT2          Read byte from RAM into R2
      0410 0A36'
E0256 0412 0282          CI    R2,'. '*256      Is this char. a '.'?
```

```
      0414 2E00
E0257 0416 1605          JNE   GETSWX            If not, error
E0258 0418 06A0          BL    @RDBYT2           Read byte from RAM into R2
      041A 0A36'
E0259 041C C2E0          MOV   @LEVEL3,R11       Restore return address
      041E E00C
E0260 0420 045B          RT                      Return to caller
E0261 0422 0460  GETSWX  B     @BADATT           ERROR EXIT
      0424 0358'
E0262                    *
E0263                    ********************************************************************
```

```
E0265                ************************************************************
E0266                *
E0267                *       FORMAT MEDIA ROUTINE  (CC-40 type FORMAT command)
E0268                *
E0269 0426 9824  FORENT CB    @OPCODE(R4),@HO1 Is opcode a close?
      0428 004A
      042A 0EF8'
E0270 042C 131C         JEQ   FORENX           If yes, do nothing
E0271 042E 9824         CB    @OPCODE(R4),@HOO Is opcode an open?
      0430 004A
      0432 0EF0'
E0272 0434 1302         JEQ   FOREN1           If yes, continue
E0273 0436 0460  BADOPC B     @BADOP           If not, error
      0438 0AE4'
E0274 043A D920  FOREN1 MOVB  @FORMOP,@COMMAD(R4) Command = Format Media
      043C 0ED0'
      043E 005E
E0275 0440 04E4  FOREN2 CLR   @RECNO(R4)       Record Number = 0
      0442 0060
E0276 0444 04E4         CLR   @BUFLEN(R4)      Buffer Length = 0
      0446 0062
E0277 0448 C907         MOV   R7,@WDATLN(R4)   R7 = length of rem. opt. string
      044A 0064
E0278 044C C046         MOV   R6,R1            R1 = address of rem.opt. string
E0279 044E 06A0         BL    @MXMIT           Communicate over hexbus
      0450 0BCA'
E0280 0452 0460         B     @TIMOUT          TIMEOUT EXIT
      0454 0AF0'
E0281 0456 C024  FOREN3 MOV   @LRECLN(R4),R0   R0 = Logical Record Length
      0458 0052
E0282 045A 1603         JNE   FOREN4           If LRL<>0, continue
E0283 045C C920         MOV   @HOOFF,@LRECLN(R4) Set LRL to 255
      045E 0F00'
      0460 0052
E0284 0462 0460  FOREN4 B     @OPEN8           Use part of open routine
      0464 05FE'
E0285 0466 0460  FORENX B     @EXIT            Exit used with a close command
      0468 0B3E'
E0286                *
E0287                ************************************************************
```

```
E0289                ************************************************************
E0290                *
E0291                *       VERIFY SAVED FILE ROUTINE  (.VE. switch)
E0292                *
E0293                *       User is allowed to use the verify switch when opcode
E0294                *       is SAVE, OPEN, CLOSE, or WRITE.
E0295                *
E0296 046A D024      VERENT MOVB @OPCODE(R4),R0   Store opcode in MSByte R0
      046C 004A
E0297 046E 9800             CB   R0,@H06          Is opcode a SAVE?
      0470 0F02'
E0298 0472 1309             JEQ  VERFYX           If yes, go set verify flag
E0299 0474 9800             CB   R0,@H03          Is opcode a WRITE?
      0476 0F04'
E0300 0478 1306             JEQ  VERFYX           If not, go set verify flag
E0301 047A 9800             CB   R0,@H00          Is opcode an OPEN?
      047C 0EF0'
E0302 047E 1303             JEQ  VERFYX           If yes, go set verify flag
E0303 0480 9800             CB   R0,@H01          Is opcode a CLOSE?
      0482 0EF8'
E0304 0484 16D8             JNE  BADOPC           If not, invalid .VE. opcode
E0305 0486 E2A0      VERFYX SOC  @H1000,R10       Set VERIFY flag in R10
      0488 0F0C'
E0306 048A 0460             B    @OPOK1           Go back to main stream
      048C 0242'
E0307                *
E0308                ************************************************************
E0309                *
E0310                *       SPECIAL SWITCHES AND ENTRY ADDRESSES
E0311                *
E0312 048E 4F46      SPSWTB DATA 'OF',FORENT       Format Media
E0313 0492 4143             DATA 'AC',CATENT      Catalog File
E0314 0496 4556             DATA 'EV',VERENT      Verify Saved File
E0315 049A 5254             DATA 'RT',TRFENT      Transfer Raw Data
E0316 049E 4252             DATA 'BR',RESENT      Reset Hexbus
E0317 04A2 4C53             DATA 'LS',SLVENT      Slave Mode
E0318 04A6 0000             DATA 0                End of table marker
E0319 04A8   4D      FORTBL TEXT 'MRTAM DEAI'     Reversed FO'RMAT MEDIA' string
E0320 04B2   C0      ACCMOD BYTE >C0              Access mode : update
E0321 04B3   80             BYTE >80              Access mode : output
E0322 04B4   40             BYTE >40              Access mode : input
E0323 04B5   00             BYTE >00              Access mode : append
E0324                *
E0325                ************************************************************
```

```
E0327                ************************************************************
E0328                *
E0329                *       GET NUMERIC VALUE FROM F.D. ROUTINE
E0330                *       Range allowed from 0 to >FFFF.
E0331                *
E0332                *       Upon Exit:  R3 = numeric value from F.D.
E0333                *
E0334                *       Registers Destoryed:  R0,R1,R2,R3
E0335                *
E0336                *       Calls Subroutines:    RDBYT1
E0337                *
E0338 04B6 C80B      GETNUM MOV  R11,@LEVEL3      Store return address
      04B8 E00C
E0339 04BA 04C3             CLR  R3               Clear R3
E0340 04BC C042             MOV  R2,R1            R1 = copy of last char. in F.D.
E0341 04BE 0281      GETNM0 CI   R1,'0'*256       Is char. before '0'?
      04C0 3000
E0342 04C2 1A19             JL   GETNMX           If yes, error
E0343 04C4 0281             CI   R1,'9'*256       Is char. after '9'?
      04C6 3900
E0344 04C8 1B16             JH   GETNMX           If yes, error
E0345 04CA 0981             SRL  R1,8             R1 = ascii value of number
E0346 04CC 0221             AI   R1,-'0'          R1 = actual value of digit
      04CE FFD0
E0347 04D0 C083             MOV  R3,R2            R2 = accumulated value
E0348 04D2 38A0             MPY  @D10,R2          R2 & R3 = 10 * R2
      04D4 0EF0'
E0349 04D6 C082             MOV  R2,R2            Did overflow occur (R2<>0)?
E0350 04D8 160E             JNE  GETNMX           If yes, error
E0351 04DA A0C1             A    R1,R3            R3 = 10 * R3 + R1
E0352 04DC 180C             JOC  GETNMX           If carry, error
E0353 04DE C000             MOV  R0,R0            More characters in F.D.
E0354 04E0 1307             JEQ  GETNM2           If not, continue
E0355 04E2 04C1             CLR  R1               Clear R1
E0356 04E4 06A0             BL   @RDBYT1          Read byte from RAM into R1
      04E6 0A26'
E0357 04E8 0600             DEC  R0               One less char. to parse
E0358 04EA 0281             CI   R1,'.'*256       Is char a '.'?
      04EC 2E00
E0359 04EE 16E7             JNE  GETNM0           If not, continue building value
E0360 04F0 C2E0      GETNM2 MOV  @LEVEL3,R11      Restore return address
      04F2 E00C
E0361 04F4 045B             RT                    Return to caller
E0362 04F6 0460      GETNMX B    @BADATT          ERROR EXIT
      04F8 0358'
E0363                *
E0364                ************************************************************
```

```
 0006                    COPY DLU09.KSG.HEXBUS.DILLO.OPEN
F0001                ***************************************************************
F0002                *
F0003                *       OPEN COMMAND ROUTINE
F0004                *       Upon entry:  R6 = pointer to data xmit buffer
F0005                *                    R7 = # characters remaining in F.D.
F0006                *
F0007 04FA 22A0  OPEN    COC   @H8000,R10        Circular intrpt. requested?
      04FC 0F06'
F0008 04FE 130D          JEQ   OPENCT            If not, continue
F0009 0500 22A0          COC   @H2000,R10        Is it an RS232-type device?
      0502 0F0A'
F0010 0504 160A          JNE   OPENCT            If not, continue
F0011 0506 C806          MOV   R6,@OPENBF        Store R6 at OPENBF
      0508 E00E
F0012 050A C807          MOV   R7,@OPENBF+2      Store R7 at OPENBF+2
      050C E010
F0013 050E 06A0          BL    @CLOSLS           Try to close device
      0510 067E'
F0014 0512 C1A0          MOV   @OPENBF,R6        Restore R6
      0514 E00E
F0015 0516 C1E0          MOV   @OPENBF+2,R7      Restore R7
      0518 E010
F0016 051A C024  OPENCT  MOV   @LRECLN(R4),R0    R0 = Logical Record Length
      051C 0052
F0017 051E 22A0          COC   @H8000,R10        Circular intrpt. requested?
      0520 0F06'
F0018 0522 1602          JNE   OPENC             If not, continue
F0019 0524 0200          LI    R0,80             If yes, force LRL to 80 bytes
      0526 0050
F0020 0528 C900  OPENC   MOV   R0,@RECLEN(R4)    Pass LRL to ALC PAB
      052A 0068
F0021 052C 160D          JNE   OPEN01            If LRL<>0, continue
F0022 052E 04C0          CLR   R0                Clear R0
F0023 0530 D024          MOVB  @DEVCOD(R4),R0    MSByte of R0 = device code
      0532 005D
F0024 0534 1309          JEQ   OPEN01            If device=0, continue
F0025 0536 9800          CB    R0,@H08           Is device code > 8?
      0538 0F08'
F0026 053A 1B06          JH    OPEN01            If yes, continue
F0027 053C 0200          LI    R0,255            Tape unit: force LRL to 255
      053E 00FF
F0028 0540 C900          MOV   R0,@RECLEN(R4)    Pass LRL to ALC PAB
      0542 0068
F0029 0544 C900          MOV   R0,@LRECLN(R4)    Pass LRL to user's PAB        2
      0546 0052
```

```
F0031                ***************************************************************
F0032                *
F0033                *       CONVERT CONSOLE ATTRIBUTES TO ALC FORMAT
F0034                *
F0035 0548 04C1  OPEN01 CLR  R1                     Clear R1
F0036 054A D064         MOVB @FLGSTS(R4),R1         MSByte of R1 = console attrib.
      054C 004B
F0037 054E C081         MOV  R1,R2                  R2 = copy of console attrib.
F0038 0550 0241         ANDI R1,>0600               Mask out bits 5 & 6 of byte
      0552 0600
F0039 0554 22A0         COC  @H1000,R10             Is VERIFY flag set?
      0556 0F0C'
F0040 0558 1605         JNE  OPEN02                 If not, continue
F0041 055A 8801         C    R1,@H0400              If yes, are we in input mode?
      055C 0EEA'
F0042 055E 13CB         JEQ  GETNMX                 If yes, error
F0043 0560 C060         MOV  @H0400,R1      *** If not, force to input mode
      0562 0EEA'
F0044 0564 0991  OPEN02 SRL  R1,9                   Make it a value (0 - 4)
F0045 0566 D061         MOVB @ACCMOD(R1),R1         Convert to ALC access mode
      0568 04B2'
F0046 056A 20A0         COC  @H0100,R2              Relative or Sequential?
      056C 0EFE'
F0047 056E 1602         JNE  OPEN1                  If relative, continue
F0048 0570 B060         AB   @H2000,R1              Set sequential bit in R1
      0572 0F0A'
F0049 0574 20A0  OPEN1  COC  @H0800,R2              Display or Internal?
      0576 0F08'
F0050 0578 1602         JNE  OPEN2                  If internal, continue
F0051 057A B060         AB   @H0800,R1              Set display bit in R1
      057C 0F08'
F0052 057E 20A0  OPEN2  COC  @H1000,R2              Fixed or Variable?
      0580 0F0C'
F0053 0582 1302         JEQ  OPEN3                  If variable, continue
F0054 0584 B060         AB   @H1000,R1              Set fixed bit in R1
      0586 0F0C'
F0055 0588 D901  OPEN3  MOVB R1,@ALCFLG(R4)         Save attrib. in ALC PAB
      058A 005C
F0056 058C D920         MOVB @OPENOP,@COMMAD(R4) Command = open
      058E 0EC6'
      0590 005E
F0057 0592 D024         MOVB @OPCODE(R4),R0
      0594 004A
F0058 0596 1608         JNE  OPEN4
F0059 0598 06A0         BL   @CHKDSK
      059A 15A0'
F0060 059C 05A8'        DATA OPEN4                  Non-disk device
F0061 059E 0AD8'        DATA FILERR                 File already open
F0062 05A0 0AEA'        DATA SIZERR                 No block available
F0063 05A2 05A4'        DATA STFNOP                 Block available
F0064 05A4 06A0  STFNOP BL   @SAV$FN                Put file name in LUNO block
      05A6 15F6'
F0065 05A8 C924  OPEN4  MOV  @RECNUM(R4),@RECNO(R4) Pass Record Number      2
      05AA 0050
      05AC 0060
F0066 05AE 0200         LI   R0,4                   Buffer Length = 4
      05B0 0004
F0067 05B2 C900         MOV  R0,@BUFLEN(R4)         Store in ALC PAB
      05B4 0062
F0068 05B6 0227         AI   R7,3                   Data length = rem.bytes in F.D.
```

```
       05B8 0003
F0069  05BA C907          MOV   R7,@WDATLN(R4)    Store in ALC PAB
       05BC 0064
F0070  05BE C046          MOV   R6,R1             R1 = xmit data buffer address
F0071  05C0 06A0          BL    @MXMIT            Communicate over hexbus
       05C2 0BCA'
F0072  05C4 0460          B     @TIMOUT           TIMEOUT ERROR EXIT
       05C6 0AF0'
F0073  05C8 9920          CB    @H00,@OPCODE(R4)  Was opcode an open?
       05CA 0EF0'
       05CC 004A
F0074  05CE 1621          JNE   LSRTN             If not, it was load or save
```

```
F0076                ***************************************************************
F0077                *
F0078                *       HANDLE DEFAULT LOGICAL RECORD LENGTH
F0079                *
F0080 05D0 D024              MOVB @COMSTS(R4),R0    Any errors from device?
      05D2 006A
F0081 05D4 1652              JNE  RET               If yes, stop here
F0082 05D6 C920              MOV  @OPENBF+2,@RECNUM(R4) Store returned rec.num.   2
      05D8 E010
      05DA 0050
F0083 05DC C024              MOV  @RECLEN(R4),R0    Requesting default LRL?
      05DE 0068
F0084 05E0 1305              JEQ  OPEN6             If yes, continue
F0085 05E2 8800              C    R0,@OPENBF        If not, does length match?
      05E4 E00E
F0086 05E6 130B              JEQ  OPEN8             If yes, continue              1
F0087 05E8 0460  OPENX       B    @BADATO           If not, error
      05EA 0ADE'
F0088 05EC 22A0  OPEN6       COC  @H0800,R10        Is PAB in CPU memory? (R10,B4)
      05EE 0F08'
F0089 05F0 1303              JEQ  OPEN5             If yes, continue
F0090 05F2 D020              MOVB @OPENBF,R0        If not, is ret. LRL > 255?
      05F4 E00E
F0091 05F6 16F8              JNE  OPENX             If yes, error (VDP LRL>255)
F0092 05F8 C920  OPEN5       MOV  @OPENBF,@LRECLN(R4) Update LRL in console PAB
      05FA E00E
      05FC 0052
F0093                *
F0094                ***************************************************************
F0095                *
F0096                *       SHOULD SERVICE REQUEST BE ENABLED?
F0097                *
F0098 05FE 06A0  OPEN8       BL   @DSKXOP
      0600 1630'
F0099 0602 22A0              COC  @H8000,R10        Circular intrpt. requested?
      0604 0F06'
F0100 0606 1639              JNE  RET               If not, return (see close)
F0101 0608 0460              B    @ENSERV           If yes, enable service request
      060A 0618'
F0102                *
F0103                ***************************************************************
F0104                *
F0105                *       LOAD & SAVE OPEN SEQUENCE       06/01/83  KSG
F0106                *
F0107 060C C80B  OPENLS      MOV  R11,@LEVEL2       Save return address
      060E E00A
F0108 0610 10BB              JMP  OPEN3             Enter open routine
F0109 0612 C2E0  LSRTN       MOV  @LEVEL2,R11       Restore return address
      0614 E00A
F0110 0616 045B              RT                     Return to caller
```

```
F0112                ***************************************************************
F0113                *
F0114                *       ENABLE SERVICE REQUEST
F0115                *
F0116                *       Note: Service requests are only used for RS232
F0117                *       circular interrupt mode, in this version of the DSR.
F0118                *
F0119 0618 42A0  ENSERV SZC   @H8000,R10        Reset circular interrupt flag
      061A 0F06'
F0120 061C 026A         ORI   R10,>4000         Set flag to enable intrpt.
      061E 4000
F0121 0620 D920         MOVB  @ENSVOP,@COMMAD(R4) Command = enable service
      0622 0ECE'
      0624 005E
F0122 0626 04E4         CLR   @BUFLEN(R4)       Buffer Length = 0
      0628 0062
F0123 062A 04E4         CLR   @WDATLN(R4)       Data length = 0
      062C 0064
F0124 062E 06A0         BL    @MXMIT            Communicate over hexbus
      0630 0BCA'
F0125 0632 0460         B     @TIMOUT           TIMEOUT ERROR EXIT
      0634 0AF0'
F0126 0636 D024         MOVB  @COMSTS(R4),R0    Any errors in communicating?
      0638 006A
F0127 063A 1302         JEQ   ENSRV1            If not, set enable byte
F0128 063C 0460         B     @LSERR            If yes, error
      063E 07F2'
F0129 0640 D824  ENSRV1 MOVB  @DEVCOD(R4),@SRVREQ Set service request enable
      0642 005D
      0644 E072
F0130 0646 1019         JMP   RET
F0131 0648
F0132                ***************************************************************
```

```
 0007                    COPY DLU09.KSG.HEXBUS.DILLO.CLOSE
G0001                ***************************************************************
G0002                *
G0003                *       CLOSE OPCODE ROUTINE
G0004                *
G0005 0648 D920  CLOSE  MOVB @CLOSOP,@COMMAD(R4) Command = close
      064A 0EC7'
      064C 005E
G0006 064E 9824         CB   @OPCODE(R4),@CLOSOP
      0650 004A
      0652 0EC7'
G0007 0654 1602         JNE  CLOSE2
G0008 0656 06A0         BL   @COMDSK
      0658 161A'
G0009 065A 04E4  CLOSE2 CLR  @RECNO(R4)        Record Number = 0
      065C 0060
G0010 065E 04E4         CLR  @BUFLEN(R4)       Buffer length = 0
      0660 0062
G0011 0662 04E4         CLR  @WDATLN(R4)       Data length = 0
      0664 0064
G0012 0666 06A0         BL   @MXMIT            Communicate over hexbus
      0668 0BCA'
G0013 066A 0460         B    @TIMOUT           TIMEOUT ERROR EXIT
      066C 0AF0'
G0014 066E 9824  CLOSE1 CB   @OPCODE(R4),@HO1 Is this the close opcode?
      0670 004A
      0672 0EF8'
G0015 0674 16CE         JNE  LSRTN             If not, load/save return
G0016 0676 06A0         BL   @DSKXCD           Release RAM block
      0678 1636'
G0017 067A 0460  RET    B    @COMRTN           If yes, common return
      067C 0AAC'
G0018                ***************************************************************
G0019                *
G0020                *       LOAD & SAVE CLOSE ENTRY
G0021                *
G0022 067E C80B  CLOSLS MOV  R11,@LEVEL2       Save return address
      0680 E00A
G0023 0682 10E2         JMP  CLOSE             Enter close routine
G0024                *
G0025                ***************************************************************
```

```
 0008                        COPY DLU09.KSG.HEXBUS.DILLO.READWRIT
H0001                  ************************************************************
H0002                  *
H0003                  *       READ OPCODE ROUTINE
H0004                  *       User abort is checked if RS232-type device is used.
H0005                  *       This maintains compatibility with the Home Computer.
H0006                  *
H0007 0684 22A0  READ  COC   @H2000,R10        Is it an RS232- type device?
      0686 0F0A'
H0008 0688 1602        JNE   READO             If not, continue
H0009 068A 06A0        BL    @CHABRT           If yes, check for user abort
      068C 0E88'
H0010 068E D920  READO MOVB  @READOP,@COMMAD(R4) Command = read
      0690 0EC9'
      0692 005E
H0011 0694 04E4  READ1 CLR   @CHRCNT(R4)       Character count = 0
      0696 004E
H0012 0698 06A0        BL    @COMDSK           Process disk device
      069A 161A'
H0013 069C C024        MOV   @LRECLN(R4),R0    R0 = Logical Record Length
      069E 0052
H0014 06A0 C900        MOV   R0,@BUFLEN(R4)    Store LRL in ALC PAB
      06A2 0062
H0015 06A4 C924        MOV   @RECNUM(R4),@RECNO(R4) Store record # in ALC PAB
      06A6 0050
      06A8 0060
H0016 06AA 04E4        CLR   @WDATLN(R4)       Data length = 0
      06AC 0064
H0017 06AE C1E4        MOV   @BUFADR(R4),R7    R7 = address of I/O buffer
      06B0 004C
H0018 06B2 06A0        BL    @MXMIT            Communicate over hexbus
      06B4 0BCA'
H0019 06B6 0460        B     @TIMOUT           TIMEOUT ERROR EXIT
      06B8 0AF0'
H0020 06BA C924        MOV   @RDATLN(R4),@CHRCNT(R4) Pass char.cnt.to con.PAB
      06BC 0066
      06BE 004E
H0021 06C0 D024  READ3 MOVB  @COMSTS(R4),R0    Any communication errors?
      06C2 006A
H0022 06C4 1607        JNE   READ2             If yes, stop here
H0023 06C6 D024        MOVB  @FLGSTS(R4),R0    R0 = I/O attributes
      06C8 004B
H0024 06CA 2420        CZC   @H1000,R0         Is record type Fixed?
      06CC 0F0C'
H0025 06CE 1602        JNE   READ2             If not, stop here
H0026 06D0 05A4        INC   @RECNUM(R4)       If yes, update record number
      06D2 0050
H0027 06D4 0460  READ2 B     @COMRTN           Common return
      06D6 0AAC'
H0028                  *
H0029                  ************************************************************
```

```
H0031                ***************************************************************
H0032                *
H0033                *       WRITE OPCODE ROUTINE
H0034                *       User abort is checked as in READ routine.
H0035                *
H0036 06D8 22A0  WRITE  COC   @H2000,R10        Is it an RS232-type device?
      06DA 0F0A'
H0037 06DC 1602         JNE   WRITEO            If not, continue
H0038 06DE 06A0         BL    @CHABRT           If yes, check for user abort
      06E0 0E88'
H0039 06E2 06A0  WRITEO BL    @COMDSK           Process disk device
      06E4 161A'
H0040 06E6 D024         MOVB  @FLGSTS(R4),R0    Get flag bit from PAB          1
      06E8 004B
H0041 06EA 0240         ANDI  R0,>1000          Is file FIXED?                 1
      06EC 1000
H0042 06EE 1603         JNE   WRTVAR            If not, use character count    1
H0043 06F0 C024         MOV   @LRECLN(R4),R0    If yes, use LRL                1
      06F2 0052
H0044 06F4 1002         JMP   WRITR             Go set up data length          1
H0045 06F6 C024  WRTVAR MOV   @CHRCNT(R4),R0    Use character count value      1
      06F8 004E
H0046 06FA C900  WRITR  MOV   R0,@WDATLN(R4)    Set up data length for hexbus  1
      06FC 0064
H0047 06FE 04E4         CLR   @BUFLEN(R4)       Buffer length = 0
      0700 0062
H0048 0702 C924         MOV   @RECNUM(R4),@RECNO(R4) Pass record number
      0704 0050
      0706 0060
H0049 0708 D920         MOVB  @WRITOP,@COMMAD(R4) Using WRITE command
      070A 0ECA'
      070C 005E
H0050 070E 22A0         COC   @H1000,R10        Is VERIFY flag set?
      0710 0F0C'
H0051 0712 1603         JNE   WRITE1            If not, use WRITE command
H0052 0714 D920         MOVB  @VEROP,@COMMAD(R4) If yes, use VERIFY command
      0716 0ECF'
      0718 005E
H0053 071A C064  WRITE1 MOV   @BUFADR(R4),R1    R1 = address of data buffer
      071C 004C
H0054 071E 06A0         BL    @MXMIT            Communicate over hexbus
      0720 0BCA'
H0055 0722 0460         B     @TIMOUT           TIMEOUT ERROR EXIT
      0724 0AF0'
H0056 0726 10CC         JMP   READ3             Take care of record number
H0057                *
H0058                ***************************************************************
```

```
 0009                       COPY DLU09.KSG.HEXBUS.DILLO.SAVELOAD
I0001              ************************************************************
I0002              *
I0003              *       SAVE OPCODE ROUTINE
I0004              *
I0005              *       This routine first sends the "DO YOU SUPPORT SAVE?"
I0006              *       command.  If device does, a "SAVE" command is sent.
I0007              *       If device does not support save, an "OPEN", "WRITE",
I0008              *       "CLOSE" sequence is used.
I0009              *
I0010 0728 C806    SAVE    MOV   R6,@OPENBF        Store R6 temporarily
      072A E00E
I0011 072C C807            MOV   R7,@OPENBF+2      Store R7 temporarily
      072E E010
I0012 0730 D920            MOVB  @INQSAV,@COMMAD(R4) Command = support save?
      0732 0ED8'
      0734 005E
I0013 0736 04E4            CLR   @RECNO(R4)        Record number = 0
      0738 0060
I0014 073A 04E4            CLR   @WDATLN(R4)       Data length = 0
      073C 0064
I0015 073E 04E4            CLR   @BUFLEN(R4)       Buffer length = 0
      0740 0062
I0016 0742 06A0            BL    @MXMIT            Communicate
      0744 0BCA'
I0017 0746 0460            B     @TIMOUT           TIMEOUT ERROR EXIT
      0748 0AF0'
I0018 074A C1A0            MOV   @OPENBF,R6        Restore R6
      074C E00E
I0019 074E C1E0            MOV   @OPENBF+2,R7      Restore R7
      0750 E010
I0020 0752 D024            MOVB  @COMSTS(R4),R0    Was an error returned?
      0754 006A
I0021 0756 1304            JEQ   SAVED             If not, device supports SAVE
I0022 0758 9800            CB    R0,@FORMOP        Is error "command not supp."?
      075A 0ED0'
I0023 075C 1313            JEQ   SAVEN             If yes, do OPEN/WRITE/CLOSE
I0024 075E 1047            JMP   SAVEX             If not, exit with error code
I0025 0760 D920    SAVED   MOVB  @SAVEOP,@COMMAD(R4) If not, command = save
      0762 0ED7'
      0764 005E
I0026 0766 22A0            COC   @H1000,R10        Is VERIFY flag set?
      0768 0F0C'
I0027 076A 1603            JNE   SAVE01            If not, use SAVE command
I0028 076C D920            MOVB  @VERPOP,@COMMAD(R4) If yes, use VER.PGM. command
      076E 0EDA'
      0770 005E
I0029 0772 C907    SAVE01  MOV   R7,@WDATLN(R4)    Save R7 (# bytes in PAB)
      0774 0064
I0030 0776 C046            MOV   R6,R1             R1 = RAM buffer address
I0031 0778 06A0            BL    @MXMIT            Communicate
      077A 0BCA'
I0032 077C 0460            B     @TIMOUT           TIMEOUT ERROR EXIT
      077E 0AF0'
I0033 0780 0460            B     @COMRTN           Normal exit
      0782 0AAC'
I0034 0784 D060    SAVEN   MOVB  @H88,R1           Set SAVE attributes
      0786 0EF6'
I0035 0788 22A0            COC   @H1000,R10        Is VERIFY flag set?
      078A 0F0C'
```

```
I0036 078C 1602          JNE  SAVEV              If not, continue
I0037 078E D060          MOVB @H48,R1            If yes, use VERIFY attributes
      0790 0EF7'
I0038 0792 C924  SAVEV   MOV  @RECNUM(R4),@RECLEN(R4) Pass program size
      0794 0050
      0796 0068
I0039 0798 06A0          BL   @OPENLS            Open device first
      079A 060C'
I0040 079C D024          MOVB @COMSTS(R4),R0     Any errors on open?
      079E 006A
I0041 07A0 1302          JEQ  SAVE1              If not, continue
I0042 07A2 0460  SAVE0   B    @COMRTN            If yes, common return
      07A4 0AAC'
I0043 07A6 8920  SAVE1   C    @OPENBF,@RECNUM(R4) Requested size granted?
      07A8 E00E
      07AA 0050
I0044 07AC 1620          JNE  SAVEX              If not, error
I0045 07AE 22A0          COC  @H1000,R10         Is verify flag on?
      07B0 0F0C'
I0046 07B2 1604          JNE  SAVE2              If not, continue
I0047 07B4 D920          MOVB @VEROP,@COMMAD(R4) If yes, Command = verify
      07B6 0ECF'
      07B8 005E
I0048 07BA 1003          JMP  SAVE3              Continue
I0049 07BC D920  SAVE2   MOVB @WRITOP,@COMMAD(R4) Command = write
      07BE 0ECA'
      07C0 005E
I0050 07C2 04E4  SAVE3   CLR  @RECNO(R4)         Record Number = 0
      07C4 0060
I0051 07C6 04E4          CLR  @BUFLEN(R4)        Buffer length = 0
      07C8 0062
I0052 07CA C924          MOV  @RECNUM(R4),@WDATLN(R4) Data length = pgm. size
      07CC 0050
      07CE 0064
I0053 07D0 C064          MOV  @BUFADR(R4),R1     R1 = data buffer address
      07D2 004C
I0054 07D4 06A0          BL   @MXMIT             Communicate with device
      07D6 0BCA'
I0055 07D8 0460          B    @TIMOUT            TIMEOUT ERROR EXIT
      07DA 0AF0'
I0056 07DC D024          MOVB @COMSTS(R4),R0     Any errors returned by device?
      07DE 006A
I0057 07E0 1608          JNE  LSERR              If yes, error
I0058 07E2 06A0          BL   @CLOSLS            Close the device
      07E4 067E'
I0059 07E6 0460          B    @COMRTN            Common return
      07E8 0AAC'
```

```
I0061 07EA 06A0  SAVEY  BL    @CLOSLS          Close the device (see LOAD)
      07EC 067E'
I0062 07EE 0460  SAVEX  B     @SIZERR          PROGRAM SIZE ERROR EXIT
      07F0 0AEA'
I0063            *
I0064            ********************************************************************
I0065            *
I0066            *       READ/WRITE ERROR AFTER OPEN FOR LOAD/SAVE
I0067            *
I0068 07F2 D924  LSERR  MOVB @COMSTS(R4),@ALCFLG(R4) Temp. store the error
      07F4 006A
      07F6 005C
I0069 07F8 06A0         BL    @CLOSLS          Close the device
      07FA 067E'
I0070 07FC D924         MOVB @ALCFLG(R4),@COMSTS(R4) Return the actual error
      07FE 005C
      0800 006A
I0071 0802 0460         B     @COMRTN          Common return
      0804 0AAC'
I0072            *
I0073            ********************************************************************
```

```
I0075                ***************************************************************
I0076                *
I0077                *       LOAD OPCODE ROUTINE
I0078                *
I0079                *       This reoutine first sends a "LOAD" command. If device
I0080                *       does not support load, an "OPEN", "READ", "CLOSE"
I0081                *       sequence is used.
I0082                *
I0083 0806 C806  LOAD    MOV   R6,@OPENBF        Store R6 temporarily
      0808 E00E
I0084 080A C807          MOV   R7,@OPENBF+2      Store R7 temporarily
      080C E010
I0085 080E D920          MOVB  @LOADOP,@COMMAD(R4) Command = load
      0810 0ED6'
      0812 005E
I0086 0814 04E4          CLR   @RECNO(R4)        Record number = 0
      0816 0060
I0087 0818 C924          MOV   @RECNUM(R4),@BUFLEN(R4) Buffer length = pgm.len.
      081A 0050
      081C 0062
I0088 081E C907          MOV   R7,@WDATLN(R4)    Data length = remaining PAB
      0820 0064
I0089 0822 C046          MOV   R6,R1             R1 = address of PAB rem.
I0090 0824 C1E4          MOV   @BUFADR(R4),R7    R7 = destination buffer
      0826 004C
I0091 0828 06A0          BL    @MXMIT            Communicate
      082A 0BCA'
I0092 082C 0460          B     @TIMOUT           TIMEOUT ERROR EXIT
      082E 0AF0'
I0093 0830 D024          MOVB  @COMSTS(R4),R0    Was an error returned?
      0832 006A
I0094 0834 1310          JEQ   LOAD0             If not, we're done loading
I0095 0836 9800          CB    R0,@FORMOP        Was error "command not supp."?
      0838 0ED0'
I0096 083A 160D          JNE   LOAD0             If not, exit with error
I0097 083C C1A0          MOV   @OPENBF,R6        If yes, send OPEN/READ/CLOSE
      083E E00E
I0098 0840 C1E0          MOV   @OPENBF+2,R7      Restore R7
      0842 E010
I0099 0844 04E4  LOADN   CLR   @RECLEN(R4)       Requesting record length
      0846 0068
I0100 0848 D060          MOVB  @H48,R1           R1 = int., seq., var., input
      084A 0EF7'
I0101 084C 06A0          BL    @OPENLS           Open device first
      084E 060C'
I0102 0850 D024          MOVB  @COMSTS(R4),R0    Any errors after open?
      0852 006A
I0103 0854 1302          JEQ   LOAD1             If not, continue
I0104 0856 0460  LOAD0   B     @COMRTN           If yes, return with error
      0858 0AAC'
I0105 085A 8920  LOAD1   C     @OPENBF,@RECNUM(R4) Is pgm. size > buf. size?
      085C E00E
      085E 0050
I0106 0860 1BC4          JH    SAVEY             If yes, error
I0107 0862 D920          MOVB  @READOP,@COMMAD(R4) Command = read
      0864 0EC9'
      0866 005E
I0108 0868 04E4          CLR   @RECNO(R4)        Record Number = 0
      086A 0060
I0109 086C 04E4          CLR   @WDATLN(R4)       Data length = 0
```

```
                086E 0064
IO110           0870 C924          MOV  @RECNUM(R4),@BUFLEN(R4) Buffer length = pgm.size
                0872 0050
                0874 0062
IO111           0876 C924          MOV  @BUFADR(R4),@RECLEN(R4) Use RECLEN as buffer ptr
                0878 004C
                087A 0068
IO112           087C 04E4          CLR  @RECNUM(R4)       Clear recv. data accumulator
                087E 0050
IO113           0880 C1E4  LOAD2   MOV  @RECLEN(R4),R7    R7 = data buffer address
                0882 0068
IO114           0884 06A0          BL   @MXMIT            Communicate over hexbus
                0886 0BCA'
IO115           0888 0460          B    @TIMOUT           TIMEOUT ERROR EXIT
                088A 0AF0'
IO116           088C D024          MOVB @COMSTS(R4),R0    Any errors returned?
                088E 006A
IO117           0890 160A          JNE  LOAD3             If yes, go test EOF
IO118           0892 A924          A    @RDATLN(R4),@RECNUM(R4) Accumulate pgm. size
                0894 0066
                0896 0050
IO119           0898 6924          S    @RDATLN(R4),@BUFLEN(R4) Reduce local buffer size
                089A 0066
                089C 0062
IO120           089E A924          A    @RDATLN(R4),@RECLEN(R4) Advance buffer pointer
                08A0 0066
                08A2 0068
IO121           08A4 10ED          JMP  LOAD2             Loop to get all records
IO122           08A6 9824  LOAD3    CB   @COMSTS(R4),@EOFERR Have we reached EOF?
                08A8 006A
                08AA 0F14'
IO123           08AC 16A2          JNE  LSERR             If not, some other error
IO124           08AE 06A0          BL   @CLOSLS           Close the device
                08B0 067E'
IO125           08B2 0460          B    @COMRTN           Common return
                08B4 0AAC'
IO126                         *
IO127                         ***************************************************************
```

```
 0010                    COPY DLU09.KSG.HEXBUS.DILLO.RETTYDEL
J0001                ***************************************************************
J0002                *
J0003                *       REWIND OPCODE ROUTINE
J0004                *
J0005 08B6 D920  REWIND MOVB @RWDOP,@COMMAD(R4) Command = rewind
      08B8 0ECB'
      08BA 005E
J0006 08BC 06A0         BL    @COMDSK
      08BE 161A'
J0007 08C0 C924  REWIN1 MOV   @RECNUM(R4),@RECNO(R4) Pass record # to ALC PAB
      08C2 0050
      08C4 0060
J0008 08C6 04E4         CLR   @BUFLEN(R4)       Buffer Length = 0
      08C8 0062
J0009 08CA 04E4         CLR   @WDATLN(R4)       Data Length = 0
      08CC 0064
J0010 08CE 06A0         BL    @MXMIT            Communicate over hexbus
      08D0 0BCA'
J0011 08D2 0460         B     @TIMOUT           TIMEOUT ERROR EXIT
      08D4 0AF0'
J0012 08D6 D0A4         MOVB  @COMSTS(R4),R2    Was restore successful?       2
      08D8 006A
J0013 08DA 1607         JNE   REWIN2            If not, exit here             2
J0014 08DC D0A4         MOVB  @FLGSTS(R4),R2    R2 = attributes byte          2
      08DE 004B
J0015 08E0 20A0         COC   @H0100,R2         Is file Sequential?           2
      08E2 0EFE'
J0016 08E4 1302         JEQ   REWIN2            If not, exit here             2
J0017 08E6 04E4         CLR   @RECNUM(R4)       Set Record Number to zero     2
      08E8 0050
J0018 08EA 0460  REWIN2 B     @COMRTN           Common return                 2
      08EC 0AAC'
J0019                *
J0020                ***************************************************************
J0021                *
J0022                *       TTY BREAK OPCODE ROUTINE
J0023                *
J0024 08EE D920  TTYBRK MOVB @BRKOP,@COMMAD(R4) Command = TTY break
      08F0 0ED2'
      08F2 005E
J0025 08F4 10E5         JMP   REWIN1            Use rewind routine for rest
J0026                *
J0027                ***************************************************************
```

```
J0029                ***************************************************************
J0030                *
J0031                *       DELETE OPCODE COMMAND ROUTINE
J0032                *       Sends 'Delete Open File' command first.  If 'File Not
J0033                *       Open' error is returned, routine sends a 'Delete Un-
J0034                *       opened File' command.
J0035                *
J0036 08F6 D920  DELETE MOVB @DELOOP,@COMMAD(R4) Command = delete open file
      08F8 0EC8'
      08FA 005E
J0037 08FC C807         MOV  R7,@LEVEL2       Temp. store rem. bytes in F.D.
      08FE E00A
J0038 0900 C806         MOV  R6,@OPENBF       Temp. store addr. of rem. bytes
      0902 E00E
J0039 0904 06A0         BL   @CHKDSK
      0906 15A0'
J0040 0908 0910'        DATA DELIT            Non-disk device
J0041 090A 0910'        DATA DELIT            File already open
J0042 090C 0938'        DATA DELET2           File not open
J0043 090E 0938'        DATA DELET2           File not open
J0044 0910 04E4  DELIT  CLR  @RECNO(R4)       Record Number = 0
      0912 0060
J0045 0914 04E4         CLR  @BUFLEN(R4)      Buffer Length = 0
      0916 0062
J0046 0918 04E4         CLR  @WDATLN(R4)      Data Length = 0
      091A 0064
J0047 091C 06A0         BL   @MXMIT           Communicate over hexbus
      091E 0BCA'
J0048 0920 0460         B    @TIMOUT          TIMEOUT ERROR EXIT
      0922 0AF0'
J0049 0924 22A0         COC  @H0200,R10       Is this a DSK device?
      0926 0EE8'
J0050 0928 1603         JNE  DELET1           If not, continue
J0051 092A 06A0         BL   @DSKXCD          If yes, do disk processing
      092C 1636'
J0052 092E 1012         JMP  DELEXT           Exit
J0053 0930 9824  DELET1 CB   @COMSTS(R4),@NOTOPN Returned 'File not open'?
      0932 006A
      0934 0EFB'
J0054 0936 160E         JNE  DELEXT
J0055 0938 D920  DELET2 MOVB @DELNOP,@COMMAD(R4) Command = plain delete
      093A 0ECC'
      093C 005E
J0056 093E 04E4         CLR  @BUFLEN(R4)      Buffer Length = 0
      0940 0062
J0057 0942 C920         MOV  @LEVEL2,@WDATLN(R4) Data Length = F.D.bytes rem.
      0944 E00A
      0946 0064
J0058 0948 C060         MOV  @OPENBF,R1       R1 = addr. of rem. opt. string
      094A E00E
J0059 094C 06A0         BL   @MXMIT           Communicate with hexbus
      094E 0BCA'
J0060 0950 0460         B    @TIMOUT          TIMEOUT ERROR EXIT
      0952 0AF0'
J0061 0954 0460  DELEXT B    @COMRTN          Common return
      0956 0AAC'
J0062                *
J0063                ***************************************************************
```

```
 0011                    COPY DLU09.KSG.HEXBUS.DILLO.STATUS
K0001              ***************************************************************
K0002              *
K0003              *       STATUS OPCODE ROUTINE
K0004              *
K0005              *       This routine first send a 99/X Home Computer "STATUS"
K0006              *       command.  If device supports this (i.e., HFDS), the
K0007              *       status byte is sent directly to the Screen Offset
K0008              *       in the PAB without conversion (Home computer format).
K0009              *       If device does not support this, a HEXBUS "STATUS"
K0010              *       command is sent, and the byte returned is converted
K0011              *       from HEXBUS format to Home computer format.
K0012              *
K0013 0958 D920  STATUS MOVB @STATHC,@COMMAD(R4) First send 99/X status
      095A 0ED9'
      095C 005E
K0014 095E C924         MOV  @RECNUM(R4),@RECNO(R4) Pass record number       1
      0960 0050
      0962 0060
K0015 0964 0200         LI   R0,1               Buffer length = 1
      0966 0001
K0016 0968 C900         MOV  R0,@BUFLEN(R4)
      096A 0062
K0017 096C C907         MOV  R7,@WDATLN(R4)     Data length = file name length
      096E 0064
K0018 0970 C046         MOV  R6,R1              R1 points to file name in PAB
K0019 0972 C806         MOV  R6,@OPENBF         Save R6 for now
      0974 E00E
K0020 0976 06A0         BL   @MXMIT             Send Home Computer STATUS
      0978 0BCA'
K0021 097A 0460         B    @TIMOUT
      097C 0AF0'
K0022 097E C1E4         MOV  @WDATLN(R4),R7     Restore R7
      0980 0064
K0023 0982 C1A0         MOV  @OPENBF,R6         Restore R6
      0984 E00E
K0024 0986 D024         MOVB @COMSTS(R4),R0     Was an error returned?
      0988 006A
K0025 098A 132D         JEQ  STATSX             If not, we're done with STATUS
K0026 098C 9800         CB   R0,@FORMOP         Was error "command not supp."?
      098E 0ED0'
K0027 0990 162A         JNE  STATSX             If not, return with error
K0028 0992 D920  STATUN MOVB @STSOP,@COMMAD(R4) Now send normal STATUS
      0994 0ECD'
      0996 005E
K0029 0998 06A0         BL   @COMDSK            Process for disk devices
      099A 161A'
K0030 099C 04E4         CLR  @WDATLN(R4)         Data length = 0
      099E 0064
K0031 09A0 06A0         BL   @MXMIT              Send message & read resp.
      09A2 0BCA'
K0032 09A4 0460         B    @TIMOUT             TIMEOUT ERROR EXIT
      09A6 0AF0'
K0033 09A8 D024         MOVB @COMSTS(R4),R0     Test for comm. error
      09AA 006A
K0034 09AC 1302         JEQ  STATS1             Communication ok
K0035 09AE 0460         B    @COMRTN            Communication error
      09B0 0AAC'
K0036 09B2 04C2  STATS1 CLR  R2                 Set R2 = 0
K0037 09B4 D024         MOVB @SCNOFF(R4),R0     R0 = returned status byte
```

```
      09B6 0054
K0038 09B8 0A10          SLA   R0,1              ALC bit 7  = EOF
K0039 09BA 1702          JNC   STATS2             EOF set?
K0040 09BC B0A0          AB    @H01,R2            Set EOF bit 7
      09BE 0EF8'
K0041 09C0 0A10  STATS2  SLA   R0,1              ALC bit 6  = random access
K0042 09C2 1802          JOC   STATS5             Random access set?
K0043 09C4 B0A0          AB    @H04,R2            SET VARIABLE RECORD LENGTH
      09C6 0EFB'
K0044 09C8 0A10  STATS5  SLA   R0,1              ALC bit 5 = file protect
K0045 09CA 1702          JNC   STATS3             File protected?
K0046 09CC B0A0          AB    @H40,R2            Set F.P. bit 1
      09CE 0EFA'
K0047 09D0 0A10  STATS3  SLA   R0,1              Strip off all but bits 4&3
K0048 09D2 0960          SRL   R0,6               Put in position 1 & 0
K0049 09D4 D000          MOVB  R0,R0              Display bit set?
K0050 09D6 1305          JEQ   STATS4
K0051 09D8 9020          CB    @H01,R0            Internal bit set?
      09DA 0EF8'
K0052 09DC 1602          JNE   STATS4
K0053 09DE B0A0          AB    @H10,R2            Set internal bit 3
      09E0 0EF9'
K0054 09E2 D902  STATS4  MOVB  R2,@SCNOFF(R4)     Store proper status
      09E4 0054
K0055 09E6 0460  STATSX  B     @COMRTN            Status command complete
      09E8 0AAC'
K0056            *
K0057            ********************************************************************
```

```
 0012                    COPY DLU09.KSG.HEXBUS.DILLO.VDPCPU
L0001                ***************************************************************
L0002                *
L0003                *       VDP PAB ACCESS FOR READ/WRITE
L0004                *
L0005                *       Registers Destroyed:   R1
L0006                *
L0007                *       Subroutines Called:    none
L0008                *
L0009 09EA C064      PABACS MOV  @PABPTR(R4),R1    R1 = VDP PAB pointer
      09EC 005A
L0010 09EE 6064             S    @DEVLEN(R4),R1   R1 points to start of name
      09F0 0058
L0011 09F2 0221             AI   R1,-10           R1 points to start of PAB
      09F4 FFF6
L0012 09F6 A07B             A    *R11+,R1         R1 = R1 + offset
L0013 09F8 1002             JMP  VDPRD            Prepare for VDP read
L0014                *
L0015                ***************************************************************
L0016                *
L0017                *       VDP ACCESS FOR READ/WRITE DATA
L0018                *
L0019                *       Upon Entry: R1 contains VDP reference address
L0020                *
L0021                *       Registers Destroyed:   none
L0022                *
L0023                *       Subroutines Called:    none
L0024                *
L0025 09FA 0261      VDPWD  ORI  R1,>4000          Set bit for write address
      09FC 4000
L0026 09FE 06C1      VDPRD  SWPB R1                Send out LSByte of R1
L0027 0A00 D7C1             MOVB R1,*R15
L0028 0A02 06C1             SWPB R1                Send out MSByte of R1
L0029 0A04 D7C1             MOVB R1,*R15
L0030 0A06 045B             RT                     Return to caller
L0031                *
L0032                ***************************************************************
L0033                *
L0034                *       CPU PAB ACCESS FOR READ/WRITE
L0035                *       Similar to PABACS, but sets R9 to start of CPU PAB.
L0036                *
L0037                *       Upon Exit:  R9 contains CPU reference address
L0038                *
L0039                *       Registers Destroyed: R9
L0040                *
L0041 0A08 C264      CPUACS MOV  @PABPTR(R4),R9    R9 points to end of device name
      0A0A 005A
L0042 0A0C 6264             S    @DEVLEN(R4),R9   R9 points to beg. of dev. name
      0A0E 0058
L0043 0A10 0229             AI   R9,-14           R9 points to beg. of CPU PAB
      0A12 FFF2
L0044 0A14 045B             RT                     Return to caller
L0045                *
L0046                ***************************************************************
```

```
L0048                   ************************************************************
L0049                   *
L0050                   *       READ A BYTE FROM VDP OR CPU INTO R0
L0051                   *
L0052                   *       Upon Exit:  R0 contains byte read from RAM
L0053                   *
L0054                   *       Registers Destroyed:  R0,R9
L0055                   *
L0056 0A16 22A0  RDBYTO COC   @H0800,R10        Is PAB in CPU RAM?
      0A18 0F08'
L0057 0A1A 1303         JEQ   RDB0               If yes, read from CPU RAM
L0058 0A1C D02F         MOVB  @VRD(R15),R0       If not, read from VDP RAM
      0A1E FBFE
L0059 0A20 1001         JMP   RDBRTO             Continue
L0060 0A22 D039  RDB0   MOVB  *R9+,R0            Read from CPU RAM
L0061 0A24 045B  RDBRTO RT                       Return to caller
L0062                   *
L0063                   ************************************************************
L0064                   *
L0065                   *       READ A BYTE FROM VDP OR CPU INTO R1
L0066                   *
L0067                   *       Upon Exit:  R1 contains byte read from RAM
L0068                   *
L0069                   *       Registers Destroyed:  R1,R9
L0070                   *
L0071 0A26 22A0  RDBYT1 COC   @H0800,R10        Is PAB in CPU RAM?
      0A28 0F08'
L0072 0A2A 1303         JEQ   RDB1               If yes, read from CPU RAM
L0073 0A2C D06F         MOVB  @VRD(R15),R1       If not, read from VDP RAM
      0A2E FBFE
L0074 0A30 1001         JMP   RDBRT1             Continue
L0075 0A32 D079  RDB1   MOVB  *R9+,R1            Read from CPU RAM
L0076 0A34 045B  RDBRT1 RT                       Return to caller
L0077                   *
L0078                   ************************************************************
```

```
L0080                   ***************************************************************
L0081                   *
L0082                   *       READ A BYTE FROM VDP OR CPU INTO R2
L0083                   *
L0084                   *       Upon Exit:  R2 contains byte read from RAM
L0085                   *
L0086                   *       Registers Destroyed:  R2,R9
L0087                   *
L0088 0A36 22A0  RDBYT2 COC   @H0800,R10        Is PAB in CPU RAM?
      0A38 0F08'
L0089 0A3A 1303         JEQ   RDB2              If yes, read from CPU RAM
L0090 0A3C D0AF         MOVB  @VRD(R15),R2      If not, read from VDP RAM
      0A3E FBFE
L0091 0A40 1001         JMP   RDBRT2            Continue
L0092 0A42 D0B9  RDB2   MOVB  *R9+,R2           Read from CPU RAM
L0093 0A44 045B  RDBRT2 RT                      Return to caller
L0094                   *
L0095                   ***************************************************************
L0096                   *
L0097                   *       READ A BYTE FROM VDP OR CPU INTO R3
L0098                   *
L0099                   *       Upon Exit:  R3 contains byte read from RAM
L0100                   *
L0101                   *       Registers Destroyed:  R3,R9
L0102                   *
L0103 0A46 22A0  RDBYT3 COC   @H0800,R10        Is PAB in CPU RAM?
      0A48 0F08'
L0104 0A4A 1303         JEQ   RDB3              If yes, read from CPU RAM
L0105 0A4C D0EF         MOVB  @VRD(R15),R3      If not, read from VDP RAM
      0A4E FBFE
L0106 0A50 1001         JMP   RDBRT3            Continue
L0107 0A52 D0F9  RDB3   MOVB  *R9+,R3           Read from CPU RAM
L0108 0A54 045B  RDBRT3 RT                      Return to caller
L0109                   *
L0110                   ***************************************************************
```

```
L0112                ***************************************************************
L0113                *
L0114                *       SET UP FOR READ BASED ON CONTENT OF R1
L0115                *
L0116                *       Registers Destroyed:   R9 (set to start of buff.)
L0117                *
L0118                *       Subroutines Called:    VDPRD
L0119                *
L0120 0A56 C80B  SUREAD MOV   R11,@LEVEL3       Store return address
      0A58 E00C
L0121 0A5A 22A0         COC   @H0800,R10        Is PAB in CPU RAM?
      0A5C 0F08'
L0122 0A5E 1305         JEQ   SUR1              If yes, set for CPU RAM read
L0123 0A60 0241         ANDI  R1,>3FFF          If not, set for VDP RAM read
      0A62 3FFF
L0124 0A64 06A0         BL    @VDPRD            Access VDP RAM
      0A66 09FE'
L0125 0A68 1001         JMP   SURRTN            Continue
L0126 0A6A C241  SUR1   MOV   R1,R9             Set R9 for CPU RAM access
L0127 0A6C C2E0  SURRTN MOV   @LEVEL3,R11       Restore return address
      0A6E E00C
L0128 0A70 045B         RT                      Return to caller
L0129                *
L0130                ***************************************************************
L0131                *
L0132                *       SET UP FOR WRITE BASED ON CONTENT OF R1
L0133                *
L0134                *       Registers Destroyed:   R9 (set to start of buff.)
L0135                *
L0136                *       Subroutines Called:    VDPWD
L0137                *
L0138 0A72 C80B  SUWRIT MOV   R11,@LEVEL3       Store return address
      0A74 E00C
L0139 0A76 22A0         COC   @H0800,R10        Is PAB in CPU RAM?
      0A78 0F08'
L0140 0A7A 1305         JEQ   SUW1              If yes, set for CPU RAM write
L0141 0A7C 0241         ANDI  R1,>3FFF          If not, set for VDP RAM write
      0A7E 3FFF
L0142 0A80 06A0         BL    @VDPWD            Access VDP RAM
      0A82 09FA'
L0143 0A84 1001         JMP   SUWRTN            Continue
L0144 0A86 C241  SUW1   MOV   R1,R9             Set R9 for CPU RAM access
L0145 0A88 C2E0  SUWRTN MOV   @LEVEL3,R11       Restore return address
      0A8A E00C
L0146 0A8C 045B         RT                      Return to caller
L0147                *
L0148                ***************************************************************
```

```
L0150                ********************************************************************
L0151                *
L0152                *       WRITE FROM R0 TO PAB
L0153                *
L0154                *       Registers Destroyed:   R9
L0155                *
L0156 0A8E 22A0  WTBYTO COC   @H0800,R10        Is PAB in CPU RAM?
      0A90 0F08'
L0157 0A92 1303         JEQ   WB0               If yes, write to CPU RAM
L0158 0A94 DBC0         MOVB  R0,@VWD(R15)      If not, write to VDP RAM
      0A96 FFFE
L0159 0A98 1001         JMP   WBORTN            Continue
L0160 0A9A DE40  WB0    MOVB  R0,*R9+           Write byte to CPU RAM
L0161 0A9C 045B  WBORTN RT                      Return to caller
L0162                *
L0163                ********************************************************************
L0164                *
L0165                *       SEND BYTES OF ZERO TO THE HEXBUS
L0166                *
L0167                *       Upon entry:    R6 contains the number of zeros to send
L0168                *
L0169                *       Registers Destroyed:   R0,R6,R5
L0170                *
L0171 0A9E C14B  SENDZ  MOV   R11,R5            Save return address
L0172 0AA0 04C0         CLR   R0                R0 = 0
L0173 0AA2 06A0  SENDZ1 BL    @BYOUT            Send a byte of zeros
      0AA4 0DFE'
L0174 0AA6 0606         DEC   R6                One less byte to send
L0175 0AA8 16FC         JNE   SENDZ1            Continue sending bytes out
L0176 0AAA 0455         B     *R5               Return to caller
L0177                *
L0178                ********************************************************************
```

```
 0013                    COPY DLU09.KSG.HEXBUS.DILLO.EXITS
M0001                ***************************************************************
M0002                *
M0003                *       COMMON RETURN AREA
M0004                *
M0005                *       This section handles error code conversions and
M0006                *       makes arrangements to exit the DSR.
M0007                *
M0008  0AAC D0A4     COMRTN MOVB @COMSTS(R4),R2    R2 = returned error
       0AAE 006A
M0009  0AB0 04C1            CLR  R1               R1 = 0
M0010  0AB2 0982            SRL  R2,8             Make it a word
M0011  0AB4 0282            CI   R2,HERLEN        Does it exceed table length?
       0AB6 001C
M0012  0AB8 1403            JHE  COMRT1           If yes, process separately
M0013  0ABA D062            MOVB @HOMERR(R2),R1   R1 = home computer error code
       0ABC 0EAA'
M0014  0ABE 101D            JMP  BADEXT           Go to pass error to console
M0015  0AC0 0282     COMRT1 CI   R2,MEDFUL        'Media Full' error?
       0AC2 0020
M0016  0AC4 1312            JEQ  SIZERR           If yes, generate error 4
M0017  0AC6 0282            CI   R2,SLVILL        'Illegal in Slave Mode' error?
       0AC8 00FE
M0018  0ACA 1315            JEQ  DEVERR           If yes, generate error 6
M0019  0ACC 0282            CI   R2,>60           Is error less that >60?        1
       0ACE 0060
M0020  0AD0 1A03            JL   FILERR           If yes, generate file error    1
M0021  0AD2 0282            CI   R2,>6F            Is error between >60 and >6F? 1
       0AD4 006F
M0022  0AD6 120F            JLE  DEVERR           If yes, generate device error 1
M0023  0AD8 0201     FILERR LI   R1,>E000         Generate error 7:file error
       0ADA E000
M0024  0ADC 100E            JMP  BADEXT           Go to pass to console
M0025  0ADE 0201     BADATO LI   R1,>4000         Generate error 2:bad open att.
       0AE0 4000
M0026  0AE2 100B            JMP  BADEXT           Go to pass to console
M0027  0AE4 0201     BADOP  LI   R1,>6000         Generate error 3:illegal op.
       0AE6 6000
M0028  0AE8 1008            JMP  BADEXT           Go to pass to console
M0029  0AEA 0201     SIZERR LI   R1,>8000         Generate error 4:out of buffer
       0AEC 8000
M0030  0AEE 1005            JMP  BADEXT           Go to pass to console
M0031  0AF0 D920     TIMOUT MOVB @HFF,@COMSTS(R4) Set timeout error code >FF
       0AF2 0F01'
       0AF4 006A
M0032  0AF6 0201     DEVERR LI   R1,>C000         Generate error 6:device error
       0AF8 C000
M0033  0AFA 0911     BADEXT SRL  R1,1             Change upper 3 bits to MSN
M0034  0AFC D901            MOVB R1,@ERSTAT(R4)   Set the error code             1
       0AFE 0056
M0035  0B00 131E            JEQ  EXIT             If no error, continue          1
M0036  0B02 D1E4            MOVB @NAMLEN(R4),R7   Calculate length of file name 1
       0B04 0057
M0037  0B06 0987            SRL  R7,8                                            1
M0038  0B08 61E4            S    @DEVLEN(R4),R7                                  1
       0B0A 0058
M0039  0B0C 0607            DEC  R7                                              1
M0040  0B0E C1A4            MOV  @PABPTR(R4),R6   Calculate file name pointer    1
       0B10 005A
M0041  0B12 0586            INC  R6                                              1
```

```
M0042 0B14 06A0          BL   @CHKDSK           If error, check LUNO table    1
      0B16 15A0'
M0043 0B18 0B3E'         DATA EXIT              1: Exit for non-DSK device    1
M0044 0B1A 0B20'         DATA FORCE             2: Force close if file found  1
M0045 0B1C 0B3E'         DATA EXIT              3: Exit if file not found     1
M0046 0B1E 0B3E'         DATA EXIT              4: Exit if file not found     1
M0047 0B20 06A0  FORCE   BL   @DSKXCD           Wipe out the filename entry   1
      0B22 1636'
M0048 0B24 D920          MOVB @CLOSOP,@COMMAD(R4) Attempt to close the file   1
      0B26 0EC7'
      0B28 005E
M0049 0B2A 04E4          CLR  @RECNO(R4)        Set record number to 0        1
      0B2C 0060
M0050 0B2E 04E4          CLR  @BUFLEN(R4)       Set buffer length to 0        1
      0B30 0062
M0051 0B32 04E4          CLR  @WDATLN(R4)       Set data length to 0          1
      0B34 0064
M0052 0B36 06A0          BL   @MXMIT            Send close command to bus     1
      0B38 0BCA'
M0053 0B3A 0460          B    @TIMOUT           Timeout error exit            1
      0B3C 0AF0'
M0054 0B3E 06A0  EXIT    BL   @RSTINT           Make sure intrpt. properly set
      0B40 0D7A'
M0055 0B42 D060          MOVB @HEXCTR,R1        R1 = control register
      0B44 5FFC
M0056 0B46 5060          SZCB @DISABL,R1        Set chip to disable
      0B48 0012'
M0057 0B4A D801          MOVB R1,@HEXCTW        Tell the control byte
      0B4C 5FFA
```

```
M0059                ***************************************************************
M0060                *
M0061                *       COPY SCRATCH PAB BACK TO USER'S PAB IN CPU OR VDP
M0062                *
M0063                *       See entry to DSR for PAB format conversions.
M0064                *
M0065 0B4E 22A0  EXIT1  COC   @H0800,R10        Is PAB in CPU RAM?
      0B50 0F08'
M0066 0B52 1325         JEQ   EXPABC            If yes, handle PAB in CPU
M0067 0B54 D1E4         MOVB  @FLGSTS(R4),R7    VDP: Mask off error flags
      0B56 004B
M0068 0B58 0247         ANDI  R7,>1FFF
      0B5A 1FFF
M0069 0B5C D907         MOVB  R7,@FLGSTS(R4)
      0B5E 004B
M0070 0B60 D1E4         MOVB  @ERSTAT(R4),R7    VDP: Convert CPU to VDP errors
      0B62 0056
M0071 0B64 0A17         SLA   R7,1
M0072 0B66 0247         ANDI  R7,>E000
      0B68 E000
M0073 0B6A F907         SOCB  R7,@FLGSTS(R4)    Combine errors into FS
      0B6C 004B
M0074 0B6E C1E4         MOV   @LRECLN(R4),R7    Convert char. count
      0B70 0052
M0075 0B72 06C7         SWPB  R7
M0076 0B74 D907         MOVB  R7,@CHRCNT(R4)
      0B76 004E
M0077 0B78 D924         MOVB  @SCNOFF(R4),@LRECLN(R4) Convert LRL
      0B7A 0054
      0B7C 0052
M0078 0B7E D924         MOVB  @NAMLEN(R4),@LRECLN+1(R4)
      0B80 0057
      0B82 0053
M0079 0B84 06A0         BL    @PABACS           Set up for VDP write
      0B86 09EA'
M0080 0B88 4000         DATA  >4000
M0081 0B8A 0205         LI    R5,10             VDP PAB is 10 bytes
      0B8C 000A
M0082 0B8E C184         MOV   R4,R6
M0083 0B90 0226         AI    R6,OPCODE         R6 = >834A
      0B92 004A
M0084 0B94 DBF6  COPYV  MOVB  *R6+,@VWD(R15)    Write a byte to VDP
      0B96 FFFE
M0085 0B98 0605         DEC   R5                One less byte to write
M0086 0B9A 16FC         JNE   COPYV             Continue copying PAB
M0087 0B9C 100D         JMP   ALLDON
M0088 0B9E D924  EXPABC MOVB  @LUNO(R4),@PABLUN(R4) Give assigned LUNO to user
      0BA0 005F
      0BA2 0055
M0089 0BA4 06A0         BL    @CPUACS           Set up for CPU write
      0BA6 0A08'
M0090 0BA8 0205         LI    R5,14             CPU PAB has 14 bytes
      0BAA 000E
M0091 0BAC C184         MOV   R4,R6
M0092 0BAE 0226         AI    R6,OPCODE         R6 = >834A
      0BB0 004A
M0093 0BB2 CE76  COPYC  MOV   *R6+,*R9+         Write a word to CPU
M0094 0BB4 0645         DECT  R5                One less word to write
M0095 0BB6 16FD         JNE   COPYC             Continue copying PAB
M0096 0BB8 05E0  ALLDON INCT  @LEVEL1           Accept return address
```

```
       0BBA E008
M0097  0BBC C2E0          MOV  @LEVEL1,R11       Restore the return address
       0BBE E008
M0098  0BC0 D820          MOVB @MAPRC0,@MAPPER   Restore mapper
       0BC2 0EE4'
       0BC4 8810
M0099  0BC6 1000          NOP                    Let Mapper finish operation   2
M0100  0BC8 045B          RT                     Return to console
M0101               *
M0102               ************************************************************
```

```
 0014                    COPY DLU09.KSG.HEXBUS.DILLO.XMITRECV
 N0001                ***************************************************************
 N0002                *
 N0003                *       MASTER MODE COMMUNICATION ROUTINE
 N0004                *       Writes message to hexbus and reads response
 N0005                *       Upon entry: R11    = Timeout error return
 N0006                *                   R11+4  = Normal return
 N0007                *                   R1     = Address of data to follow ALC PAB
 N0008                *                   R7     = Address to store returned data
 N0009                *
 N0010                *       Registers destroyed: 0,1,2,3,5,6,7,8,11
 N0011                *
 N0012                *       HEXBUS PAB SENT OVER THE BUS:
 N0013                *       ____________________________
 N0014                *      |  DEVICE CODE               |  Device Code
 N0015                *      |  COMMAND CODE              |  Command Code
 N0016                *      |  LOGICAL UNIT NUMBER (LUNO)|  Luno
 N0017                *      |  LSBYTE OF RECORD NUMBER   |  Record Number
 N0018                *      |  MSBYTE OF RECORD NUMBER   |
 N0019                *      |  LSBYTE OF BUFFER LENGTH   |  Buffer Length
 N0020                *      |  MSBYTE OF BUFFER LENGTH   |
 N0021                *      |  LSBYTE OF DATA LENGTH     |  Data Length
 N0022                *      |  MSBYTE OF DATA LENGTH     |
 N0023                *      |  data bytes, bytes 10-?    |  Data (if any)
 N0024                *       ----------------------------
 N0025                *
 N0026                *       HEXBUS DEVICE RESPONSE FORMAT:
 N0027                *       ____________________________
 N0028                *      |  LSBYTE OF DATA LENGTH     |  Data length
 N0029                *      |  MSBYTE OF DATA LENGTH     |
 N0030                *      |  data bytes, if any        |  Data (if any)
 N0031                *      |  OPERATION STATUS BYTE     |  Status byte
 N0032                *       ----------------------------
 N0033                *
 N0034 0BCA C14B  MXMIT  MOV  R11,R5          R5 = timeout return address
 N0035 0BCC 06A0         BL   @SUREAD         Set up for read from RAM
       0BCE 0A56'
 N0036 0BD0 06A0         BL   @PRECOM         Prepare to communicate
       0BD2 0DD8'
 N0037 0BD4 D060         MOVB @HEXCTR,R1      R1 = control register
       0BD6 5FFC
 N0038 0BD8 F060         SOCB @REN,R1         Set read enable bit
       0BDA 001C'
 N0039 0BDC F060         SOCB @WEN,R1         Set write enable bit
       0BDE 001A'
 N0040 0BE0 D801         MOVB R1,@HEXCTW      Tell the control byte
       0BE2 5FFA
 N0041 0BE4 D024         MOVB @DEVCOD(R4),R0  MSByte R0 = device code
       0BE6 005D
 N0042 0BE8 06A0         BL   @BYOUT          Send device code over hexbus
       0BEA 0DFE'
 N0043 0BEC D024         MOVB @COMMAD(R4),R0  MSByte R0 = command code
       0BEE 005E
 N0044 0BF0 06A0         BL   @BYOUT          Send command code over hexbus
       0BF2 0DFE'
 N0045 0BF4 D024         MOVB @LUNO(R4),R0    MSByte R0 = LUNO
       0BF6 005F
 N0046 0BF8 06A0         BL   @BYOUT          Send LUNO over hexbus
       0BFA 0DFE'
 N0047 0BFC D024         MOVB @RECNO+1(R4),R0 MSByte R0 = LSByte of record #
```

```
       0BFE 0061
N0048  0C00 06A0          BL   @BYOUT              Send LSB record # over hexbus
       0C02 0DFE'
N0049  0C04 D024          MOVB @RECNO(R4),R0       MSByte R0 = MSByte of record #
       0C06 0060
N0050  0C08 06A0          BL   @BYOUT              Send MSB record # over hexbus
       0C0A 0DFE'
N0051  0C0C D024          MOVB @BUFLEN+1(R4),R0    MSByte R0 = LSByte of buf. len.
       0C0E 0063
N0052  0C10 06A0          BL   @BYOUT              Send LSB of buffer length
       0C12 0DFE'
N0053  0C14 D024          MOVB @BUFLEN(R4),R0      MSByte R0 = MSByte of buf. len.
       0C16 0062
N0054  0C18 06A0          BL   @BYOUT              Send MSB of buffer length
       0C1A 0DFE'
N0055  0C1C C024          MOV  @WDATLN(R4),R0      R0 = data length (to be sent)
       0C1E 0064
N0056  0C20 9920          CB   @SAVEOP,@COMMAD(R4) Is this a SAVE operation?
       0C22 0ED7'
       0C24 005E
N0057  0C26 1304          JEQ  MXMITQ              If yes, adjust data length
N0058  0C28 9920          CB   @VERPOP,@COMMAD(R4) Is this a VERIFY operation?
       0C2A 0EDA'
       0C2C 005E
N0059  0C2E 1603          JNE  MXMIT1              If not, continue
N0060  0C30 A024  MXMITQ  A    @RECNUM(R4),R0      If yes, adjust data length
       0C32 0050
N0061  0C34 0580          INC  R0                  Add to include zero byte
N0062  0C36 22A0  MXMIT1  COC  @H8000,R10          Circular intrpt. requested?
       0C38 0F06'
N0063  0C3A 1602          JNE  MXMIT6              If not, continue
N0064  0C3C 0220          AI   R0,11               If yes, include '.T=C.R=N.EC'
       0C3E 000B
N0065  0C40 06C0  MXMIT6  SWPB R0                  Send LSByte of data length
N0066  0C42 06A0          BL   @BYOUT
       0C44 0DFE'
N0067  0C46 06C0          SWPB R0                  Send MSByte of data length
N0068  0C48 06A0          BL   @BYOUT
       0C4A 0DFE'
N0069  0C4C 22A0          COC  @H0400,R10          Are we in a subroutine?
       0C4E 0EEA'
N0070  0C50 1358          JEQ  SBR$EX              If yes, exit to it
```

```
N0072                ***************************************************************
N0073                *
N0074                *       MXMIT: This section transmits the data field, if any.
N0075                *              Address of buffer was previously set.
N0076                *              Note special case for open.
N0077                *
N0078 0C52 C1A4  XMIT3  MOV  @WDATLN(R4),R6   R6 = data length (to be sent)
      0C54 0064
N0079 0C56 133A         JEQ  MRECV            If data length = 0, skip
N0080 0C58 D024         MOVB @COMMAD(R4),R0   Is this the open command?
      0C5A 005E
N0081 0C5C 160F         JNE  MXMIT2           If not, skip
N0082                *
N0083                ***************************************************************
N0084                *
N0085                *       In open command, send LRL, Attributes byte, and any
N0086                *       other switches left in the F.D.
N0087                *
N0088 0C5E D024         MOVB @RECLEN+1(R4),R0 MSByte R0 = LSByte of LRL
      0C60 0069
N0089 0C62 06A0         BL   @BYOUT           Send LSB of LRL over hexbus
      0C64 0DFE'
N0090 0C66 D024         MOVB @RECLEN(R4),R0   MSByte R0 = MSByte of LRL
      0C68 0068
N0091 0C6A 06A0         BL   @BYOUT           Send MSB of LRL over hexbus
      0C6C 0DFE'
N0092 0C6E D024         MOVB @ALCFLG(R4),R0   MSByte R0 = attributes byte
      0C70 005C
N0093 0C72 06A0         BL   @BYOUT           Send attributes over hexbus
      0C74 0DFE'
N0094 0C76 0226         AI   R6,-3            More than this to send?
      0C78 FFFD
N0095 0C7A 1306         JEQ  MXMIT7           If not, skip
N0096 0C7C 06A0  MXMIT2 BL   @RDBYTO          Read byte from RAM into R0
      0C7E 0A16'
N0097 0C80 06A0         BL   @BYOUT           Send a byte of data over hexbus
      0C82 0DFE'
N0098 0C84 0606         DEC  R6               One less byte to send out
N0099 0C86 16FA         JNE  MXMIT2           Loop until all data sent
N0100 0C88 9920  MXMIT7 CB   @SAVEOP,@COMMAD(R4) Is this a SAVE command?
      0C8A 0ED7'
      0C8C 005E
N0101 0C8E 1304         JEQ  MXMITR           If yes, process it
N0102 0C90 9920         CB   @VERPOP,@COMMAD(R4) Is this a VERIFY command?
      0C92 0EDA'
      0C94 005E
N0103 0C96 160F         JNE  MXMIT4           If not, continue
N0104 0C98 04C0  MXMITR CLR  R0               Send a zero byte after the f.n.
N0105 0C9A 06A0         BL   @BYOUT
      0C9C 0DFE'
N0106 0C9E C064         MOV  @BUFADR(R4),R1   Send the program itself out
      0CA0 004C
N0107 0CA2 06A0         BL   @SUREAD
      0CA4 0A56'
N0108 0CA6 C1A4         MOV  @RECNUM(R4),R6
      0CA8 0050
N0109 0CAA 06A0  MXMIT3 BL   @RDBYTO
      0CAC 0A16'
N0110 0CAE 06A0         BL   @BYOUT
      0CB0 0DFE'
```

```
NO111 OCB2 0606          DEC   R6
NO112 OCB4 16FA          JNE   MXMIT3
NO113 OCB6 22A0 MXMIT4   COC   @H8000,R10       Circular intrpt. requested?
      OCB8 0F06'
NO114 OCBA 1608          JNE   MRECV            If not, skip
NO115 OCBC 0206          LI    R6,TRANS         R6 points to '.T=C...' string
      OCBE 0E9E'
NO116 OCC0 D036 MXMIT8   MOVB  *R6+,R0          MSByte of R0 = byte to be sent
NO117 OCC2 06A0          BL    @BYOUT           Send byte of string over hexbus
      OCC4 0DFE'
NO118 OCC6 0286          CI    R6,TRANEN        Have we reachedend of string?
      OCC8 0EA9'
NO119 OCCA 16FA          JNE   MXMIT8           Continue transmitting string
```

```
NO121                ***************************************************************
NO122                *
NO123                *       Master now expects the slave device to respond to
NO124                *       the message just sent.  Data is routed to buffer
NO125                *       pointed to by R7.  Note special case for open, in
NO126                *       which data is sent to special 4 byte buffer
NO127                *
NO128 OCCC C047   MRECV  MOV  R7,R1             R1 = data receive buffer addr.
NO129 OCCE 06A0          BL   @SUWRIT           Set up for write to RAM
      OCD0 0A72'
NO130 OCD2 06A0          BL   @BYTIN            Get LSByte of data length
      OCD4 0E18'
NO131 OCD6 D900          MOVB RO,@RDATLN+1(R4)  Store in scratchpad
      OCD8 0067
NO132 OCDA 06A0          BL   @BYTIN            Get MSByte of data length
      OCDC 0E18'
NO133 OCDE D900          MOVB RO,@RDATLN(R4)    Store in scratchpad
      OCE0 0066
NO134 OCE2 C1A4          MOV  @RDATLN(R4),R6    R6 = data length to receive
      OCE4 0066
NO135 OCE6 1328          JEQ  MRECV3            If no data, skip to read stat. V
NO136 OCE8 8906          C    R6,@BUFLEN(R4)    Is data length <= buf. len.
      OCEA 0062
NO137 OCEC 1214          JLE  MRECV2            If yes, continue
NO138 OCEE 0586  MRECO5  INC  R6                Set R6 to include status
NO139 OCF0 06A0  MRECV5  BL   @BYTIN            Read and toss a byte
      OCF2 0E18'
NO140 OCF4 0606          DEC  R6                One less byte to toss
NO141 OCF6 16FC          JNE  MRECV5            Continue tossing bytes
NO142 OCF8 D020          MOVB @HOC,RO           Set error condition (disk)
      OCFA 0F10'
NO143 OCFC 22A0          COC  @HO400,R10        Is this a disk operation
      OCFE 0EEA'
NO144 OD00 1604          JNE  MRECV7            If not, continue
NO145 OD02 C2C5  SBR$EX  MOV  R5,R11            If yes, return to disk routines
NO146 OD04 022B          AI   R11,4             Get proper rerturn address
      OD06 0004
NO147 OD08 045B          RT                     Return to disk routine
NO148 OD0A D900  MRECV7  MOVB RO,@COMSTS(R4)    Update hexbus status
      OD0C 006A
NO149 OD0E 06A0          BL   @RSTINT           Check interrupt level
      OD10 0D7A'
NO150 OD12 0460          B    @DEVERR           Exit with error
      OD14 0AF6'
NO151 OD16 D024  MRECV2  MOVB @COMMAD(R4),RO    Is this an open command?
      OD18 005E
NO152 OD1A 133F          JEQ  MRECOP            If yes, go to special process
NO153 OD1C 9020          CB   @STSOP,RO         If not, is it a status command?
      OD1E 0ECD'
NO154 OD20 1351          JEQ  MRECST            If yes, go to special process
NO155 OD22 9020          CB   @STATHC,RO        If not, is it H.C. status?
      OD24 0ED9'
NO156 OD26 134E          JEQ  MRECST            If yes, treat as status command
NO157 OD28 C186  MRECV8  MOV  R6,R6             Are there bytes to fill buffer?
NO158 OD2A 1306          JEQ  MRECV3            If not, skip
NO159 OD2C 06A0  MRECV9  BL   @BYTIN            Receive a byte from the bus
      OD2E 0E18'
NO160 OD30 06A0          BL   @WTBYTO           Write byte to RAM from RO
      OD32 0A8E'
NO161 OD34 0606          DEC  R6                One less byte to write
```

```
N0162 0D36 16FA          JNE  MRECV9          Continue reading from the bus
```

```
N0164                    ***************************************************************
N0165                    *
N0166                    *       LAST STEP FOR NORMAL EXIT        05/27/83  KSG
N0167                    *
N0168 0D38 06A0  MRECV3  BL    @BYTIN              Read status byte from hexbus
      0D3A 0E18'
N0169 0D3C 22A0          COC   @H0400,R10          Is this a disk manager ftn.?
      0D3E 0EEA'
N0170 0D40 13E0          JEQ   SBR$EX              If yes, exit routine now
N0171 0D42 D900          MOVB  R0,@COMSTS(R4)      Store it
      0D44 006A
N0172 0D46 22A0  MRECV6  COC   @H4000,R10          Circular intrpt. enabled?
      0D48 0F0E'
N0173 0D4A 160C          JNE   MREC06              If not, continue
N0174 0D4C D024          MOVB  @COMSTS(R4),R0      Was communication successful?
      0D4E 006A
N0175 0D50 1609          JNE   MREC06              If not, continue
N0176 0D52 D060          MOVB  @HEXCTR,R1          R1 = control byte
      0D54 5FFC
N0177 0D56 F060          SOCB  @BAVIAE,R1          Enable BAV falling intrpt.
      0D58 0014'
N0178 0D5A D801          MOVB  R1,@HEXCTW          Tell the control byte
      0D5C 5FFA
N0179 0D5E 026A          ORI   R10,>100            Set flag bit 7 in R10
      0D60 0100
N0180 0D62 1002          JMP   MREC07              Continue
N0181 0D64 06A0  MREC06  BL    @RSTINT             Make sure intrpt. properly set
      0D66 0D7A'
N0182 0D68 D060  MREC07  MOVB  @HEXCTR,R1          R1 = control byte
      0D6A 5FFC
N0183 0D6C 5060          SZCB  @DISABL,R1          Set chip to disable
      0D6E 0012'
N0184 0D70 D801          MOVB  R1,@HEXCTW          Tell the control byte
      0D72 5FFA
N0185 0D74 0225          AI    R5,4                Create normal return address
      0D76 0004
N0186 0D78 0455          B     *R5                 Return to caller
N0187                    *
N0188                    ***************************************************************
N0189                    *
N0190                    *       RESTORE CIR. INTRPT. ENABLE TO PROPER LEVEL
N0191                    *
N0192 0D7A C04A  RSTINT  MOV   R10,R1              R1 = copy of R10 flags
N0193 0D7C D0A0          MOVB  @HEXCTR,R2          R2 = control byte
      0D7E 5FFC
N0194 0D80 0241          ANDI  R1,>0100            Is bit 7 (cir. intrpt.) set?
      0D82 0100
N0195 0D84 1305          JEQ   RSTIN1              If no, disable cir. intrpt.
N0196 0D86 F0A0          SOCB  @BAVIAE,R2          Enable BAV falling intrpt.
      0D88 0014'
N0197 0D8A D802          MOVB  R2,@HEXCTW          Tell the control register
      0D8C 5FFA
N0198 0D8E 1004          JMP   RSTIN2
N0199 0D90 50A0  RSTIN1  SZCB  @BAVIAE,R2          Disable BAV falling intrpt.
      0D92 0014'
N0200 0D94 D802          MOVB  R2,@HEXCTW          Tell the control byte
      0D96 5FFA
N0201 0D98 045B  RSTIN2  RT                        Return to caller
N0202                    *
N0203                    ***************************************************************
```

```
N0205                    ***************************************************************
N0206                    *
N0207                    *       SPECIAL PROCESS TO RECEIVE OPEN RESPONSE
N0208                    *
N0209                    *       Respose data length must be 4 bytes.
N0210                    *
N0211 0D9A 0208  MRECOP  LI    R8,OPENBF        R8 points to special 4byte buf.
      0D9C E00E
N0212 0D9E 0286          CI    R6,4             Are there 4 bytes coming back?
      0DA0 0004
N0213 0DA2 16A5          JNE   MREC05           If not, error
N0214 0DA4 06A0  MRCOP1  BL    @BYTIN           Read a byte from the bus
      0DA6 0E18'
N0215 0DA8 DE00          MOVB  R0,*R8+          Store byte in spcial buffer
N0216 0DAA 0606          DEC   R6               One less byte to read
N0217 0DAC 16FB          JNE   MRCOP1           Continue until all are read
N0218 0DAE C020          MOV   @OPENBF,R0       R0 = LRL returned
      0DB0 E00E
N0219 0DB2 06C0          SWPB  R0               Put bytes in proper order
N0220 0DB4 C800          MOV   R0,@OPENBF       Store real LRL returned in buf.
      0DB6 E00E
N0221 0DB8 C020          MOV   @OPENBF+2,R0     R0 = Record Number returned   2
      0DBA E010
N0222 0DBC 06C0          SWPB  R0               Put bytes in proper order     2
N0223 0DBE C800          MOV   R0,@OPENBF+2     Store read Rec. Num. in buf.  2
      0DC0 E010
N0224 0DC2 10BA          JMP   MRECV3           Continue
N0225                    *
N0226                    ***************************************************************
N0227                    *
N0228                    *       SPECIAL PROCESS TO RECEIVE STATUS RESPONSE
N0229                    *
N0230                    *       Response data length must be 1 byte.
N0231                    *
N0232 0DC4 0208  MRECST  LI    R8,SCNOFF        R8 = screen offset
      0DC6 0054
N0233 0DC8 A204          A     R4,R8            R8 = R8 + >8300
N0234 0DCA 0286          CI    R6,1             Is the data length 1?
      0DCC 0001
N0235 0DCE 168F          JNE   MREC05           If not, error
N0236 0DD0 06A0          BL    @BYTIN           Read a byte from the hexbus
      0DD2 0E18'
N0237 0DD4 D600          MOVB  R0,*R8           Store byte in screen offset
N0238 0DD6 10B0          JMP   MRECV3           Continue
N0239                    *
N0240                    ***************************************************************
```

```
 0015                    COPY DLU09.KSG.HEXBUS.DILLO.PREOUTIN
00001                ***************************************************************
00002                *
00003                *       PRECOMMUNICATION PROCEDURE
00004                *
00005                *       Makes sure HSK is high, then pulls BAV low for start
00006                *       of message.
00007                *
00008                *       Registers Destroyed:   R0,R1,R2
00009                *
00010 0DD8 0200   PRECOM LI    R0,TMOUT1          20 ms timeout delay
      0DDA 0600
00011 0DDC D0A0   PRECMO MOVB  @HEXSTS,R2         R2 = hexbus status
      0DDE 5FFA
00012 0DE0 0242          ANDI  R2,SHSK            Is HSK high?
      0DE2 0100
00013 0DE4 1303          JEQ   PRECM1             If yes, set up to communicate
00014 0DE6 0600          DEC   R0                 Decrement timeout count
00015 0DE8 16F9          JNE   PRECMO             Loop until HSK=high or timeout
00016 0DEA 0455          B     *R5                TIMEOUT EXIT
00017 0DEC D060   PRECM1 MOVB  @HEXCTR,R1         R1 = control byte
      0DEE 5FFC
00018 0DF0 5060          SZCB  @CLRINT,R1         Disable interrupts
      0DF2 0010'
00019 0DF4 F060          SOCB  @BAVC,R1           Pull BAV low
      0DF6 0018'
00020 0DF8 D801          MOVB  R1,@HEXCTW         Tell the control byte
      0DFA 5FFA
00021 0DFC 045B          RT                       Return to caller
00022                *
00023                ***************************************************************
```

```
00025                ***************************************************************
00026                *
00027                *       BYTE OUTPUT ROUTINE (MASTER MODE)
00028                *
00029                *       Upon entry:  R5 = Timeout return address
00030                *                    R11= Normal return address
00031                *                    R0 = Byte to be written out (MSByte)
00032                *
00033                *       Registers destroyed:  R1,R2
00034                *
00035 0DFE D060   BYOUT   MOVB @HEXSTS,R1        R1 = hexbus status
      0E00 5FFA
00036 0E02 0241           ANDI R1,WBUSY         Is WBUSY true?
      0E04 0400
00037 0E06 1305           JEQ  BYOUT2           Go to write, if yes
00038 0E08 D060   BYOUT1  MOVB @HEXSTS,R1       R1 = hexbus status
      0E0A 5FFA
00039 0E0C 0241           ANDI R1,HSKWT         Is HSKWT true?
      0E0E 8000
00040 0E10 13FB           JEQ  BYOUT1           Loop until HSKWT is 1
00041 0E12 D800   BYOUT2  MOVB R0,@HEXMTW       Write byte from R0 to XMIT reg.
      0E14 5FF8
00042 0E16 045B           RT                    Return to caller
00043                *
00044                ***************************************************************
```

```
00046                ***************************************************************
00047                *
00048                *       BYTE INPUT ROUTINE FOR OSO CHIP
00049                *
00050                *       Upon entry:  R5 = Timeout return address
00051                *                    R11= Normal return address
00052                *
00053                *       Registers destroyed:  R0,R1,R2,R3
00054                *
00055                *       Upon exit:   R0 = byte read from hexbus (MSByte)
00056                *
00057 0E18 D060   BYTIN  MOVB @HEXSTS,R1        R1 = hexbus status
      0E1A 5FFA
00058 0E1C D0C1          MOVB R1,R3            R3 = copy of current status
00059 0E1E 0241          ANDI R1,HSKRD         Is HSKRD true?                  1
      0E20 4000
00060 0E22 162F          JNE  BYTIN9           If yes, go to read byte         1
00061 0E24 0243          ANDI R3,RBUSY         Is RBUSY true?                  1
      0E26 0200
00062 0E28 160E          JNE  BYTIN4           If yes, go to special loop      1
00063 0E2A 0202   BYTIN1 LI   R2,TMOUT4        RBUSY=0,HSKRD=0,R2=time count
      0E2C 0300
00064 0E2E D060   BYTIN2 MOVB @HEXSTS,R1       R1 = hexbus status byte
      0E30 5FFA
00065 0E32 D0C1          MOVB R1,R3            R3 = copy of current status
00066 0E34 0241          ANDI R1,HSKRD         Is HSKRD true?
      0E36 4000
00067 0E38 1624          JNE  BYTIN9           If yes, go to read byte
00068 0E3A 0243          ANDI R3,SHSK          If not, is HSK high?
      0E3C 0100
00069 0E3E 16F5          JNE  BYTIN1           If not, wait normally
00070 0E40 0602          DEC  R2               If yes, decrement timeout count
00071 0E42 16F5          JNE  BYTIN2           Loop until HSKRD=1 or timeout
00072 0E44 0455          B    *R5              TIMEDUT RETURN
00073 0E46 0202   BYTIN4 LI   R2,TMOUT4        R2 = time count
      0E48 0300
00074 0E4A D060          MOVB @HEXSTS,R1       R1 = hexbus status byte
      0E4C 5FFA
00075 0E4E D0C1          MOVB R1,R3            R3 = copy of status
00076 0E50 0241          ANDI R1,HSKRD         Is HSKRD true?
      0E52 4000
00077 0E54 1616          JNE  BYTIN9           If yes, go to read byte
00078 0E56 0243          ANDI R3,SHSK          If not, is HSK high?
      0E58 0100
00079 0E5A 16F5          JNE  BYTIN4           If not, continue waiting
00080 0E5C D060   BYTIN5 MOVB @HEXSTS,R1       If yes, R1 = new hexbus status
      0E5E 5FFA
00081 0E60 D0C1          MOVB R1,R3            R3 = copy of status
00082 0E62 0241          ANDI R1,SHSK          Is HSK high?
      0E64 0100
00083 0E66 1304          JEQ  BYTIN6           If yes, skip
00084 0E68 0243          ANDI R3,HSKRD         If not, is HSKRD true?
      0E6A 4000
00085 0E6C 160A          JNE  BYTIN9           If yes, go to read byte
00086 0E6E 10EB          JMP  BYTIN4           If not, begin again
00087 0E70 0602   BYTIN6 DEC  R2               Have we timed out?
00088 0E72 16F4          JNE  BYTIN5           If not, check HSK again
00089 0E74 D060          MOVB @HEXCTR,R1       If yes, R1 = control byte
      0E76 5FFC
00090 0E78 5060          SZCB @REN,R1          Reset read enable bit
```

```
      0E7A 001C'
00091 0E7C D801          MOVB R1,@HEXCTW       Tell the control byte
      0E7E 5FFA
00092 0E80 0455          B    *R5              TIMEOUT EXIT
00093 0E82 D020  BYTIN9  MOVB @HEXRDD,R0       Read byte from read data reg.
      0E84 5FF8
00094 0E86 045B          RT                    Return to caller
00095              *
00096              ********************************************************************
```

```
 0016                       COPY DLU09.KSG.HEXBUS.DILLO.MISC
P0001                  ***************************************************************
P0002                  *
P0003                  *       CHECK USER ABORT ROUTINE
P0004                  *       This routine checks whether the user is trying to
P0005                  *       abort an operation from the keyboard by pressing
P0006                  *       BREAK (FTN-4 on the 99/4A).  Checked only during
P0007                  *       WRITE/READ to/from an RS232-type device.
P0008                  *
P0009                  *       Registers Destroyed:  R0,R7
P0010                  *       Calls Subroutines  :  CHKBRK (console routine)
P0011                  *
P0012 0E88 C00C   CHABRT MOV   R12,R0              Save CRU base in R0
P0013 0E8A C1CB          MOV   R11,R7              Save return address in R7
P0014 0E8C 06A0          BL    @CHKBRK             Is FTN-4 pressed? (console)
      0E8E 0020
P0015 0E90 1603          JNE   BRKOFF              If not, break not pressed
P0016 0E92 C300          MOV   R0,R12              If yes, restore R12
P0017 0E94 0460          B     @DEVERR             EXIT ON BREAK PRESS
      0E96 0AF6'
P0018 0E98 C300   BRKOFF MOV   R0,R12              Restore R12
P0019 0E9A C2C7          MOV   R7,R11              Restore return address
P0020 0E9C 045B          RT                        Return to caller
P0021                  *
P0022                  ***************************************************************
```

```
P0024                ****************************************************************
P0025                *
P0026                *       SPECIAL STRING TO BE SENT IF CIR. INTRPT. ON
P0027                *
P0028 0E9E    2E     TRANS   TEXT '.T=C.R=N.EC'
P0029         0EA9'  TRANEN  EQU  $
P0030 0EAA                   EVEN
P0031                *
P0032                ****************************************************************
P0033                *
P0034                *       TABLE TO CONVERT ERROR CODES FROM ALC TO CONSOLE
P0035                *       HEXBUS codes 0 - 27 handled here.
P0036                *
P0037 0EAA    00     HOMERR  BYTE >00              HEX error 00   Console error 0
P0038 0EAB    40             BYTE >40              HEX error 01   Console error 2
P0039 0EAC    40             BYTE >40              HEX error 02   Console error 6
P0040 0EAD    E0             BYTE >E0              HEX error 03   Console error 7
P0041 0EAE    E0             BYTE >E0              HEX error 04   Console error 7
P0042 0EAF    E0             BYTE >E0              HEX error 05   Console error 7
P0043 0EB0    C0             BYTE >C0              HEX error 06   Console error 6
P0044 0EB1    A0             BYTE >A0              HEX error 07   Console error 5
P0045 0EB2    E0             BYTE >E0              HEX error 08   Console error 7
P0046 0EB3    20             BYTE >20              HEX error 09   Console error 1
P0047 0EB4    E0             BYTE >E0              HEX error 0A   Console error 7
P0048 0EB5    80             BYTE >80              HEX error 0B   Console error 4
P0049 0EB6    80             BYTE >80              HEX error 0C   Console error 4
P0050 0EB7    60             BYTE >60              HEX error 0D   Console error 3
P0051 0EB8    60             BYTE >60              HEX error 0E   Console error 3
P0052 0EB9    60             BYTE >60              HEX error 0F   Console error 3
P0053 0EBA    C0             BYTE >C0              HEX error 10   Console error 6
P0054 0EBB    40             BYTE >40              HEX error 11   Console error 2
P0055 0EBC    40             BYTE >40              HEX error 12   Console error 2
P0056 0EBD    40             BYTE >40              HEX error 13   Console error 2
P0057 0EBE    40             BYTE >40              HEX error 14   Console error 2
P0058 0EBF    40             BYTE >40              HEX error 15   Console error 2
P0059 0EC0    40             BYTE >40              HEX error 16   Console error 2
P0060 0EC1    40             BYTE >40              HEX error 17   Console error 2
P0061 0EC2    C0             BYTE >C0              HEX error 18   Console error 6
P0062 0EC3    C0             BYTE >C0              HEX error 19   Console error 6
P0063 0EC4    C0             BYTE >C0              HEX error 1A   Console error 6
P0064 0EC5    C0             BYTE >C0              HEX error 1B   Console error 6
P0065         0EC6'  HEREND  EQU  $                End of table marker
P0066         001C   HERLEN  EQU  HEREND-HOMERR    Length of table
P0067 0EC6                   EVEN
P0068                *
P0069                ****************************************************************
```

```
P0071                ****************************************************************
P0072                *
P0073                *       HEXBUS COMMAND CODE TABLE
P0074                *
P0075 0EC6   00   OPENOP BYTE >00                OPEN command
P0076 0EC7   01   CLOSOP BYTE >01                CLOSE command
P0077 0EC8   02   DELOOP BYTE >02                DELETE command (open file)
P0078 0EC9   03   READOP BYTE >03                READ command
P0079 0ECA   04   WRITOP BYTE >04                WRITE command
P0080 0ECB   05   RWDOP  BYTE >05                REWIND command
P0081 0ECC   06   DELNOP BYTE >06                DELETE command (unopened file)
P0082 0ECD   07   STSOP  BYTE >07                HEX STATUS command
P0083 0ECE   08   ENSVOP BYTE >08                ENABLE SERVICE REQUEST command
P0084 0ECF   0C   VEROP  BYTE >0C                VERIFY command
P0085 0ED0   0D   FORMOP BYTE >0D                FORMAT MEDIA command
P0086 0ED1   0E   CATOP  BYTE >0E                CATALOG command
P0087 0ED2   10   BRKOP  BYTE >10                TTY BREAK command
P0088 0ED3   11   MFPOP  BYTE >11                MODIFY FILE PROTECTION command
P0089 0ED4   13   WSECOP BYTE >13                SECTOR command
P0090 0ED5   14   MFNOP  BYTE >14                MODIFY FILE NAME command
P0091 0ED6   19   LOADOP BYTE >19                LOAD command
P0092 0ED7   1A   SAVEOP BYTE >1A                SAVE command
P0093 0ED8   1B   INQSAV BYTE >1B                DO YOU SUPPORT SAVE? command
P0094 0ED9   1C   STATHC BYTE >1C                HOME COMPUTER STATUS command
P0095 0EDA   1D   VERPOP BYTE >1D                VERIFY PROGRAM FILE command
P0096 0EDB   FE   NULLOP BYTE >FE                NULL command (see interrupt)
P0097 0EDC             EVEN
P0098                *
P0099                ****************************************************************
P0100                *
P0101                *       SERVICE REQUEST MESSAGE TABLE
P0102                *
P0103 0EDC 0A00   SRQTBL DATA >0A00              Command code / LUNO          1
P0104 0EDE 0000          DATA >0000              Record number                1
P0105 0EE0 5000          DATA >5000              Buffer Length                1
P0106 0EE2 0000          DATA >0000              Data Length                  1
P0107      0EE4' SRQEND EQU  $
P0108                *
P0109                ****************************************************************
```

```
P0111                   ********************************************************************
P0112                   *
P0113                   *       MAPPER COMMAND CODE
P0114                   *
P0115 0EE4    01        MAPRCO BYTE >01
P0116 0EE6              EVEN
P0117                   *
P0118                   ********************************************************************
P0119                   *
P0120                   *       PROGRAM CONSTANT TABLE
P0121                   *
P0122 0EE6 0080  H0080  DATA >0080              H0080 = >0080
P0123 0EE8 0200  H0200  DATA >0200              H0200 = >0200
P0124 0EEA 0400  H0400  DATA >0400              H0400 = >0400
P0125 0EEC 0004  H0004  DATA >0004              H0004 = >0004
P0126 0EEE 0009  H0009  DATA >0009              H0009 = >0009
P0127      0EF0' H00    EQU  $                  H00   = >00
P0128 0EF0 000A  D10    DATA 10                 D10   = 10
P0129 0EF2 0AFE  H0A    DATA >0AFE              H0A   = >0A
P0130      0EF3' HFE    EQU  $-1                HFE   = >FE
P0131 0EF4 0214  H02    DATA >0214              H02   = >02
P0132 0EF6 8848  H88    DATA >8848              H88   = >88
P0133      0EF7' H48    EQU  $-1                H48   = >48
P0134 0EF8 0110  H01    DATA >0110              H01   = >01
P0135      0EF9' H10    EQU  $-1                H10   = >10
P0136 0EFA 4004  H40    DATA >4004              H40   = >40
P0137      0EFB' H04    EQU  $-1                H04   = >04
P0138      0EFB' NOTOPN EQU  H04                File not open error code (HEX)
P0139 0EFC 09E0  H09    DATA >09E0              H09   = >09
P0140 0EFE 0100  H0100  DATA >0100              H0100 = >0100
P0141 0F00 00FF  H00FF  DATA >00FF              H00FF = >00FF
P0142      0F01' HFF    EQU  $-1                HFF   = >FF
P0143 0F02 0600  H06    DATA >0600              H0600 = >0600
P0144 0F04 0300  H03    DATA >0300              H0300 = >0300
P0145      0F06' H80    EQU  $                  H80   = >80
P0146 0F06 8000  H8000  DATA >8000              H8000 = >8000
P0147      0F08' H08    EQU  $                  H08   = >08
P0148 0F08 0800  H0800  DATA >0800              H0800 = >0800
P0149 0F0A 2000  H2000  DATA >2000              H2000 = >2000
P0150 0F0C 1000  H1000  DATA >1000              H1000 = >1000
P0151 0F0E 4000  H4000  DATA >4000              H4000 = >4000
P0152 0F10 0CFF  H0C    DATA >0CFF              H0C   = >0C
P0153 0F12   46  D70    BYTE 70                 D70   = 70
P0154 0F13   14  D20    BYTE 20                 D20   = 20
P0155 0F14   07  H07    BYTE >07                H07   = >07
P0156      0F14' EOFERR EQU  H07                EOF error code (HEX)
P0157 0F16              EVEN
P0158                   *
P0159                   ********************************************************************
```

```
 0017                    COPY DLU09.KSG.HEXBUS.DILLO.INTRPT
Q0001              ************************************************************
Q0002              *
Q0003              *       INTERRUPT SERVICE ROUTINE
Q0004              *       Currently supports: RS232/1 circular interrupt
Q0005              *                           RS232/2 circular interrupt
Q0006              *                           Slave mode interrupt
Q0007              *
Q0008              *       >8300,1 = 0 VDP; 1 CPU; 1X USER  buffer/inrpt loc.
Q0009              *
Q0010              *       Registers Destroyed:  R1,R2,R3,R4,R5,R6,R7,R8,R10,R11
Q0011              *                             R7 is used to hold return addr.
Q0012              *                             R11 has low-level return addr.
Q0013              *
Q0014              *       Calls Subroutines:    PRECOM, BYOUT, BYTIN, VDPWD
Q0015              *
Q0016              *       This routine may use: R1,R2,R3,R4,R5,R6,R7,R8,R10
Q0017              *       Before exiting, R8 must be cleared.
Q0018              *
Q0019 0F16 D1E0    INTEN0 MOVB @HEXCTR,R7        R7 = hexbus control byte
      0F18 5FFC
Q0020 0F1A 0247           ANDI R7,>F000         Any interrupts enabled?
      0F1C F000
Q0021 0F1E 1601           JNE  INTEN1           If yes, continue processing
Q0022 0F20 045B           RT                    If not, it wasn't me (i hope)
Q0023 0F22 D160    INTEN1 MOVB @HEXSTS,R5       R5 = hexbus status byte
      0F24 5FFA
Q0024 0F26 D185           MOVB R5,R6            R6 = copy of current status
Q0025 0F28 0245           ANDI R5,>F000         Mask off only interrupt bits
      0F2A F000
Q0026 0F2C 0547           INV  R7               R7 = INV interrupt enable mask
Q0027 0F2E 4147           SZC  R7,R5            R5 = i.mask AND intrpt. occured
Q0028 0F30 1601           JNE  INTEN2           If nonzero, continue processing
Q0029 0F32 045B           RT                    If 0, it wasn't me (i'm sure)
Q0030 0F34 02A4    INTEN2 STWP R4               Set up scratchpad RAM pointer
Q0031 0F36 0224           AI   R4,->E0          R4 = >8300
      0F38 FF20
Q0032 0F3A C064           MOV  @CBUFST(R4),R1   Is content of >8300,1 positive?
      0F3C 0000
Q0033 0F3E 1503           JGT  INTENC           If yes, continue processing
Q0034 0F40 1302           JEQ  INTENC           If not, branch to user-defined
Q0035 0F42 0A21           SLA  R1,2             Entry address= 4*LS 14 bits(R1)
Q0036 0F44 0451           B    *R1              Branch to user-defined routine
Q0037 0F46 2160    INTENC COC  @BAVAIE,R5       Did BAV go HIGH? (SLAVE MODE)
      0F48 0016'
Q0038 0F4A 1305           JEQ  SLINTR           If yes, circular interrupt
Q0039 0F4C 2160           COC  @BAVIAE,R5       Did BAV go LOW?  (MASTER MODE)
      0F4E 0014'
Q0040 0F50 130C           JEQ  INTEN6           If yes, end of current message
Q0041 0F52 C1CB           MOV  R11,R7           R7 = return address
Q0042 0F54 101F           JMP  INTXXX           Neither interrupt occured
Q0043 0F56 D160    SLINTR MOVB @HEXCTR,R5       Get a copy of control register
      0F58 5FFC
Q0044 0F5A F160           SOCB @REN,R5          Set REN bit
      0F5C 001C'
Q0045 0F5E 5160           SZCB @BAVAIE,R5       Disable this interrupt
      0F60 0016'
Q0046 0F62 D805           MOVB R5,@HEXCTW       Write to the control register
      0F64 5FFA
Q0047 0F66 C1CB           MOV  R11,R7           R7 = return address
```

```
Q0048 0F68 1015          JMP   INTXXX          Exit to wait for next message
Q0049 0F6A C200  INTEN6  MOV   R0,R8           Save R0 in R8
Q0050 0F6C C1CB          MOV   R11,R7          Save return address in R7
Q0051 0F6E D046          MOVB  R6,R1           R1 = copy of current status
Q0052 0F70 0241          ANDI  R1,SBAV         Is BAV really low?
      0F72 0800
Q0053 0F74 1306          JEQ   INTENX          If not, ignore this interrupt
Q0054 0F76 D060          MOVB  @HEXCTR,R1      If yes, did I drop BAV myself?
      0F78 5FFC
Q0055 0F7A 2060          COC   @BAVC,R1        Check if I set BAV low
      0F7C 0018'
Q0056 0F7E 1305          JEQ   INTX            If yes, exit
Q0057 0F80 100D          JMP   INTEN3          If not, service
Q0058 0F82 D060  INTENX  MOVB  @HEXCTR,R1      Get a copy of the control byte
      0F84 5FFC
Q0059 0F86 5060          SZCB  @DISABL,R1      Disable communication
      0F88 0012'
Q0060 0F8A F060  INTX    SOCB  @BAVIAE,R1      Enable service req. interrupt
      0F8C 0014'
Q0061 0F8E D801          MOVB  R1,@HEXCTW      Write to the chip
      0F90 5FFA
Q0062 0F92 C008          MOV   R8,R0           Restore R0
Q0063 0F94 04C8  INTXXX  CLR   R8              Clear R8 before exiting
Q0064 0F96 0457          B     *R7             Return to console
```

```
Q0066                ***************************************************************
Q0067                *
Q0068                *       CIRCULAR INTERRUPT HANDLER
Q0069                *
Q0070 0F98 0460  INTNUL B     @SENNUL           Go send a null command
      0F9A 106C'
Q0071 0F9C 2D43  INTEN3 XOP   3,5               Map in DSR ram area
Q0072 0F9E D2A0         MOVB  @SRVREQ,R10       Get service request dev. code
      0FA0 E072
Q0073 0FA2 13FA         JEQ   INTNUL            No one is enabled for srv.req.
Q0074 0FA4 D820         MOVB  @MAPRCO,@MAPPER   Restore mapper
      0FA6 0EE4'
      0FA8 8810
Q0075 0FAA 1000         NOP                     Let Mapper finish operation   2
Q0076 0FAC 098A         SRL   R10,8             Make it a word
Q0077 0FAE 0205  INTEN5 LI    R5,INTEX          Timeout exit
      0FB0 1068'
Q0078 0FB2 06A0         BL    @PRECOM           Prepare to communicate
      0FB4 0DD8'
Q0079 0FB6 D020         MOVB  @HEXCTR,R0        Get copy of control register
      0FB8 5FFC
Q0080 0FBA F020         SOCB  @REN,R0           Set REN bit
      0FBC 001C'
Q0081 0FBE F020         SOCB  @WEN,R0           Set WEN bit
      0FC0 001A'
Q0082 0FC2 D800         MOVB  R0,@HEXCTW        Write a new control value
      0FC4 5FFA
Q0083 0FC6 C00A         MOV   R10,R0            R0 = device code
Q0084 0FC8 06C0         SWPB  R0                Move code to MSByte of R0
Q0085 0FCA 06A0         BL    @BYOUT            Send device code over hexbus
      0FCC 0DFE'
Q0086 0FCE 0206         LI    R6,SRQTBL         R6 => service request table   1
      0FD0 0EDC'
Q0087 0FD2 D036  INTEN4 MOVB  *R6+,R0           Get a byte into R0            1
Q0088 0FD4 06A0         BL    @BYOUT            Send a byte over the bus      1
      0FD6 0DFE'
Q0089 0FD8 0286         CI    R6,SRQEND         End of service req. message?  1
      0FDA 0EE4'
Q0090 0FDC 16FA         JNE   INTEN4            If not, continue sending      1
Q0091 0FDE 06A0         BL    @BYTIN            Read first byte of response
      0FE0 0E18'
Q0092 0FE2 C180         MOV   R0,R6             MSByte of R6 = first byte
Q0093 0FE4 06A0         BL    @BYTIN            Read second byte of reponse
      0FE6 0E18'
Q0094 0FE8 06C6         SWPB  R6                LSByte of R6 = first byte
Q0095 0FEA D180         MOVB  R0,R6             R6 = returned data length
Q0096 0FEC C186         MOV   R6,R6             Is there any data?
Q0097 0FEE 1323         JEQ   INTLN3            If not, skip
```

```
00099                   ************************************************************
00100                   *
00101                   *       Receive data from device and store into cir. buffer
00102                   *
00103 0FF0 06A0  INTLN2 BL    @BYTIN              Read a byte from the bus
      0FF2 0E18'
00104 0FF4 C064         MOV   @CBUFST(R4),R1      R1 = content of >8300
      0FF6 0000
00105 0FF8 2060         COC   @H4000,R1           Is buffer in CPU?
      0FFA 0F0E'
00106 0FFC 1356         JEQ   BUFCPU              If yes, process CPU buffer
00107 0FFE D064         MOVB  @CBUFOF(R4),R1      R1 = content of >8304
      1000 0004
00108 1002 B060         AB    @H01,R1             Increment by one
      1004 0EF8'
00109 1006 9901         CB    R1,@CBUFLN(R4)      Exceeding buffer?
      1008 0002
00110 100A 1201         JLE   INTLN               If not, skip
00111 100C 04C1         CLR   R1                  R1 points to beginning of buf.
00112 100E 9901  INTLN  CB    R1,@CBUFRO(R4)      Buffer overflow?
      1010 0003
00113 1012 1604         JNE   INTLN1              If not, continue
00114 1014 0200         LI    R0,>FE00            R0 = code for overwrite
      1016 FE00
00115 1018 D064         MOVB  @CBUFOF(R4),R1      Rewrite R1
      101A 0004
00116 101C D901  INTLN1 MOVB  R1,@CBUFOF(R4)      Update offset
      101E 0004
00117 1020 0981         SRL   R1,8                Make R1 a word
00118 1022 A064         A     @CBUFST(R4),R1      Add base address of buffer
      1024 0000
00119 1026 0241         ANDI  R1,>3FFF            Just to be safe
      1028 3FFF
00120 102A 06A0         BL    @VDPWD              Set up for VDP write
      102C 09FA'
00121 102E DBC0         MOVB  R0,@VWD(R15)        Write byte to VDP RAM
      1030 FFFE
00122 1032 0606  INTLN5 DEC   R6                  One less byte to read
00123 1034 16DD         JNE   INTLN2              Loop until all data received
00124 1036 06A0  INTLN3 BL    @BYTIN              Read the status byte from bus
      1038 0E18'
00125 103A 9800         CB    R0,@H0A             'It was not me' error?
      103C 0EF2'
00126 103E 1314         JEQ   INTEX               If yes, exit
00127 1040 D000         MOVB  R0,R0               Was there any error?
00128 1042 1312         JEQ   INTEX               If not, exit
00129 1044 C064         MOV   @CBUFST(R4),R1      R1 = content >8300
      1046 0000
00130 1048 2060         COC   @H4000,R1           Is buffer in CPU?
      104A 0F0E'
00131 104C 1346         JEQ   BUFCP3              If yes, handle CPU buffer
00132 104E 0200         LI    R0,>FF00            If not, R0 = hard error
      1050 FF00
00133 1052 D064         MOVB  @CBUFOF(R4),R1      Overwrite last byte
      1054 0004
00134 1056 0981         SRL   R1,8                Make R1 a word
00135 1058 A064         A     @CBUFST(R4),R1      Find last byte in buffer
      105A 0000
00136 105C 0241         ANDI  R1,>3FFF            Just to be safe
      105E 3FFF
```

```
G0137 1060 06A0          BL   @VDPWD             Set up for VDP write
      1062 09FA'
G0138 1064 DBC0          MOVB RO,@VWD(R15)       Write byte to VDP RAM
      1066 FFFE
G0139 1068 0460  INTEX   B    @INTENX            Exit from interrupt routine
      106A 0F82'
G0140 106C 0205  SENNUL  LI   R5,INTEX           R5 = TIMEOUT EXIT
      106E 1068'
G0141 1070 D820          MOVB @MAPRCO,@MAPPER    Restore mapper
      1072 0EE4'
      1074 8810
G0142 1076 1000          NOP                     Let Mapper finish operation   2
G0143 1078 06A0          BL   @PRECOM            Set up for communication
      107A 0DD8'
G0144 107C D020          MOVB @HEXCTR,RO         Get copy of control byte
      107E 5FFC
G0145 1080 F020          SOCB @WEN,RO            Set WEN bit
      1082 001A'
G0146 1084 D800          MOVB RO,@HEXCTW         Write the control info
      1086 5FFA
G0147 1088 04C0          CLR  RO                 RO =0
G0148 108A 06A0          BL   @BYOUT             Send device code = 0
      108C 0DFE'
G0149 108E D020          MOVB @NULLOP,RO         Get null command
      1090 0EDB'
G0150 1092 06A0          BL   @BYOUT             Send null command code
      1094 0DFE'
G0151 1096 0206          LI   R6,7               Seven zero bytes to send
      1098 0007
G0152 109A 06A0          BL   @SENDZ             Send 7 zero bytes over bus
      109C 0A9E'
G0153 109E D020  NULL1   MOVB @HEXSTS,RO         RO = hexbus status byte
      10A0 5FFA
G0154 10A2 0240          ANDI RO,HSKWT           Write complete?
      10A4 8000
G0155 10A6 13FB          JEQ  NULL1              If not, keep watching
G0156 10A8 10DF          JMP  INTEX              If yes, exit
```

```
00158                ***************************************************************
00159                *
00160                *       HANDLE BUFFER IN CPU
00161                *       1st word = buffer address
00162                *       2nd word = buffer size
00163                *       3rd word = user read offset
00164                *       4th word = interrupt routine write offset
00165                *
00166 10AA 06C2   BUFCPU SWPB R2                 LSByte R2 = data
00167 10AC D080          MOVB RO,R2              MBSyte R2 = data from RO
00168 10AE 0A21          SLA  R1,2               R1 points to control area
00169 10B0 0221          AI   R1,6               R1 points to write offset
      10B2 0006
00170 10B4 C011          MOV  *R1,RO             RO = write offset
00171 10B6 0221          AI   R1,-4              R1 points to buffer size
      10B8 FFFC
00172 10BA 0580          INC  RO                 Advance read offset by 1
00173 10BC 8C40          C    RO,*R1+            Exceeding buffer size?
00174 10BE 1201          JLE  BUFCP1             If not, skip
00175 10C0 04C0          CLR  RO                 RO = 0 (wrap around)
00176 10C2 8C40   BUFCP1 C    RO,*R1+            Buffer overflow?
00177 10C4 1603          JNE  BUFCP2             If not, skip
00178 10C6 D0A0          MOVB @HFE,R2            Overwrite last byte
      10C8 0EF3'
00179 10CA C011          MOV  *R1,RO             RO = offset to last byte
00180 10CC C440   BUFCP2 MOV  RO,*R1             Update write offset
00181 10CE 0221          AI   R1,-6              R1 points to buffer address
      10D0 FFFA
00182 10D2 A011          A    *R1,RO             RO = true address
00183 10D4 D402          MOVB R2,*RO             Store data in CPU RAM
00184 10D6 06C2          SWPB R2                 Restore R2
00185 10D8 10AC          JMP  INTLN5             Continue
00186 10DA 0A21   BUFCP3 SLA  R1,2               R1 points to control area
00187 10DC 0221          AI   R1,6               R1 points to write offset
      10DE 0006
00188 10E0 C011          MOV  *R1,RO             RO = write offset
00189 10E2 0221          AI   R1,-6              R1 points to buffer start addr.
      10E4 FFFA
00190 10E6 A011          A    *R1,RO             RO points to last byte in buf.
00191 10E8 D420          MOVB @HFF,*RO           Overwrite last byte
      10EA 0F01'
00192 10EC 10BD          JMP  INTEX              Exit from interrupt routine
00193                *
00194                ***************************************************************
```

```
 0018                       COPY DLU09.KSG.HEXBUS.DILLO.RESET
R0001                *************************************************************
R0002                *
R0003                *      RESET THE HEXBUS (ACCESSED BY OPEN COMMAND)
R0004                *
R0005 10EE D024  RESENT MOVB @OPCODE(R4),R0   Is the opcode an open?
      10F0 004A
R0006 10F2 1307         JEQ  RESEN1          If yes, continue with reset
R0007 10F4 9800         CB   R0,@H01         Is the opcode a close?
      10F6 0EF8'
R0008 10F8 1338         JEQ  RESENX          If yes, do nothing and exit
R0009 10FA 0460  RESBAD B    @BADOP          Illegal opcode error
      10FC 0AE4'
R0010 10FE 0460  RESBA1 B    @BADATO         Bad open attributes error
      1100 0ADE'
R0011 1102 D024  RESEN1 MOVB @DEVCOD(R4),R0  Is device code equal to zero?
      1104 005D
R0012 1106 16FB         JNE  RESBA1          If not, error
R0013 1108 C1C7         MOV  R7,R7           More characters after dev.code?
R0014 110A 16F9         JNE  RESBA1          If yes, error
R0015 110C 0205         LI   R5,RESEN2       R5 = timeout return address
      110E 1144'
R0016 1110 04E4         CLR  @COMSTS(R4)     Clear communication status
      1112 006A
R0017 1114 06A0         BL   @PRECOM         Set up for communication
      1116 0DD8'
R0018 1118 D020         MOVB @HEXCTR,R0      Get copy of control byte
      111A 5FFC
R0019 111C F020         SOCB @WEN,R0         Set WEN bit
      111E 001A'
R0020 1120 D800         MOVB R0,@HEXCTW      Write new control reg.
      1122 5FFA
R0021 1124 04C0         CLR  R0              R0 = 0
R0022 1126 06A0         BL   @BYOUT          Send zero device code
      1128 0DFE'
R0023 112A D020         MOVB @RESTOP,R0      Get reset command code
      112C 116E'
R0024 112E 06A0         BL   @BYOUT          Send reset command to bus
      1130 0DFE'
R0025 1132 0206         LI   R6,7            Seven zero bytes to send
      1134 0007
R0026 1136 06A0         BL   @SENDZ          Send 7 zeros to bus
      1138 0A9E'
R0027 113A D1A0  RES9   MOVB @HEXSTS,R6      R6 = hexbus status byte
      113C 5FFA
R0028 113E 0246         ANDI R6,HSKWT        Is HSKWT true? (of last byte)
      1140 8000
R0029 1142 13FB         JEQ  RES9            If not, wait until it is
R0030 1144 D1A0  RESEN2 MOVB @HEXCTR,R6      Disable communications
      1146 5FFC
R0031 1148 51A0         SZCB @DISABL,R6
      114A 0012'
R0032 114C D806         MOVB R6,@HEXCTW
      114E 5FFA
R0033 1150 04C6         CLR  R6              Clear disk OPEN buffers
R0034 1152 D806         MOVB R6,@LUN250
      1154 E012
R0035 1156 D806         MOVB R6,@LUN251
      1158 E029
R0036 115A D806         MOVB R6,@LUN252
```

```
      115C E040
R0037 115E D806          MOVB R6,@LUN253
      1160 E057
R0038 1162 D806          MOVB R6,@SRVREQ        Clear service request device
      1164 E072
R0039 1166 0460  RESEN3  B    @FOREN3           Use FORMAT to handle LRL
      1168 0456'
R0040 116A 0460  RESENX  B    @EXIT             Exit from reset routine
      116C 0B3E'
R0041 116E FF00  RESTOP  DATA >FF00             Reset hexbus command
R0042                    *
R0043                    ********************************************************************
```

```
 0019                       COPY DLU09.KSG.HEXBUS.DILLO.RAWDATA
S0001                  ***************************************************************
S0002                  *
S0003                  *       TRANSFER RAW DATA COMMAND (.TR switch)
S0004                  *       Supports open, close, read, and write
S0005                  *
S0006  1170 D024  TRFENT MOVB  @OPCODE(R4),R0    R0 = current opcode
       1172 004A
S0007  1174 9800         CB    R0,@H04           Is opcode <= 3?
       1176 0EFB'
S0008  1178 14C0         JHE   RESBAD            If not, error
S0009  117A 9800         CB    R0,@H01           Is it a close opcode?
       117C 0EF8'
S0010  117E 13F5         JEQ   RESENX            If yes, exit
S0011  1180 9800         CB    R0,@H02           Is it a read opcode?
       1182 0EF4'
S0012  1184 133B         JEQ   TRFRD             If yes, go to read routine
S0013  1186 9800         CB    R0,@H03           Is it a write opcode?
       1188 0F04'
S0014  118A 130D         JEQ   TRFWT             If yes, go to write routine
S0015                  *
S0016                  ***************************************************************
S0017                  *
S0018                  *       TRANSFER RAW DATA *** OPEN ***
S0019                  *
S0020  118C 9824  TRFOPN CB    @FLGSTS(R4),@H10  Attrib. = disp,var,seq,update?
       118E 004B
       1190 0EF9'
S0021  1192 16B5         JNE   RESBA1            If not, error
S0022  1194 C024         MOV   @LRECLN(R4),R0    Does LRL = 0?
       1196 0052
S0023  1198 1303         JEQ   TROPN1            If yes, continue
S0024  119A 8800         C     R0,@H0009         If not, is LRL < 9?
       119C 0EEE'
S0025  119E 1AAF         JL    RESBA1            If yes, error
S0026  11A0 04E4  TROPN1 CLR   @COMSTS(R4)       Clear error status
       11A2 006A
S0027  11A4 10E0         JMP   RESEN3            Exit
S0028                  *
S0029                  ***************************************************************
```

```
S0031                    ************************************************************
S0032                    *
S0033                    *       WRITE RAW DATA TO HEXBUS
S0034                    *       BAV should remain low on exit
S0035                    *
S0036 11A6 C1E4  TRFWT   MOV   @CHRCNT(R4),R7    R7 = number of bytes to write
      11A8 004E
S0037 11AA 0205          LI    R5,TRFWT2         R5 = timeout error address
      11AC 11F0'
S0038 11AE 06A0          BL    @PRECOM           Prepare to communicate
      11B0 0DD8'
S0039 11B2 D060          MOVB  @HEXCTR,R1        Get control register
      11B4 5FFC
S0040 11B6 F060          SOCB  @REN,R1           Set REN bit
      11B8 001C'
S0041 11BA F060          SOCB  @WEN,R1           Set WEN bit
      11BC 001A'
S0042 11BE D801          MOVB  R1,@HEXCTW        Tell the chip
      11C0 5FFA
S0043 11C2 C064          MOV   @BUFADR(R4),R1    R1 = data buffer address
      11C4 004C
S0044 11C6 06A0          BL    @SUREAD           Set up for read
      11C8 0A56'
S0045 11CA 06A0  TRFWT1  BL    @RDBYTO           Read byte into R0
      11CC 0A16'
S0046 11CE 06A0          BL    @BYOUT            Send byte out to hexbus
      11D0 0DFE'
S0047 11D2 0607          DEC   R7                One less byte to send
S0048 11D4 16FA          JNE   TRFWT1            Continue sending bytes out
S0049                    *
S0050                    ************************************************************
S0051                    *
S0052                    *       TEST FOR A  RESPONSE MESSAGE
S0053                    *
S0054 11D6 0202  TRFWT6  LI    R2,TMOUT4         R2 = timeout counter
      11D8 0300
S0055 11DA D060  TRFWT3  MOVB  @HEXSTS,R1        R1 = hexbus status
      11DC 5FFA
S0056 11DE D0C1          MOVB  R1,R3             R3 = copy of current status
S0057 11E0 0241          ANDI  R1,HSKRD          Is there a byte ready to read?
      11E2 4000
S0058 11E4 1607          JNE   TRFWT4            If yes, then exit with BAV low
S0059 11E6 0243          ANDI  R3,SHSK           Is HSK low?
      11E8 0100
S0060 11EA 16F5          JNE   TRFWT6            If yes, don't timeout
S0061 11EC 0602          DEC   R2                Decrement timeout counter
S0062 11EE 16F5          JNE   TRFWT3            Loop until byte ready / timeout
S0063 11F0 0460  TRFWT2  B     @TIMOUT           TIMEOUT EXIT
      11F2 0AF0'
S0064 11F4 0460  TRFWT4  B     @EXIT1            Exit while holding HSK low
      11F6 0B4E'
S0065                    *
S0066                    ************************************************************
```

```
S0068                   ***************************************************************
S0069                   *
S0070                   *       READ RAW DATA FROM HEXBUS
S0071                   *       Upon entry, BAV and HSK must be low
S0072                   *
S0073 11F8 0460  TOFRD2 B     @TRFRD2           Bus error exit
      11FA 129A'
S0074 11FC D060  TRFRD  MOVB  @HEXSTS,R1        R1 = hexbus status
      11FE 5FFA
S0075 1200 D0C1         MOVB  R1,R3             R3 = copy of current status
S0076 1202 0241         ANDI  R1,SBAV           Is BAV high?
      1204 0800
S0077 1206 13F8         JEQ   TOFRD2            If yes, error
S0078 1208 0243         ANDI  R3,SHSK           Is HSK high?
      120A 0100
S0079 120C 13F5         JEQ   TOFRD2            If yes, error
S0080 120E C064         MOV   @BUFADR(R4),R1    R1 = data buffer address
      1210 004C
S0081 1212 06A0         BL    @SUWRIT           Set up for write to RAM
      1214 0A72'
S0082 1216 0205         LI    R5,TRFRD7         R5 = timeout error exit
      1218 1296'
S0083 121A C1E4         MOV   @LRECLN(R4),R7    R7 = Logical Record Length
      121C 0052
S0084 121E 0227         AI    R7,-3             R7 = R7 - 3 (Buffer size)
      1220 FFFD
S0085 1222 06A0         BL    @BYTIN9           Read a byte from the bus
      1224 0E82'
S0086 1226 D900         MOVB  R0,@RDATLN+1(R4)  Store in scratchpad
      1228 0067
S0087 122A 06A0         BL    @WTBYTO           Write byte from R0
      122C 0A8E'
S0088 122E 06A0         BL    @BYTIN            Read another byte from the bus
      1230 0E18'
S0089 1232 D900         MOVB  R0,@RDATLN(R4)    Store in scratchpad
      1234 0066
S0090 1236 06A0         BL    @WTBYTO           Write byte from R0
      1238 0A8E'
S0091 123A C1A4         MOV   @RDATLN(R4),R6    R6 = data length returned
      123C 0066
S0092 123E 81C6         C     R6,R7             Does data len. = buf.size - 3?
S0093 1240 1202         JLE   TRFRD1            If yes, continue
S0094 1242 6187         S     R7,R6             R6 = extra bytes to ignore
S0095 1244 1002         JMP   TRFRD3            Continue
S0096 1246 C1C6  TRFRD1 MOV   R6,R7             R7 = data length
S0097 1248 04C6         CLR   R6                R6 = extra bytes = 0
S0098 124A C1C7  TRFRD3 MOV   R7,R7             Any data to read?
S0099 124C 1306         JEQ   TRFRD9            If not, skip
S0100 124E 06A0  TRFRD8 BL    @BYTIN            Read a byte from the bus
      1250 0E18'
S0101 1252 06A0         BL    @WTBYTO           Write byte from R0
      1254 0A8E'
S0102 1256 0607         DEC   R7                One less byte to read
S0103 1258 16FA         JNE   TRFRD8            Continue reading bytes
S0104 125A C186  TRFRD9 MOV   R6,R6             Any extra bytes to ignore?
S0105 125C 130E         JEQ   TRFRD5            If not, continue
S0106 125E 06A0  TRFRD4 BL    @BYTIN            Read an extra byte from the bus
      1260 0E18'
S0107 1262 0606         DEC   R6                One less byte to ignore
S0108 1264 16FC         JNE   TRFRD4            Continue ignoring until done
```

```
S0109 1266 06A0          BL   @BYTIN            Get the status byte and toss it
      1268 0E18'
S0110 126A 0200          LI   R0,12*256         R0 = buffer size error
      126C 0C00
S0111 126E D900          MOVB R0,@COMSTS(R4)    Return buffer full status
      1270 006A
S0112 1272 06A0          BL   @WTBYTO           Write byte from R0
      1274 0A8E'
S0113 1276 0460  TRFRD6  B    @SIZERR           Exit on buffer size error
      1278 0AEA'
S0114 127A 06A0  TRFRD5  BL   @BYTIN            Read the status byte
      127C 0E18'
S0115 127E D900          MOVB R0,@COMSTS(R4)    Store it in RAM
      1280 006A
S0116 1282 06A0          BL   @WTBYTO           Write byte from R0
      1284 0A8E'
S0117 1286 C024          MOV  @RDATLN(R4),R0    Number of bytes recv.
      1288 0066
S0118 128A 0220          AI   R0,>0003            is data len + 3
      128C 0003
S0119 128E C900          MOV  R0,@CHRCNT(R4)    Update character count in PAB
      1290 004E
S0120 1292 0460          B    @EXIT             Normal return
      1294 0B3E'
S0121 1296 0460  TRFRD7  B    @TIMOUT           TIMEOUT ERROR EXIT
      1298 0AF0'
S0122 129A 0200  TRFRD2  LI   R0,27*256         R0 = bus error code
      129C 1B00
S0123 129E D900          MOVB R0,@COMSTS(R4)    Store in RAM
      12A0 006A
S0124 12A2 0460          B    @DEVERR           BUS ERROR EXIT
      12A4 0AF6'
S0125                    *
S0126                    ********************************************************************
```

```
 0020                    COPY DLU09.KSG.HEXBUS.DILLO.CATALOG
T0001                ***************************************************************
T0002                *
T0003                *       CATALOG COMMAND ROUTINE
T0004                *       Supports open, read, and close
T0005                *
T0006  12A6 D024   CATENT MOVB @OPCODE(R4),R0    Is it an open opcode?
       12A8 004A
T0007  12AA 130A          JEQ  CATOPN            If yes, jump to open catalog
T0008  12AC 9800          CB   R0,@H01           Is it a close opcode?
       12AE 0EF8'
T0009  12B0 1305          JEQ  CATENX            If yes, exit
T0010  12B2 9800          CB   R0,@H02           Is it a read opcode?
       12B4 0EF4'
T0011  12B6 1312          JEQ  CATRD             If yes, jump to read catalog
T0012  12B8 0460   CATBAD B    @BADOP            Illegal opcode exit
       12BA 0AE4'
T0013  12BC 0460   CATENX B    @EXIT              Normal exit
       12BE 0B3E'
T0014  12C0 9824   CATOPN CB   @FLGSTS(R4),@H04 Attrib.= disp,seq,var,input
       12C2 004B
       12C4 0EFB'
T0015  12C6 1302          JEQ  CATOP1            If yes, continue
T0016  12C8 0460   CATOP2 B    @BADATT            Bad attribute exit
       12CA 0358'
T0017  12CC C024   CATOP1 MOV  @LRECLN(R4),R0    Does LRL = 0?
       12CE 0052
T0018  12D0 13FB          JEQ  CATOP2            If yes, error
T0019  12D2 D920          MOVB @H00,@COMSTS(R4) Clear ALC error byte
       12D4 0EF0'
       12D6 006A
T0020  12D8 0460          B    @COMRTN           Normal exit
       12DA 0AAC'
T0021                *
T0022                ***************************************************************
T0023                *
T0024                *       CATALOG READ SECTION
T0025                *
T0026  12DC D920   CATRD  MOVB @CATOP,@COMMAD(R4) Command = catalog
       12DE 0ED1'
       12E0 005E
T0027  12E2 0460          B    @READ1            Use read routine
       12E4 0694'
T0028                *
T0029                ***************************************************************
```

```
 0021                       COPY DLU09.KSG.HEXBUS.DILLO.SLAVE
U0001                  ***************************************************************
U0002                  *
U0003                  *       ENTRY TO SLAVE MODE ROUTINES  (.SL. switch)
U0004                  *       Open, Close, Read, Write are supported.
U0005                  *
U0006                  *       This routine tests for a valid slave mode operation
U0007                  *       and then branches to the appropriate slave routine.
U0008                  *
U0009 12E6 D0A4   SLVENT MOVB  @OPCODE(R4),R2    R2 = current opcode
      12E8 004A
U0010 12EA 0982          SRL   R2,8              Make it a word
U0011 12EC 0280          CI    R0,3              Is opcode 0 - 3?
      12EE 0003
U0012 12F0 1B04          JH    SLVBAD            If not, error
U0013 12F2 0A12          SLA   R2,1              R2 = R2*2 for indexing table
U0014 12F4 C062          MOV   @SLVTAB(R2),R1    R1 = slave entry address
      12F6 12FE'
U0015 12F8 0451          B     *R1               Go to the slave routine
U0016 12FA 0460   SLVBAD B     @BADOP            ERROR EXIT
      12FC 0AE4'
U0017 12FE 1306' SLVTAB  DATA  SLVOPN            Entry address for slave open
U0018 1300 134A'         DATA  SLVCLS            Entry address for slave close
U0019 1302 13D8'         DATA  SLVRD             Entry address for slave read
U0020 1304 14A0'         DATA  SLVWR             Entry address for slave write
U0021                  *
U0022                  ***************************************************************
```

```
U0024                    ************************************************************
U0025                    *
U0026                    *       SLAVE OPEN ROUTINE
U0027                    *       This routine handles the LRL, makes sure device code
U0028                    *       is between 0 and 255, ignores any present bus activ-
U0029                    *       ity, and enables an interrupt on rising BAV.
U0030                    *
U0031 1306 9824  SLVOPN CB    @FLGSTS(R4),@H10 Display, var., seq., update?
      1308 004B
      130A 0EF9'
U0032 130C 16DD         JNE   CATOP2           If not, error
U0033 130E C024         MOV   @LRECLN(R4),R0   Does LRL = 0?
      1310 0052
U0034 1312 1604         JNE   SLVOP2           If not, continue
U0035 1314 C920         MOV   @H00FF,@LRECLN(R4) Return default LRL (255)
      1316 0F00'
      1318 0052
U0036 131A 1003         JMP   SLVOP3           Continue
U0037 131C 8800  SLVOP2 C     R0,@H0009        Is LRL at least >= 9?
      131E 0EEE'
U0038 1320 1AD3         JL    CATOP2           If not, error
U0039 1322 D020  SLVOP3 MOVB  @HEXSTS,R0       R0 = hexbus status byte
      1324 5FFA
U0040 1326 0240         ANDI  R0,SBAV          Is BAV low?
      1328 0800
U0041 132A 1605         JNE   SLBAVL           If yes, ignore this message
U0042 132C D020         MOVB  @HEXCTR,R0       If not, set REN bit
      132E 5FFC
U0043 1330 F020         SOCB  @REN,R0
      1332 001C'
U0044 1334 1006         JMP   SLOPEX           And exit open routine
U0045 1336 D020  SLBAVL MOVB  @HEXCTR,R0       Reset REN bit & enable intrpt.
      1338 5FFC
U0046 133A 5020         SZCB  @REN,R0
      133C 001C'
U0047 133E F020         SOCB  @BAVAIE,R0
      1340 0016'
U0048 1342 D800  SLOPEX MOVB  R0,@HEXCTW       Update control byte
      1344 5FFA
U0049 1346 0460         B     @EXIT1           Leave until a new message sent
      1348 0B4E'
U0050                    *
U0051                    ************************************************************
U0052                    *
U0053                    *       SLAVE CLOSE ROUTINE
U0054                    *       This routine cuts off communication from the bus.
U0055                    *
U0056 134A D020  SLVCLS MOVB  @HEXCTR,R0       Disable communication
      134C 5FFC
U0057 134E 5020         SZCB  @DISABL,R0
      1350 0012'
U0058 1352 10F7         JMP   SLOPEX
U0059                    *
U0060                    ************************************************************
```

```
U0062                ***************************************************************
U0063                *
U0064                *       SLAVE MODE BYTE OUTPUT ROUTINE
U0065                *       Tests for BAV high, in which case an error is re-
U0066                *       turned.  This routine writes a byte from R0 to the
U0067                *       hexbus.
U0068                *
U0069                *       Registers Destroyed:  R1,R2
U0070                *
U0071                *       Calls Subroutines  :  none
U0072                *
U0073 1354 D060      SYOUT   MOVB @HEXSTS,R1         R1 = hexbus status
      1356 5FFA
U0074 1358 D081              MOVB R1,R2             R2 = copy of current status
U0075 135A 0241              ANDI R1,SBAV           Is BAV high?
      135C 0800
U0076 135E 1602              JNE  SYOUT1            If not, continue
U0077 1360 0460      SYOUTE  B    @DEVERR           If yes, error
      1362 0AF6'
U0078 1364 0242      SYOUT1  ANDI R2,WBUSY          Is WBUSY true?
      1366 0400
U0079 1368 130A              JEQ  SYOUT3            If not, go to write byte
U0080 136A D060      SYOUT2  MOVB @HEXSTS,R1        If yes, R1 = new status
      136C 5FFA
U0081 136E D081              MOVB R1,R2             R2 = copy of current status
U0082 1370 0241              ANDI R1,HSKWT          Is HSKWT true?
      1372 8000
U0083 1374 1604              JNE  SYOUT3            If yes, go to write byte
U0084 1376 0242              ANDI R2,SBAV           If not, is BAV still low?
      1378 0800
U0085 137A 16F7              JNE  SYOUT2            If yes, continue waiting
U0086 137C 10F1              JMP  SYOUTE            If not, error
U0087 137E D800      SYOUT3  MOVB R0,@HEXMTW        Write out byte
      1380 5FF8
U0088 1382 045B              RT
U0089                *
U0090                ***************************************************************
```

```
U0092                ***************************************************************
U0093                *
U0094                *       SLAVE MODE BYTE INPUT ROUTINE
U0095                *       This routine reads a byte from the hexbus into R0.
U0096                *       HSK SHOULD BE left low on exit (contains code to
U0097                *       hold HSK low, on last byte read, to work with
U0098                *       future hardware.
U0099                *
U0100                *       Upon entry: R5 = BAV high error exit
U0101                *
U0102                *       Upon exit:  R0 = byte read from bus
U0103                *
U0104                *       Registers Destroyed:  R0,R2,R3
U0105                *
U0106                *       Calls Subroutines  :  none
U0107                *
U0108 1384 D0A0  SYTIN  MOVB @HEXSTS,R2         R2 = hexbus status
       1386 5FFA
U0109 1388 D0C2         MOVB R2,R3              R3 = copy of current status
U0110 138A 0242         ANDI R2,HSKRD           Is HSKRD true?
       138C 4000
U0111 138E 1618         JNE  SYTIN4             If yes, read the byte
U0112 1390 D083         MOVB R3,R2              R2 = copy of current status
U0113 1392 0242         ANDI R2,RBUSY           Is RBUSY true?
       1394 0200
U0114 1396 1604         JNE  SYTIN2             If yes, wait for byte
U0115 1398 0243         ANDI R3,SBAV            If not, is BAV low?
       139A 0800
U0116 139C 16F3         JNE  SYTIN              If yes, wait for byte
U0117 139E 1009         JMP  RESREN             If not, go reset REN bit & exit
U0118 13A0 D0A0  SYTIN2 MOVB @HEXSTS,R2         R2 = hexbus status
       13A2 5FFA
U0119 13A4 D0C2         MOVB R2,R3              R3 = copy of current status
U0120 13A6 0242         ANDI R2,HSKRD           Is HSKRD true?
       13A8 4000
U0121 13AA 160A         JNE  SYTIN4             If yes, go to read byte
U0122 13AC 0243         ANDI R3,SBAV            If not, is BAV still low?
       13AE 0800
U0123 13B0 16F7         JNE  SYTIN2             If yes, continue waiting
U0124 13B2 D0E0  RESREN MOVB @HEXCTR,R3         Reset REN bit
       13B4 5FFC
U0125 13B6 50E0         SZCB @REN,R3
       13B8 001C'
U0126 13BA D803         MOVB R3,@HEXCTW
       13BC 5FFA
U0127 13BE 0455         B    *R5                ERROR EXIT ON BAV HIGH
U0128 13C0 D020  SYTIN4 MOVB @SLAST,R0          Treat this as last byte?
       13C2 E073
U0129 13C4 1306         JEQ  SYTIN5             If not, just read it
U0130 13C6 D020         MOVB @HEXCTR,R0         If yes, get control byte
       13C8 5FFC
U0131 13CA F020         SOCB @CR7,R0            HOLD HSK LOW here!
       13CC 001E'
U0132 13CE D800         MOVB R0,@HEXCTW         Rewrite control byte
       13D0 5FFA
U0133 13D2 D020  SYTIN5 MOVB @HEXRDD,R0         Read a byte from the bus
       13D4 5FF8
U0134 13D6 045B         RT                      Return to caller
U0135                *
U0136                ***************************************************************
```

```
U0138                ***************************************************************
U0139                *
U0140                *       SLAVE READ ROUTINE
U0141                *
U0142                *       Monitors bus for its device code (R8).
U0143                *
U0144  13D8 C1A4     SLVRD   MOV   @BUFADR(R4),R6    R6 = address of data buffer
       13DA 004C
U0145  13DC C1E4             MOV   @LRECLN(R4),R7    R7 = size of buffer
       13DE 0052
U0146  13E0 D224             MOVB  @DEVCOD(R4),R8    R8 = device code (MSByte)
       13E2 005D
U0147  13E4 0205             LI    R5,DEVERR         R5 = BAV high exit
       13E6 0AF6'
U0148  13E8 C046             MOV   R6,R1             R1 allows access to buffer
U0149  13EA 06A0             BL    @SUWRIT           Set up for write
       13EC 0A72'
U0150  13EE 04C6             CLR   R6                R6 = total # of bytes recv.
U0151  13F0 D020             MOVB  @HEXCTR,R0        Get control byte
       13F2 5FFC
U0152  13F4 5020             SZCB  @BAVAIE,R0        Disable BAV rising interrupt
       13F6 0016'
U0153  13F8 5020             SZCB  @WEN,R0           Disable chip for writing
       13FA 001A'
U0154  13FC D800             MOVB  R0,@HEXCTW        Rewrite control byte
       13FE 5FFA
U0155  1400 2020             COC   @H0200,R0         Is REN bit already set?
       1402 0EE8'
U0156  1404 1301             JEQ   SLVRD1            If yes, continue
U0157  1406 100F             JMP   SLRD1             If not, wait for BAV high
U0158  1408 D0A0     SLVRD1  MOVB  @HEXSTS,R2        R2 = hexbus status byte
       140A 5FFA
U0159  140C 0242     SLVRD0  ANDI  R2,HSKRD          Is HSKRD true? (MSG STARTED?)
       140E 4000
U0160  1410 13FB             JEQ   SLVRD1            If not, keep checking for start
U0161  1412 06A0             BL    @SYTIN4           Read in FIRST BYTE of message
       1414 13C0'
U0162  1416 9200             CB    R0,R8             Is is the proper device code?
U0163  1418 1312             JEQ   SLVRD2            If yes, continue
U0164  141A D020             MOVB  @HEXCTR,R0        Get control byte
       141C 5FFC
U0165  141E 5020             SZCB  @REN,R0           Reset REN to ignore HSKs
       1420 001C'
U0166  1422 D800             MOVB  R0,@HEXCTW        Rewrite control byte
       1424 5FFA
U0167  1426 D0A0     SLRD1   MOVB  @HEXSTS,R2        R2 = hexbus status
       1428 5FFA
U0168  142A 0242             ANDI  R2,SBAV           Is BAV high?
       142C 0800
U0169  142E 16FB             JNE   SLRD1             Loop until BAV goes high
U0170  1430 D020             MOVB  @HEXCTR,R0        Get control byte
       1432 5FFC
U0171  1434 F020             SOCB  @REN,R0           Set REN to latch HSKs
       1436 001C'
U0172  1438 D800             MOVB  R0,@HEXCTW        Rewrite control byte
       143A 5FFA
U0173  143C 10E7             JMP   SLVRD0            Continue looking for new messg.
U0174  143E 06A0     SLVRD2  BL    @WTBYTO           Write byte from R0 TO RAM
       1440 0A8E'
U0175  1442 06C8             SWPB  R8                Device code = LSByte of R8
```

```
U0176 1444 D200          MOVB  R0,R8           Save received device code in R8
U0177 1446 0586          INC   R6              Increment bytes received
U0178 1448 0286          CI    R6,8            Have we received 8 bytes of PAB?
      144A 0008
U0179 144C 1303          JEQ   SLVRD3          If yes, PAB received
U0180 144E 06A0          BL    @SYTIN          If not, read another byte
      1450 1384'
U0181 1452 10F5          JMP   SLVRD2          Continue until PAB received
U0182 1454 06A0  SLVRD3  BL    @HLDHSK         Hold HSK low (this may be it)
      1456 14E8'
U0183 1458 06A0          BL    @SYTIN          Read MSByte of data length
      145A 1384'
U0184 145C 06A0          BL    @WTBYTO         Write to buffer in RAM
      145E 0A8E'
U0185 1460 06C8          SWPB  R8              Set up R8 to recv. data
U0186 1462 D200          MOVB  R0,R8           R8 now = data length
U0187 1464 C188          MOV   R8,R6           R6 = data length
U0188 1466 0228          AI    R8,9            R8 = total characters recvd.
      1468 0009
U0189 146A 81C8          C     R8,R7           Too much data?
U0190 146C 1B15          JH    SLVRD7          If yes, error
U0191 146E C186          MOV   R6,R6           If not, any data to recv.?
U0192 1470 130F          JEQ   SLVRD6          If not, exit
U0193 1472 0606          DEC   R6              Set to recv. all but last byte
U0194 1474 06A0          BL    @RELHSK         Release HSK to get data
      1476 14F0'
U0195 1478 06A0  SLVRD4  BL    @SYTIN          Read a byte of data
      147A 1384'
U0196 147C 06A0          BL    @WTBYTO         Write byte from R0
      147E 0A8E'
U0197 1480 0606          DEC   R6              One less byte of data to read
U0198 1482 16FA          JNE   SLVRD4          Continue reading data
U0199 1484 06A0          BL    @HLDHSK         Set to hold HSK low after last
      1486 14E8'
U0200 1488 06A0          BL    @SYTIN          Read last byte of data
      148A 1384'
U0201 148C 06A0          BL    @WTBYTO         Sotre byte in RAM buffer
      148E 0A8E'
U0202 1490 C908  SLVRD6  MOV   R8,@CHRCNT(R4)  Return character count in PAB
      1492 004E
U0203 1494 0460          B     @EXIT1          Exit DSR (HSK is low)
      1496 0B4E'
U0204 1498 06A0  SLVRD7  BL    @RELHSK         Error, release HSK
      149A 14F0'
U0205 149C 0460          B     @SIZERR         ERROR EXIT
      149E 0AEA'
U0206                    *
U0207                    ***************************************************************
```

```
U0209                ***************************************************************
U0210                *
U0211                *       SLAVE WRITE ROUTINE
U0212                *
U0213                *       Inclused code to allow use of slave mode on current
U0214                *       OSO chip, by having the master send an extra byte
U0215                *       of data over the bus.  This automatically hold HSK
U0216                *       low, until the slave is ready to respond.  This
U0217                *       routine then reads the OSO data register, which
U0218                *       releases HSK, just before the slave responds.  Thus,
U0219                *       The master will have to know to send the extra byte.
U0220                *
U0221                *       When OSO is redesigned, CR7 in the control register
U0222                *       should allow manual control of HSK.  This bit is
U0223                *       therefore set (HSK low) just before the last byte
U0224                *       from the master is read.  Effectively HSK is held
U0225                *       low until the CR7 bit is cleared, at which point
U0226                *       HSK is released and the response is sent back.
U0227                *
U0228 14A0 D020  SLVWR  MOVB @HEXCTR,RO        Get control byte
      14A2 5FFC
U0229 14A4 F020         SOCB @WEN,RO           Set WEN bit
      14A6 001A'
U0230 14A8 D800         MOVB RO,@HEXCTW        Rewrtie control byte
      14AA 5FFA
U0231 14AC C1A4         MOV  @BUFADR(R4),R6    R6 = data buffer address
      14AE 004C
U0232 14B0 C1E4         MOV  @CHRCNT(R4),R7    R7 = character count to be sent
      14B2 004E
U0233 14B4 D060         MOVB @HEXSTS,R1        R1 = hexbus status
      14B6 5FFA
U0234 14B8 D0C1         MOVB R1,R3             R3 = copy of current status
U0235 14BA 0241         ANDI R1,SHSK           Is HSK high?
      14BC 0100
U0236 14BE 1312         JEQ  SLVWR2            If yes, error
U0237 14C0 0243         ANDI R3,SBAV           Is BAV high?
      14C2 0800
U0238 14C4 130F         JEQ  SLVWR2            If yes, error
U0239 14C6 C046         MOV  R6,R1             Allow access to data buffer
U0240 14C8 06A0         BL   @SUREAD           Set up for read from RAM
      14CA 0A56'
U0241 14CC D020         MOVB @HEXRDD,RO        FAKE SLAVE MODE ABILITY
      14CE 5FF8
U0242 14D0 06A0         BL   @RELHSK           Release HSK before responding
      14D2 14F0'
U0243 14D4 06A0  SLVWR1 BL   @RDBYTO           Read byte from RAM into RO
      14D6 0A16'
U0244 14D8 06A0         BL   @SYOUT            Write a byte to the hexbus
      14DA 1354'
U0245 14DC 0607         DEC  R7                One less byte to write
U0246 14DE 16FA         JNE  SLVWR1            Continue writing to hexbus
U0247 14E0 0460  SLVWRX B    @EXIT1            Exit DSR until next message
      14E2 0B4E'
U0248 14E4 0460  SLVWR2 B    @DEVERR           ERROR EXIT
      14E6 0AF6'
U0249                *
U0250                ***************************************************************
```

```
U0252                ***************************************************************
U0253                *
U0254                *       ROUTINES FOR CONTROLLING SLAVE HSK LINE
U0255                *
U0256                *       To control future hardware release of OSO chip
U0257                *
U0258  14E8 0700     HLDHSK SETO RO                  Set flag to hold HSK low
U0259  14EA D800             MOVB RO,@SLAST
       14EC E073
U0260  14EE 045B             RT
U0261  14F0 04C0     RELHSK CLR  RO                  Set flag for auto HSK
U0262  14F2 D800             MOVB RO,@SLAST
       14F4 E073
U0263  14F6 D020             MOVB @HEXCTR,RO         Release HSK line
       14F8 5FFC
U0264  14FA 5020             SZCB @CR7,RO
       14FC 001E'
U0265  14FE D800             MOVB RO,@HEXCTW
       1500 5FFA
U0266  1502 045B             RT
U0267                *
U0268                ***************************************************************
```

```
 0022                    COPY DLU09.KSG.HEXBUS.DILLO.DISKRT
V0001                ***************************************************************
V0002                *
V0003                *       TABLE OF LUNO/FILE NAME BLOCKS
V0004                *
V0005  1504 E012     LUNTBL DATA LUN250            LUNO 250 / BLOCK 1
V0006  1506 FA00            DATA >FA00
V0007  1508 E029            DATA LUN251           LUNO 251 / BLOCK 2
V0008  150A FB00            DATA >FB00
V0009  150C E040            DATA LUN252           LUNO 252 / BLOCK 3
V0010  150E FC00            DATA >FC00
V0011  1510 E057            DATA LUN253           LUNO 253 / BLOCK 4
V0012  1512 FD00            DATA >FD00
V0013  1514 0000            DATA >0000            End of table markers
V0014                *
V0015                ***************************************************************
V0016                *
V0017                *       DISK MANAGER ERROR TABLE
V0018                *
V0019  1516   50     DMERR  BYTE >50              HFDS error >60 : DMGR >50      1
V0020  1517   06            BYTE >06              HFDS error >61 : DMGR >06      1
V0021  1518   07            BYTE >07              HFDS error >62 : DMGR >07      1
V0022  1519   11            BYTE >11              HFDS error >63 : DMGR >11      1
V0023  151A   21            BYTE >21              HFDS error >64 : DMGR >21[41]  1
V0024  151B   21            BYTE >21              HFDS error >65 : DMGR >21[51]  1
V0025  151C   21            BYTE >21              HFDS error >66 : DMGR >21      1
V0026  151D   22            BYTE >22              HFDS error >67 : DMGR >22      1
V0027  151E   23            BYTE >23              HFDS error >68 : DMGR >23      1
V0028  151F   28            BYTE >28              HFDS error >69 : DMGR >28      1
V0029  1520   31            BYTE >31              HFDS error >6A : DMGR >31      1
V0030  1521   33            BYTE >33              HFDS error >6B : DMGR >33      1
V0031  1522   35            BYTE >35              HFDS error >6C : DMGR >35 [U]  1
V0032  1523   35            BYTE >35              HFDS error >6D : DMGR >35 [U]  1
V0033  1524   35            BYTE >35              HFDS error >6E : DMGR >35 [U]  1
V0034  1525   35            BYTE >35              HFDS error >6F : DMGR >35 [U]  1
V0035  1526                 EVEN
V0036                *
V0037                ***************************************************************
```

```
V0039                    ************************************************************
V0040                    *
V0041                    *       INITIALIZATION CODE FOR DISK SUBROUTINES
V0042                    *
V0043 1526 02A4  ENTER   STWP  R4                  Get workspace pointer into R4
V0044 1528 0224          AI    R4,->E0             Set R4 to beginning of RAM
      152A FF20
V0045 152C 2D43          XOP   3,5                 Map in additional RAM
V0046 152E 1602          JNE   ENTER1              Mapper error?
V0047 1530 0460          B     @LOSCOM             If yes, error
      1532 18A8'
V0048 1534 D024  ENTER1  MOVB  @>4C(R4),R0         Get disk unit number in R0
      1536 004C
V0049 1538 D040          MOVB  R0,R1               R1 = copy of >4C
V0050 153A D080          MOVB  R0,R2               R2 = copy of >4C
V0051 153C 0240          ANDI  R0,>0F00            Mask out only unit number in R0
      153E 0F00
V0052 1540 0220          AI    R0,100*256          Convert to hexbus device code
      1542 6400
V0053 1544 D900          MOVB  R0,@DEVCOD(R4)      Store device code in PAB
      1546 005D
V0054 1548 020A          LI    R10,>0400           Set 'SUBR' entry flag
      154A 0400
V0055 154C 0241          ANDI  R1,>4000            Is CPU RAM flag set?
      154E 4000
V0056 1550 1302          JEQ   ENTERV              If not, continue
V0057 1552 026A          ORI   R10,>0800           If yes, set CPU RAM select
      1554 0800
V0058 1556 0242  ENTERV  ANDI  R2,>3000            Mask out only DSR version #
      1558 3000
V0059 155A 0942          SRL   R2,4                MSByte R2 = DSR version #
V0060 155C D802          MOVB  R2,@DSKLUN+1        Store the DSR version # in RAM
      155E E071
V0061 1560 04E4          CLR   @LUNO(R4)           Set ALC PAB to zero
      1562 005F
V0062 1564 04E4          CLR   @RECNO(R4)
      1566 0060
V0063 1568 04E4          CLR   @BUFLEN(R4)
      156A 0062
V0064 156C 04E4          CLR   @WDATLN(R4)
      156E 0064
V0065 1570 C807          MOV   R7,@LEVEL1          Save main return address
      1572 E008
V0066 1574 045B          RT                        Return to caller
V0067                    *
V0068                    ************************************************************
```

```
V0070                ***************************************************************
V0071                *
V0072                *       COMPARE 2 FILE DESCRIPTORS FOR DISK
V0073                *
V0074                *       Upon Entry:   R6 points to the file name
V0075                *                     R7 is the length of the file name
V0076                *                     R3 points to file name in LUNO table
V0077                *
V0078                *       Upon Exit:    EQ status bit set if there is a match
V0079                *                     EQ status bit reset if no match found
V0080                *
V0081                *       Registers Destroyed:  R0,R1,R3,R5
V0082                *
V0083                *       Subroutines Called:   SUREAD, RDBYT1
V0084                *
V0085 1576 C14B  COMPAR MOV   R11,R5             Save return address in R0
V0086 1578 9933         CB    *R3+,@DEVCOD(R4) Is it the same device?
      157A 005D
V0087 157C 1610         JNE   COMPRX           If not, no match
V0088 157E C007         MOV   R7,R0            R0 = file name length
V0089 1580 06C0         SWPB  R0               Get length in MSByte
V0090 1582 9033         CB    *R3+,R0          Are names the same length?
V0091 1584 160C         JNE   COMPRX           If not, no match
V0092 1586 06C0         SWPB  R0               Restore R0
V0093 1588 C000         MOV   R0,R0            Is name length zero?
V0094 158A 1309         JEQ   COMPRX           Treated as a match
V0095 158C C046         MOV   R6,R1            R1 points to file name in PAB
V0096 158E 06A0         BL    @SUREAD          Set up for read of file name
      1590 0A56'
V0097 1592 06A0  COMPRL BL    @RDBYT1          Read a byte from RAM
      1594 0A26'
V0098 1596 9073         CB    *R3+,R1          Compare with stored name byte
V0099 1598 1602         JNE   COMPRX           If different, no match
V0100 159A 0600         DEC   R0               One less byte to compare
V0101 159C 16FA         JNE   COMPRL           Continue until all byte comp.
V0102 159E 0455  COMPRX B     *R5              Return to calling routine
V0103                *
V0104                ***************************************************************
```

```
V0106                ***************************************************************
V0107                *
V0108                *       CHECK DISK FILE DESCRIPTOR ROUTINE
V0109                *
V0110                *       Upon entry:  R6 points to the file name
V0111                *                    R7 is the length of the file name
V0112                *
V0113                *       Upon exit:   Exit 1 => Not a disk device
V0114                *                    Exit 2 => File name match found
V0115                *                    Exit 3 => No match, no space available
V0116                *                    Exit 4 => No match, space is available
V0117                *
V0118                *       Registers Destroyed:   R0,R2,R3,R8
V0119                *
V0120                *       Subroutines Called:    COMPAR
V0121                *
V0122 15A0 C20B  CHKDSK MOV  R11,R8              Save return address in R8
V0123 15A2 22A0         COC  @H0200,R10          Is this a DSK operation?
      15A4 0EE8'
V0124 15A6 1625         JNE  CHKDSX              **** EXIT 1 ****
V0125 15A8 05C8         INCT R8                  Prepare for next exit
V0126 15AA 04E0         CLR  @BLOKAV             Set Block Available to zero
      15AC E06E
V0127 15AE 0202         LI   R2,LUNTBL           R2 points to LUNO table
      15B0 1504'
V0128 15B2 C0F2  CHKDS4 MOV  *R2+,R3             R3 points to specific block
V0129 15B4 1310         JEQ  CHKDSC              Jump on end of table
V0130 15B6 04C0         CLR  R0                  R0 = 0
V0131 15B8 D013         MOVB *R3,R0              Is 1st byte of block zero?
V0132 15BA 1305         JEQ  CHKDS5              If yes, set block available
V0133 15BC 06A0         BL   @COMPAR             If not, compare file names
      15BE 1576'
V0134 15C0 1316         JEQ  CHKDSZ              **** EXIT 2 ****
V0135 15C2 05C2  CHKDS6 INCT R2                  Point to next block
V0136 15C4 10F6         JMP  CHKDS4              Continue block search
V0137 15C6 C020  CHKDS5 MOV  @BLOKAV,R0          Is BLOKAV already set?
      15C8 E06E
V0138 15CA 16FB         JNE  CHKDS6              If yes, continue search
V0139 15CC C803         MOV  R3,@BLOKAV          If not, set BLOKAV to address
      15CE E06E
V0140 15D0 C832         MOV  *R2+,@DSKLUN        Set DSKLUN to proper LUNO
      15D2 E070
V0141 15D4 10EE         JMP  CHKDS4              Continue block search
V0142 15D6 05C8  CHKDSC INCT R8                  Prepare for next exit (3)
V0143 15D8 C020         MOV  @BLOKAV,R0          No block available?
      15DA E06E
V0144 15DC 130A         JEQ  CHKDSX              **** EXIT 3 ****
V0145 15DE 05C8         INCT R8                  Prepare for next exit (4)
V0146 15E0 D024         MOVB @OPCODE(R4),R0      Not an OPEN operation?
      15E2 004A
V0147 15E4 1606         JNE  CHKDSX              **** EXIT 4 ****
V0148 15E6 D920         MOVB @DSKLUN,@LUNO(R4) Pass LUNO to PAB (OPEN)
      15E8 E070
      15EA 005F
V0149 15EC 1002         JMP  CHKDSX              **** EXIT 4 ****
V0150 15EE D912  CHKDSZ MOVB *R2,@LUNO(R4)       Pass LUNO to PAB (non-OPEN)
      15F0 005F
V0151 15F2 C218  CHKDSX MOV  *R8,R8              Get appropriate return address
V0152 15F4 0458         B    *R8                 Return to caller
V0153                *
```

V0154 **********

```
V0156                ***************************************************************
V0157                *
V0158                *       SAVE FILE NAME IN LUNO BLOCK
V0159                *
V0160                *       Upon entry:     BLOKAV points to the LUNO block
V0161                *                       R6 points to the file name
V0162                *                       R7 is the length of the file name
V0163                *
V0164                *       Upon exit:      File name is copied into LUNO block
V0165                *
V0166                *       Registers Destroyed:    R1,R3,R5,R8
V0167                *
V0168                *       Subroutines Called:     SUREAD, RDBYT1
V0169                *
V0170 15F6 C20B   SAV$FN MOV   R11,R8            Save return address
V0171 15F8 C0E0          MOV   @BLOKAV,R3        R3 = address of dest. block
      15FA E06E
V0172 15FC 0583          INC   R3                Point to block f.n. length
V0173 15FE C147          MOV   R7,R5             R5 = length of file name
V0174 1600 06C5          SWPB  R5                Put length in MSByte of R5
V0175 1602 DCC5          MOVB  R5,*R3+           Second byte of block = length
V0176 1604 1309          JEQ   SAV$FX            If length = 0 then exit
V0177 1606 06C5          SWPB  R5                Make R5 a word again (length)
V0178 1608 C046          MOV   R6,R1             R1 points to name in RAM
V0179 160A 06A0          BL    @SUREAD           Set up for read from RAM
      160C 0A56'
V0180 160E 06A0   SAV$LP BL    @RDBYT1           Read a file name byte into R1
      1610 0A26'
V0181 1612 DCC1          MOVB  R1,*R3+           Copy byte into block
V0182 1614 0605          DEC   R5                One less byte to copy
V0183 1616 16FB          JNE   SAV$LP            Continue copying file name
V0184 1618 0458   SAV$FX B     *R8               Return to caller
V0185                *
V0186                ***************************************************************
```

```
V0188                ***************************************************************
V0189                *
V0190                *       COMMON FILE NAME SEARCH ROUTINE
V0191                *
V0192                *       Registers Destroyed:   none
V0193                *
V0194                *       Subroutines Called:    CHKDSK
V0195                *
V0196 161A C80B   COMDSK MOV   R11,@LEVEL2       Store return address
      161C E00A
V0197 161E 06A0          BL    @CHKDSK           Call check disk routine
      1620 15A0'
V0198 1622 162A'         DATA  COMDSX            1) Not a disk device
V0199 1624 162A'         DATA  COMDSX            2) File name match
V0200 1626 0AD8'         DATA  FILERR            3) File name not found
V0201 1628 0AD8'         DATA  FILERR            4) File name not found
V0202 162A C2E0   COMDSX MOV   @LEVEL2,R11       Restore return address
      162C E00A
V0203 162E 045B          RT
V0204                *
V0205                ***************************************************************
```

```
V0207                ****************************************************************
V0208                *
V0209                *       COMMON POST-PROCESSING ROUTINE
V0210                *       This routine sets the LUNO block device code on a
V0211                *       successful DSK OPEN, and resets the code byte on a
V0212                *       successful DSK CLOSE.
V0213                *
V0214                *       Registers Destroyed:   R0,R1,R2
V0215                *
V0216                *       Subroutines Called:    none
V0217                *
V0218 1630 D0A4      DSKXOP MOVB @DEVCOD(R4),R2    R2 = device code at OPEN
      1632 005D
V0219 1634 1002             JMP  DSKEXT            Continue
V0220 1636 04C2      DSKXCD CLR  R2                R2 = block available byte
V0221 1638 1003             JMP  DSKCL1            Close regardless of error     1
V0222 163A D024      DSKEXT MOVB @COMSTS(R4),R0    Did an error occur?
      163C 006A
V0223 163E 160D             JNE  DSKEXX            If yes, exit
V0224 1640 22A0      DSKCL1 COC  @H0200,R10        Is this a DSK operation?      1
      1642 0EE8'
V0225 1644 160A             JNE  DSKEXX            If not, exit
V0226 1646 D024             MOVB @LUNO(R4),R0      R0 = current LUNO of file
      1648 005F
V0227 164A 0980             SRL  R0,8              Make it a word
V0228 164C 0220             AI   R0,-250           Get LS digit of LUNO
      164E FF06
V0229 1650 0A20             SLA  R0,2              Multiply by 4 for index
V0230 1652 0220             AI   R0,LUNTBL         Index into the table
      1654 1504'
V0231 1656 C050             MOV  *R0,R1            R0 = address of LUNO block
V0232 1658 D442             MOVB R2,*R1            Update block available byte
V0233 165A 045B      DSKEXX RT                     Return to caller
V0234                *
V0235                ****************************************************************
```

```
V0237                ***************************************************************
V0238                *
V0239                *       TRANSFER BYTES FROM VDP RAM TO BUS
V0240                *
V0241                *       Upon entry:     R1 points to VDP RAM buffer
V0242                *                       R5 points to error exit routine
V0243                *                       R6 = number of bytes to be sent
V0244                *
V0245                *       Registers Destroyed:    R0,R6,R7
V0246                *
V0247                *       Subroutines Called:     SUREAD, BYOUT
V0248                *
V0249 165C C1CB  XVTH    MOV   R11,R7              Store return address
V0250 165E 06A0          BL    @SUREAD           Set up for RAM read
      1660 0A56'
V0251 1662 06A0  XVTHLP  BL    @RDBYTO           Read byte from RAM into R0
      1664 0A16'
V0252 1666 06A0          BL    @BYOUT            Write byte to hexbus
      1668 0DFE'
V0253 166A 0606          DEC   R6                One less byte to write
V0254 166C 16FA          JNE   XVTHLP            Continue sending bytes
V0255 166E 0457          B     *R7               Return to caller
V0256                *
V0257                ***************************************************************
```

```
V0259                ***************************************************************
V0260                *
V0261                *       READ A WORD FROM THE HEXBUS
V0262                *
V0263 1670 C18B   WRDIN  MOV   R11,R6            Save return address
V0264 1672 06A0          BL    @BYTIN            Read LSByte of word
      1674 0E18'
V0265 1676 06C0          SWPB  R0                Make it the LSByte
V0266 1678 06A0          BL    @BYTIN            Read MSByte of word
      167A 0E18'
V0267 167C 0456          B     *R6               Return to caller
V0268                *
V0269                ***************************************************************
V0270                *
V0271                *       WRITE A WORD TO THE HEXBUS
V0272                *
V0273 167E C80B   WRDOUT MOV   R11,@LEVEL3       Save return address
      1680 E00C
V0274 1682 06C0          SWPB  R0                Put LSByte first
V0275 1684 06A0          BL    @BYOUT            Send LSByte of R0
      1686 0DFE'
V0276 1688 06C0          SWPB  R0                Put MSByte first
V0277 168A 06A0          BL    @BYOUT            Send MSByte of R0
      168C 0DFE'
V0278 168E C2E0          MOV   @LEVEL3,R11       Restore return address
      1690 E00C
V0279 1692 045B          RT                      Return to caller
V0280                *
V0281                ***************************************************************
```

```
V0283                ***************************************************************
V0284                *
V0285                *       COMMON DINP/DOUTP ROUTINE
V0286                *
V0287 1694 C80B  S14$15 MOV  R11,@LEVEL2       Store return address
      1696 E00A
V0288 1698 C1E4         MOV  @>4C(R4),R7       R7 = unit # and access code
      169A 004C
V0289 169C 0247         ANDI R7,>0FFF          Mask off any other bits
      169E 0FFF
V0290 16A0 06CA         SWPB R10               Put unit # in LSByte of R10
V0291 16A2 D287         MOVB R7,R10
V0292 16A4 06CA         SWPB R10
V0293 16A6 022A         AI   R10,3             Convert unit number to code
      16A8 0003
V0294 16AA 0208         LI   R8,>15            Set command to >15 (DINP)
      16AC 0015
V0295 16AE 028B         CI   R11,DINPRT        Called by DINP?
      16B0 199E'
V0296 16B2 1302         JEQ  S$1               If yes, continue
V0297 16B4 0228         AI   R8,>0001          If not, command = >16 (DOUTP)
      16B6 0001
V0298 16B8 0205  S$1    LI   R5,4*256          MSByte of R5 = 4
      16BA 0400
V0299 16BC D905         MOVB R5,@WDATLN+1(R4)  Set data length to 4
      16BE 0065
V0300 16C0 C0A4         MOV  @>4E(R4),R2       R2 points to file name
      16C2 004E
V0301 16C4 0200         LI   R0,10             R0 = file name length
      16C6 000A
V0302 16C8 0A87         SLA  R7,8              Refers to FDR?
V0303 16CA 130E         JEQ  FDR$1             If yes, process FDR
V0304 16CC 05C8         INCT R8                RW records: command= >17 or >18
V0305 16CE A900         A    R0,@WDATLN(R4)    Data length = 4 + FN length
      16D0 0064
V0306 16D2 028B         CI   R11,DINPRT        Called by DINP?
      16D4 199E'
V0307 16D6 1603         JNE  REC$1             If not, go modify params.
V0308 16D8 D907         MOVB R7,@BUFLEN(R4)    If yes, buf.len.=256*# sectors
      16DA 0062
V0309 16DC 1013         JMP  S$2               Continue
V0310 16DE D907  REC$1  MOVB R7,@WDATLN(R4)    Buf.len. = 256 * # sectors + 4
      16E0 0064
V0311 16E2 05A4         INC  @WDATLN(R4)       Buf.len. = 256 * # sectors + 5
      16E4 0064
V0312 16E6 100E         JMP  S$2               Continue
V0313 16E8 05E0  FDR$1  INCT @LEVEL2           Bump return address to 2nd
      16EA E00A
V0314 16EC 0205         LI   R5,28*256         MSByte of R5 = 28
      16EE 1C00
V0315 16F0 028B         CI   R11,DINPRT        Called by DINP?
      16F2 199E'
V0316 16F4 1605         JNE  FDR$2             If not, process as output
V0317 16F6 C900         MOV  R0,@WDATLN(R4)    Set data length to 10
      16F8 0064
V0318 16FA D905         MOVB R5,@BUFLEN+1(R4)  Set buffer length to 28
      16FC 0063
V0319 16FE 1002         JMP  S$2               Continue
V0320 1700 D905  FDR$2  MOVB R5,@WDATLN+1(R4)  Set data length to 28
      1702 0065
```

```
V0321 1704 06C8  S$2    SWPB R8                 Put command in MSByte of R8
V0322 1706 C908         MOV  R8,@COMMAD(R4)     Set command code and luno=0
      1708 005E
V0323 170A 06A0         BL   @MXMIT             Send PAB over hexbus
      170C 0BCA'
V0324 170E 0460         B    @LOSCOM            ERROR EXIT
      1710 18A8'
V0325 1712 C224         MOV  @>50(R4),R8        Get pointer to info block
      1714 0050
V0326 1716 D024         MOVB @>4C(R4),R0        Is this an extended operation?
      1718 004C
V0327 171A 1102         JLT  S$4                If yes, pointer is ok (VDP)
V0328 171C 0988         SRL  R8,8               If not, adjust for CPU offset
V0329 171E A204         A    R4,R8              Convert to real RAM address
V0330 1720 C1C7  S$4    MOV  R7,R7              Dealing with FDR?
V0331 1722 130B         JEQ  S$3                If yes, skip parameter xmit
V0332 1724 C028         MOV  @2(R8),R0          R0 = starting AU #
      1726 0002
V0333 1728 06A0         BL   @WRDOUT            Write starting AU # to bus
      172A 167E'
V0334 172C C007         MOV  R7,R0              Get number of sectors in R0
V0335 172E 06C0         SWPB R0                 Make it a word
V0336 1730 06A0         BL   @WRDOUT            Write number of AUs to the bus
      1732 167E'
V0337 1734 6920         S    @H0004,@WDATLN(R4) Four less bytes to write
      1736 0EEC'
      1738 0064
V0338 173A 0206  S$3    LI   R6,10              R6 = file name length
      173C 000A
V0339 173E C064         MOV  @>4E(R4),R1        R1 = pointer to file name
      1740 004E
V0340 1742 06A0         BL   @XVTH              Send the file name to the bus
      1744 165C'
V0341 1746 6920         S    @D10,@WDATLN(R4)   Ten less bytes to send
      1748 0EF0'
      174A 0064
V0342 174C C2E0         MOV  @LEVEL2,R11        Restore updated return address
      174E E00A
V0343 1750 045B         RT                      Return to caller
V0344              *
V0345              ********************************************************************
```

```
 0023                    COPY DLU09.KSG.HEXBUS.DILLO.DISKIO
W0001                ***************************************************
W0002                *
W0003                *       SUBROUTINE 10:  READ & WRITE SECTORS
W0004                *
W0005  1752 C1CB     RWSECT MOV  R11,R7             Store return address
W0006  1754 06A0            BL   @ENTER             Initialize
       1756 1526'
W0007  1758 C1A4            MOV  @>4C(R4),R6        R6 = Unit/RW flag
       175A 004C
W0008  175C 06C6            SWPB R6                 Put RW flag in MSByte
W0009  175E D920            MOVB @WSECOP,@COMMAD(R4) Set command code to wrt.sec.
       1760 0ED4'
       1762 005E
W0010  1764 020B            LI   R11,>0104          Set data length to >0104
       1766 0104
W0011  1768 C90B            MOV  R11,@WDATLN(R4)
       176A 0064
W0012  176C D186            MOVB R6,R6              Is this a READ operation?
W0013  176E 1306            JEQ  SKP$RS             If not, skip read init.
W0014  1770 790B            SB   R11,@COMMAD(R4)    Change to command code re.sec.
       1772 005E
W0015  1774 790B            SB   R11,@WDATLN(R4)    Change data length to >0004
       1776 0064
W0016  1778 D90B            MOVB R11,@BUFLEN(R4)    Change buffer length to >0100
       177A 0062
W0017  177C C924     SKP$RS MOV  @>50(R4),@>4A(R4) Return sector number
       177E 0050
       1780 004A
W0018  1782 21A0            COC  @H0080,R6          Test MSBit >4C for single/multi
       1784 0EE6'
W0019  1786 1609            JNE  RW$PAB             If bit = 0, goto single sector
W0020  1788 D186            MOVB R6,R6              Is this a READ SECTORS op.?
W0021  178A 1304            JEQ  WS$2$3             If not, process as a write
W0022  178C D924            MOVB @>52(R4),@BUFLEN(R4) Buffer length = sectors*256
       178E 0052
       1790 0062
W0023  1792 1003            JMP  RW$PAB             Continue process
W0024  1794 D924     WS$2$3 MOVB @>52(R4),@WDATLN(R4) Data length = sectors*256
       1796 0052
       1798 0064
W0025  179A C064     RW$PAB MOV  @>4E(R4),R1        R1 = RAM buffer pointer
       179C 004E
W0026  179E 06A0            BL   @MXMIT             Transmit the PAB on the hexbus
       17A0 0BCA'
W0027  17A2 0460            B    @LOSCOM            ERROR EXIT
       17A4 18A8'
W0028  17A6 C024            MOV  @>50(R4),R0        R0 = starting sector number
       17A8 0050
W0029  17AA 06A0            BL   @WRDOUT            Send out starting sector number
       17AC 167E'
W0030  17AE 0200            LI   R0,>0100           Set R0 for 1 sector
       17B0 0100
W0031  17B2 21A0            COC  @H0080,R6          Multiple sectors?
       17B4 0EE6'
W0032  17B6 1602            JNE  SECT$X             If not, continue
W0033  17B8 D024            MOVB @>52(R4),R0        Set R0 to number of sectors
       17BA 0052
W0034  17BC 06A0     SECT$X BL   @BYOUT             Send LSByte of # of sectors
       17BE 0DFE'
```

```
W0035 17C0 06C0          SWPB R0
W0036 17C2 06A0          BL   @BYOUT            Send MSByte of # of sectors
      17C4 0DFE'
W0037 17C6 0200          LI   R0,-4             Decrement data length (4 bytes)
      17C8 FFFC
W0038 17CA A900          A    R0,@WDATLN(R4)
      17CC 0064
W0039 17CE C1E4          MOV  @>4E(R4),R7       R7 = data buffer pointer
      17D0 004E
W0040 17D2 0205          LI   R5,XT3$RT         Set up subroutine exit
      17D4 17DA'
W0041 17D6 06A0          BL   @XMIT3
      17D8 0C52'
W0042 17DA 0460  XT3$RT  B    @LOSCOM           ERROR EXIT
      17DC 18A8'
W0043 17DE 0460          B    @FPROT5           Exit 'SUBR'
      17E0 18AC'
W0044                    *
W0045                    ***************************************************************
```

```
W0047                ***************************************************************
W0048                *
W0049                *       SUBROUTINE 11: FORMAT MEDIA
W0050                *
W0051  17E2 C1CB  DFORM  MOV   R11,R7            Store return address
W0052  17E4 06A0         BL    @ENTER            Initialize
       17E6 1526'
W0053  17E8 C1A4         MOV   @>4C(R4),R6       R6 = flag word
       17EA 004C
W0054  17EC D920         MOVB  @FORMOP,@COMMAD(R4) Command = format
       17EE 0ED0'
       17F0 005E
W0055  17F2 0207         LI    R7,>0002          Ext.Format expects 2 byte resp.
       17F4 0002
W0056  17F6 C907         MOV   R7,@BUFLEN(R4)
       17F8 0062
W0057  17FA 0207         LI    R7,>0006          Ext.Format sends 6 bytes
       17FC 0006
W0058  17FE C907         MOV   R7,@WDATLN(R4)
       1800 0064
W0059  1802 06A0         BL    @MXMIT            Send PAB to hexbus
       1804 0BCA'
W0060  1806 0460         B     @LOSCOM           ERROR EXIT
       1808 18A8'
W0061  180A 0207         LI    R7,>0001          Set for single side
       180C 0001
W0062  180E C206         MOV   R6,R8             R8 = flag word
W0063  1810 0A88         SLA   R8,8              MSByte R8 = number of tracks
W0064  1812 0209         LI    R9,4*256          R5 = interleaving/skewing
       1814 0400
W0065  1816 D020         MOVB  @DSKLUN+1,R0      Get disk version number
       1818 E071
W0066  181A 130A         JEQ   FM$XFR            If 0, transfer default params.
W0067  181C 9824         CB    @>50(R4),@H01     Is this single density?
       181E 0050
       1820 0EF8'
W0068  1822 1302         JEQ   FM$10             If yes, single density, default
W0069  1824 0267         ORI   R7,>8000          If not, set double density bit
       1826 8000
W0070  1828 06C7  FM$10  SWPB  R7                Get # of sides in MSByte
W0071  182A D1E4         MOVB  @>51(R4),R7       Set number of sides
       182C 0051
W0072  182E 06C7         SWPB  R7                Restore R7 order
W0073  1830 02A6  FM$XFR STWP  R6                R6 = workspace pointer
W0074  1832 0226         AI    R6,2*R7           R6 points to R7 in WS
       1834 000E
W0075  1836 0203         LI    R3,6              R3 = byte count to transfer
       1838 0006
W0076  183A D036  FM$LOP MOVB  *R6+,R0           R0 = byte to send
W0077  183C 06A0         BL    @BYOUT            Send byte to hexbus
       183E 0DFE'
W0078  1840 0603         DEC   R3                One less byte to send
W0079  1842 16FB         JNE   FM$LOP            Loop until all bytes sent
W0080  1844 06A0         BL    @WRDIN            Get data length returned
       1846 1670'
W0081  1848 C180         MOV   R0,R6             R6 = data length returned     1
W0082  184A 1307         JEQ   FM$EX1            If data length = 0, exit      1
W0083  184C 0286         CI    R6,>0002          Is the returned data length 2?
       184E 0002
W0084  1850 1606         JNE   FM$EXT            If not, error
```

```
W0085 1852 06A0          BL   @WRDIN           Get the number of sectors fmtd.
      1854 1670'
W0086 1856 C900          MOV  R0,@>4A(R4)      Store at >4A->4B
      1858 004A
W0087 185A 0460  FM$EX1  B    @FPROT3          Get status and exit            1
      185C 1950'
W0088 185E 0460  FM$EXT  B    @FPROT6          Toss data and error exit
      1860 189E'
W0089            *
W0090            ********************************************************************
```

```
W0092                ***************************************************************
W0093                *
W0094                *       SUBROUTINE 12:  MODIFY PROTECTION
W0095                *
W0096  1862 C1CB  FPROT  MOV   R11,R7            Store return address
W0097  1864 06A0         BL    @ENTER           Initialize
       1866 1526'
W0098  1868 D920         MOVB  @MFPOP,@COMMAD(R4) Set command code
       186A 0ED3'
       186C 005E
W0099  186E 0206         LI    R6,11            Data length = 10 chars. + prot.
       1870 000B
W0100  1872 C906         MOV   R6,@WDATLN(R4)   Set data length in PAB
       1874 0064
W0101  1876 06A0         BL    @MXMIT           Send PAB over hexbus
       1878 0BCA'
W0102  187A 0460         B     @LOSCOM          ERROR EXIT
       187C 18A8'
W0103  187E D024         MOVB  @>4D(R4),R0      Is file protection requested?
       1880 004D
W0104  1882 1302         JEQ   FPROT1           If not, unprotect the file
W0105  1884 0200         LI    R0,>8000         If yes, send protect byte
       1886 8000
W0106  1888 06A0  FPROT1 BL    @BYOUT           Send either >00 or >80 out
       188A 0DFE'
W0107  188C 0606         DEC   R6               One less byte to send out
W0108  188E C064         MOV   @>4E(R4),R1      R1 points to file name
       1890 004E
W0109  1892 06A0         BL    @XVTH            Send out the file name
       1894 165C'
W0110  1896 06A0  FPROT2 BL    @WRDIN           Get the retunred data length
       1898 1670'
W0111  189A C180         MOV   R0,R6            Is data length 0?
W0112  189C 1359         JEQ   FPROT3           If yes, go to read status byte
W0113  189E 0586  FPROT6 INC   R6               Include status in data length
W0114  18A0 06A0  FPROT4 BL    @BYTIN           Read byte from bus and toss
       18A2 0E18'
W0115  18A4 0606         DEC   R6               One less byte to toss
W0116  18A6 16FC         JNE   FPROT4           Continue tossing bytes
W0117  18A8 0200  LOSCOM LI    R0,>6000         Set Loss-of-Comm. error
       18AA 6000
W0118  18AC D040  FPROT5 MOVB  R0,R1            R1 = error code
W0119  18AE 0981         SRL   R1,8             Make it a word
W0120  18B0 0281         CI    R1,>0066         Is this error code >66?          1
       18B2 0066
W0121  18B4 1612         JNE   FPROSO           If not, continue checking        1
W0122  18B6 04C0         CLR   R0               Make sure R0 is empty            1
W0123  18B8 D024         MOVB  @COMMAD(R4),R0   Get command code sent            1
       18BA 005E
W0124  18BC 0280         CI    R0,>0D00         Format Media command?            1
       18BE 0D00
W0125  18C0 1309         JEQ   FBYTER           If yes, send byte error >21      1
W0126  18C2 0280         CI    R0,>1200         Read Sector command?             1
       18C4 1200
W0127  18C6 1306         JEQ   FBYTER           If yes, send byte error >21      1
W0128  18C8 0280         CI    R0,>1300         Write Sector command?            1
       18CA 1300
W0129  18CC 1303         JEQ   FBYTER           If yes, send byte error >21      1
W0130  18CE 0200         LI    R0,>C000         Otherwise, send 3 bit err.6      1
       18D0 C000
```

```
W0131 18D2 102E          JMP   FPROSX           Exit                              1
W0132 18D4 0200  FBYTER  LI    R0,>2100          Send byte error >21               1
      18D6 2100
W0133 18D8 102B          JMP   FPROSX           Exit                              1
W0134 18DA 0281  FPROS0  CI    R1,9              Code = 9? (WRITE PROTECT)         1
      18DC 0009
W0135 18DE 160F          JNE   FPROS1           If not, continue
W0136 18E0 04C0          CLR   R0               If yes, test which command        1
W0137 18E2 D024          MOVB  @COMMAD(R4),R0   Get the command code              1
      18E4 005E
W0138 18E6 0280          CI    R0,>0D00         Format Media command?             1
      18E8 0D00
W0139 18EA 1303          JEQ   WTPROT           If yes, send error >34            1
W0140 18EC 0280          CI    R0,>1300         Write Sector command?             1
      18EE 1300
W0141 18F0 1603          JNE   WTPRO1           If yes, send error >34            1
W0142 18F2 0200  WTPROT  LI    R0,>3400         Send error >34                    1
      18F4 3400
W0143 18F6 101C          JMP   FPROSX           Exit                              1
W0144 18F8 0200  WTPRO1  LI    R0,>2000         Otherwise, send error >20         1
      18FA 2000
W0145 18FC 1019          JMP   FPROSX           Exit
W0146 18FE 0281  FPROS1  CI    R1,>20           Code = >20?
      1900 0020
W0147 1902 1603          JNE   FPROS2           If not, continue
W0148 1904 0200          LI    R0,>8000         If yes, use >8000
      1906 8000
W0149 1908 1013          JMP   FPROSX           Exit
W0150 190A 0281  FPROS2  CI    R1,>60           Code = >60 - >6F?
      190C 0060
W0151 190E 1A08          JL    FPROS3           If not, continue
W0152 1910 0281          CI    R1,>6F           Code = >60 - >6F?
      1912 006F
W0153 1914 1B05          JH    FPROS3           If not, continue
W0154 1916 0221          AI    R1,->60          Set R1 to table index
      1918 FFA0
W0155 191A D021          MOVB  @DMERR(R1),R0    R0 = table driven error code
      191C 1516'
W0156 191E 1008          JMP   FPROSX           Exit
W0157 1920 0281  FPROS3  CI    R1,HERLEN        Does code fall in HC range?
      1922 001C
W0158 1924 1403          JHE   FPROS4           If not, continue
W0159 1926 D021          MOVB  @HOMERR(R1),R0   R0 = table driven HC error
      1928 0EAA'
W0160 192A 1002          JMP   FPROSX           Exit
W0161 192C 0200  FPROS4  LI    R0,>E000         R0 = code for all other errors
      192E E000
W0162 1930 D900  FPROSX  MOVB  R0,@>50(R4)      Store error code for user
      1932 0050
W0163 1934 D060          MOVB  @HEXCTR,R1       Get current control register
      1936 5FFC
W0164 1938 5060          SZCB  @DISABL,R1       Set for disable comm.
      193A 0012'
W0165 193C D801          MOVB  R1,@HEXCTW       Tell the chip
      193E 5FFA
W0166 1940 C2E0          MOV   @LEVEL1,R11      Restore main return address
      1942 E008
W0167 1944 05CB          INCT  R11              Adjust return address
W0168 1946 D820          MOVB  @MAPRC0,@MAPPER  Restore memory mapper
      1948 0EE4'
```

```
        194A 8810
W0169   194C 1000           NOP                     Let Mapper finish operation   2
W0170   194E 045B           RT                      Return to console
W0171   1950 06A0  FPROT3   BL    @BYTIN            Read status byte into R0
        1952 0E18'
W0172   1954 10AB           JMP   FPROT5            Process status and exit
W0173                *
W0174                ************************************************************
```

```
W0176                ***************************************************************
W0177                *
W0178                *       SUBROUTINE 13:  RENAME FILE
W0179                *
W0180  1956 C1CB  RNAME  MOV   R11,R7            Store return address
W0181  1958 06A0         BL    @ENTER            Initialize
       195A 1526'
W0182  195C D920         MOVB  @MFNOP,@COMMAD(R4) Command = Modify File Name
       195E 0ED5'
       1960 005E
W0183  1962 0206         LI    R6,21             Data = 2 names and separator
       1964 0015
W0184  1966 C906         MOV   R6,@WDATLN(R4)    Set data length to 21 bytes
       1968 0064
W0185  196A 06A0         BL    @MXMIT            Send PAB over hexbus
       196C 0BCA'
W0186  196E 0460         B     @LOSCOM           ERROR EXIT
       1970 18A8'
W0187  1972 C064         MOV   @>50(R4),R1       R1 points to old file name
       1974 0050
W0188  1976 0206         LI    R6,10             R6 = length to be sent
       1978 000A
W0189  197A 06A0         BL    @XVTH             Send old file name to bus
       197C 165C'
W0190  197E 04C0         CLR   R0                R0 = 0
W0191  1980 06A0         BL    @BYOUT            Send zero byte name sparator
       1982 0DFE'
W0192  1984 C064         MOV   @>4E(R4),R1       R1 points to new file name
       1986 004E
W0193  1988 0206         LI    R6,10             R6 = length to be sent
       198A 000A
W0194  198C 06A0         BL    @XVTH             Send the new name to the bus
       198E 165C'
W0195  1990 0460         B     @FPROT2           Go read response exit
       1992 1896'
W0196                *
W0197                ***************************************************************
```

```
W0199                ***************************************************************
W0200                *
W0201                *       SUBROUTINE 14:  DIRECT INPUT
W0202                *
W0203  1994 C1CB  DINP    MOV   R11,R7            Store return address
W0204  1996 06A0          BL    @ENTER            Initialize
       1998 1526'
W0205  199A 06A0          BL    @S14$15           Call common I/O routine
       199C 1694'
W0206  199E 1000  DINPRT  NOP                     First/second return the same
W0207  19A0 06A0          BL    @WRDIN            Get data length returned
       19A2 1670'
W0208  19A4 8900          C     R0,@BUFLEN(R4)    Is buf.len.>=data length
       19A6 0062
W0209  19A8 1202          JLE   DINP1             If not, continue
W0210  19AA 0460          B     @DIN$ER           If yes, error
       19AC 1A24'
W0211  19AE D064  DINP1   MOVB  @>4D(R4),R1       Dealing with FDRs?
       19B0 004D
W0212  19B2 1625          JNE   DINP4             If not, process records
W0213  19B4 D024          MOVB  @>4C(R4),R0       Is this an extended operation?
       19B6 004C
W0214  19B8 113D          JLT   DINPV             If yes, go handle it
W0215  19BA 0206          LI    R6,12             If not, ignore 1st 12 bytes
       19BC 000C
W0216  19BE 06A0  DINP2   BL    @BYTIN            Read and toss a byte
       19C0 0E18'
W0217  19C2 0606          DEC   R6                One less byte to toss
W0218  19C4 16FC          JNE   DINP2             Continue tossing
W0219  19C6 06A0          BL    @WRDIN            Get status and # of recs./AU
       19C8 1670'
W0220  19CA 06C0          SWPB  R0                Put back into proper order
W0221  19CC CA00          MOV   R0,@4(R8)         Store in addtl info block
       19CE 0004
W0222  19D0 06A0          BL    @WRDIN            Get number of AUs
       19D2 1670'
W0223  19D4 06C0          SWPB  R0                Put bytes in proper order
W0224  19D6 CA00          MOV   R0,@2(R8)         Store in addtl info block
       19D8 0002
W0225  19DA 06A0          BL    @WRDIN            Get EOF position and LRL
       19DC 1670'
W0226  19DE 06C0          SWPB  R0                Put bytes in proper order
W0227  19E0 CA00          MOV   R0,@6(R8)         Store in addtl info block
       19E2 0006
W0228  19E4 06A0          BL    @WRDIN            Get # of Level 3 buffers
       19E6 1670'
W0229  19E8 06C0          SWPB  R0                Put bytes in order
W0230  19EA CA00          MOV   R0,@8(R8)         Store in addtl info block
       19EC 0008
W0231  19EE 0206          LI    R6,8              Ignore last 8 bytes
       19F0 0008
W0232  19F2 06A0  DINP3   BL    @BYTIN            Read and toss a byte
       19F4 0E18'
W0233  19F6 0606          DEC   R6                One less byte to toss
W0234  19F8 16FC          JNE   DINP3             Continue tossing bytes
W0235  19FA 0460          B     @FPROT3           Go read status and exit
       19FC 1950'
W0236  19FE C058  DINP4   MOV   *R8,R1            R1 = RAM buffer address
W0237  1A00 06A0          BL    @SUWRIT           Set up for RAM write
       1A02 0A72'
```

```
W0238 1A04 C180          MOV  R0,R6              R6 = data length to recv.
W0239 1A06 C900          MOV  R0,@RDATLN(R4)     Set RDATLN in scratch RAM
      1A08 0066
W0240 1A0A 0205          LI   R5,DINP5           Set up error exit
      1A0C 1A12'
W0241 1A0E 06A0          BL   @MRECV8            Get data and status byte
      1A10 0D28'
W0242 1A12 0460  DINP5   B    @LOSCOM            ERROR EXIT
      1A14 18A8'
W0243 1A16 D000          MOVB R0,R0              Did an error occur?
W0244 1A18 1603          JNE  DINP6              If yes, error
W0245 1A1A D924          MOVB @RDATLN(R4),@>4D(R4) Store actaul # AUs read
      1A1C 0066
      1A1E 004D
W0246 1A20 0460  DINP6   B    @FPROT5            Go process errors and exit
      1A22 18AC'
W0247 1A24 0205  DIN$ER  LI   R5,DIN$E1          Set error exit
      1A26 1A2C'
W0248 1A28 06A0          BL   @MRECO5            Read and toss bytes
      1A2A 0CEE'
W0249 1A2C 0460  DIN$E1  B    @LOSCOM            ERROR EXIT
      1A2E 18A8'
W0250 1A30 0460          B    @FPROT5            Process error and exit
      1A32 18AC'
W0251 1A34 0206  DINPV   LI   R6,10              Trash 10 bytes (filename)
      1A36 000A
W0252 1A38 06A0  DINPV1  BL   @BYTIN             Read and toss a byte
      1A3A 0E18'
W0253 1A3C 0606          DEC  R6                 One less byte to toss
W0254 1A3E 16FC          JNE  DINPV1             Continue tossing bytes
W0255 1A40 C048          MOV  R8,R1              R1 points to info block
W0256 1A42 0221          AI   R1,4               Adjust to after 4th byte
      1A44 0004
W0257 1A46 06A0          BL   @VDPWD             Set up for VDP write
      1A48 09FA'
W0258 1A4A 0206          LI   R6,18              18 bytes to recv.
      1A4C 0012
W0259 1A4E 06A0  DINPV2  BL   @BYTIN             Get a byte from the bus
      1A50 0E18'
W0260 1A52 DBC0          MOVB R0,@VWD(R15)       Send it to the info block
      1A54 FFFE
W0261 1A56 0606          DEC  R6                 One less byte to recv.
W0262 1A58 16FA          JNE  DINPV2             Continue getting bytes
W0263 1A5A C048          MOV  R8,R1              R1 points to info block
W0264 1A5C 0221          AI   R1,8               Point to 8th byte of block
      1A5E 0008
W0265 1A60 06A0          BL   @VDPRD             Set up to read VDP
      1A62 09FE'
W0266 1A64 D02F          MOVB @VRD(R15),R0       Get MSByte of # AUs
      1A66 FBFE
W0267 1A68 06C0          SWPB R0                 Make room for next byte
W0268 1A6A D02F          MOVB @VRD(R15),R0       Get LSByte of # AUs
      1A6C FBFE
W0269 1A6E 06C0          SWPB R0                 Put bytes in right order
W0270 1A70 C048          MOV  R8,R1              R1 points to info block
W0271 1A72 05C1          INCT R1                 Point to 3rd byte of block
W0272 1A74 06A0          BL   @VDPWD             Set up for write to VDP
      1A76 09FA'
W0273 1A78 DBC0          MOVB R0,@VWD(R15)       Write MSByte of # AUs
      1A7A FFFE
```

```
W0274 1A7C 06C0          SWPB R0                 Get LSByte
W0275 1A7E DBC0          MOVB R0,@VWD(R15)       Write LSByte of # AUs
      1A80 FFFE
W0276 1A82 0460          B    @FPROT3            Exit
      1A84 1950'
W0277                *
W0278                ************************************************************
```

```
W0280                ***************************************************************
0281                *
W0282                *       SUBROUTINE 15: DIRECT OUTPUT
W0283                *
W0284 1A86 C1CB  DOUTP  MOV  R11,R7           Store return address
W0285 1A88 06A0         BL   @ENTER           Initialize
      1A8A 1526'
W0286 1A8C 06A0         BL   @S14$15          Call common I/O routine
      1A8E 1694'
W0287 1A90 1022  DOTRT  JMP  DOUTP2           Process records
W0288 1A92 C024         MOV  @>4C(R4),R0      Is this an extended operation?
      1A94 004C
W0289 1A96 112A         JLT  DOUTV            If yes, go handle it
W0290 1A98 04C0         CLR  R0               If not, process std. FDRs
W0291 1A9A 06A0         BL   @WRDOUT          Send zero word to bus
      1A9C 167E'
W0292 1A9E C028         MOV  @4(R8),R0        R0 = status/ recs/AU
      1AA0 0004
W0293 1AA2 06C0         SWPB R0               Put bytes in proper order
W0294 1AA4 06A0         BL   @WRDOUT          Send info to bus
      1AA6 167E'
W0295 1AA8 C028         MOV  @2(R8),R0        R0 = # of Level 2 buffers
      1AAA 0002
W0296 1AAC 06C0         SWPB R0               Put bytes in order
W0297 1AAE 06A0         BL   @WRDOUT          Send info to bus
      1AB0 167E'
W0298 1AB2 C028         MOV  @6(R8),R0        R0 = eof/lrl
      1AB4 0006
W0299 1AB6 06C0         SWPB R0               Put bytes in order
W0300 1AB8 06A0         BL   @WRDOUT          Send info to bus
      1ABA 167E'
W0301 1ABC C028         MOV  @8(R8),R0        R0 = # of Level 3 buffers
      1ABE 0008
W0302 1AC0 06C0         SWPB R0               Put bytes in order
W0303 1AC2 06A0         BL   @WRDOUT          Send infor to bus
      1AC4 167E'
W0304 1AC6 04C0         CLR  R0               R0 = data to be sent
W0305 1AC8 0206         LI   R6,8             R6 = number of bytes to send
      1ACA 0008
W0306 1ACC 06A0  DOUTP1 BL   @BYOUT           Send a byte of zeros to bus
      1ACE 0DFE'
W0307 1AD0 0606         DEC  R6               One less byte of zeros to send
W0308 1AD2 16FC         JNE  DOUTP1           Continue until 8 zeros sent
W0309 1AD4 1009         JMP  DOUTP3           Continue
W0310 1AD6 04C0  DOUTP2 CLR  R0               Process records
W0311 1AD8 06A0         BL   @BYOUT           Send out a byte of zeros
      1ADA 0DFE'
W0312 1ADC C058         MOV  *R8,R1           R1 = RAM address of buffer
W0313 1ADE C1A4         MOV  @WDATLN(R4),R6   R6 = number of bytes to send
      1AE0 0064
W0314 1AE2 0606         DEC  R6               One less byte to send (zero)
W0315 1AE4 06A0         BL   @XVTH            Transmit buffer to hexbus
      1AE6 165C'
W0316 1AE8 0460  DOUTP3 B    @FPROT2          Get response and exit
      1AEA 1896'
W0317 1AEC 52A0  DOUTV  SZCB @H0800,R10       Make sure were talking to VDP
      1AEE 0F08'
W0318 1AF0 0228         AI   R8,4             Point to 4th byte of info block
      1AF2 0004
W0319 1AF4 C048         MOV  R8,R1            R1 = VDP read address
```

```
W0320 1AF6 0206          LI   R6,18            R6 = number of bytes to send
      1AF8 0012
W0321 1AFA 06A0          BL   @XVTH            Send the FDR to the hexbus
      1AFC 165C'
W0322 1AFE 10F4          JMP  DOUTP3           Exit
W0323              *
W0324              ********************************************************************
```

```
W0326                ***************************************************************
W0327                *
W0328                *       SUBROUTINE 16:  BUFFER RELOCATION
W0329                *
W0330  1B00 02A4  RELOC  STWP R4                 Get workspace pointer in R4
W0331  1B02 0224         AI   R4,->E0           Set R4 to RAM start
       1B04 FF20
W0332  1B06 04C0         CLR  R0                Set R0 to 0
W0333  1B08 D024         MOVB @>4C(R4),R0       R0 = # of disk buffers req.
       1B0A 004C
W0334  1B0C 1306         JEQ  RELOC2            If request is 0, return error
W0335  1B0E 0280         CI   R0,>0400          Is request greater than 4?
       1B10 0400
W0336  1B12 1B03         JH   RELOC2            If yes, return an error
W0337  1B14 04E4         CLR  @>50(R4)          If not, return successfully
       1B16 0050
W0338  1B18 1002         JMP  RELOX             Go return to caller
W0339  1B1A 0724  RELOC2 SETO @>50(R4)          Set error condiiton
       1B1C 0050
W0340  1B1E 05CB  RELOX  INCT R11               Bump to second return address
W0341  1B20 045B         RT                     Return to console caller
W0342                *
W0343                ***************************************************************
```

```
 0024                    END
NO ERRORS,      NO WARNINGS
```

```
LABEL    VALUE   DEFN  REFERENCES

$        1B22'         P0029 P0065 P0107 P0127 P0130 P0133 P0135 P0137 P0142
                       P0145 P0147
ACCMOD   04B2'  E0320 F0045
ALCFLG   005C   A0174 F0055 I0068 I0070 N0092
ALLDON   0BB8'  M0096 M0087
BADATO   0ADE'  M0025 E0168 F0087 R0010
BADATT   0358'  E0168 E0142 E0146 E0148 E0157 E0163 E0185 E0190 E0261 E0362
                       T0016
BADEXT   0AFA'  M0033 M0014 M0024 M0026 M0028 M0030
BADOP    0AE4'  M0027 E0019 E0060 E0273 R0009 T0012 U0016
BADOPC   0436'  E0273 E0304
BAVAIE   0016'  B0022 Q0037 Q0045 U0047 U0152
BAVAIS   1000   A0149
BAVC     0018'  B0023 D0019 Q0055
BAVIAE   0014'  B0021 D0024 N0177 N0196 N0199 Q0039 Q0060
BAVIAS   2000   A0148
BEG1     0202'  E0005
BEG2     0224'  E0015 E0011
BLOKAV   E06E   A0128 V0126 V0137 V0139 V0143 V0171
BRKOFF   0E98'  P0018 P0015
BRKOP    0ED2'  P0087 J0024
BUFADR   004C   A0163 H0017 H0053 I0053 I0090 I0111 N0106 S0043 S0080 U0144
                       U0231
BUFCP1   10C2'  Q0176 Q0174
BUFCP2   10CC'  Q0180 Q0177
BUFCP3   10DA'  Q0186 Q0131
BUFCPU   10AA'  Q0166 Q0106
BUFLEN   0062   A0179 E0276 F0067 F0122 G0010 H0014 H0047 I0015 I0051 I0087
                       I0110 I0119 J0008 J0045 J0056 K0016 M0050 N0051 N0053
                       N0136 V0063 V0308 V0318 W0016 W0022 W0056 W0208
BYOUT    0DFE'  00035 C0031 C0033 L0173 N0042 N0044 N0046 N0048 N0050 N0052
                       N0054 N0066 N0068 N0089 N0091 N0093 N0097 N0105 N0110
                       N0117 Q0085 Q0088 Q0148 Q0150 R0022 R0024 S0046 V0252
                       V0275 V0277 W0034 W0036 W0077 W0106 W0191 W0306 W0311
BYOUT1   0E08'  00038 00040
BYOUT2   0E12'  00041 00037
BYTIN    0E18'  00057 N0130 N0132 N0139 N0159 N0168 N0214 N0236 Q0091 Q0093
                       Q0103 Q0124 S0088 S0100 S0106 S0109 S0114 V0264 V0266
                       W0114 W0171 W0216 W0232 W0252 W0259
BYTIN1   0E2A'  00063 00069
BYTIN2   0E2E'  00064 00071
BYTIN4   0E46'  00073 00062 00079 00086
BYTIN5   0E5C'  00080 00088
BYTIN6   0E70'  00087 00083
BYTIN9   0E82'  00093 00060 00067 00077 00085 S0085
CATBAD   12B8'  T0012
CATENT   12A6'  T0006 E0313
CATENX   12BC'  T0013 T0009
CATOP    0ED1'  P0086 T0026
CATOP1   12CC'  T0017 T0015
CATOP2   12C8'  T0016 T0018 U0032 U0038
CATOPN   12C0'  T0014 T0007
CATRD    12DC'  T0026 T0011
CBUFLN   0002   A0108 Q0109
CBUFOF   0004   A0110 Q0107 Q0115 Q0116 Q0133
CBUFRO   0003   A0109 Q0112
CBUFST   0000   A0107 Q0032 Q0104 Q0118 Q0129 Q0135
CHABRT   0E88'  P0012 H0009 H0038
CHKBRK   0020   A0101 P0014
CHKDS4   15B2'  V0128 V0136 V0141
```

```
LABEL     VALUE   DEFN  REFERENCES

CHKDS5    15C6'   V0137 V0132
CHKDS6    15C2'   V0135 V0138
CHKDSC    15D6'   V0142 V0129
CHKDSK    15A0'   V0122 F0059 J0039 M0042 V0197
CHKDSX    15F2'   V0151 V0124 V0144 V0147 V0149
CHKDSZ    15EE'   V0150 V0134
CHRCNT    004E    A0164 D0080 D0084 H0011 H0020 H0045 M0076 S0036 S0119 U0202
                        U0232
CLOSE     0648'   G0005 E0053 G0023
CLOSE1    066E'   G0014
CLOSE2    065A'   G0009 G0007
CLOSLS    067E'   G0022 F0013 I0058 I0061 I0069 I0124
CLOSOP    0EC7'   P0076 G0005 G0006 M0048
CLRERR    01F6'   D0095 D0085
CLRINT    0010'   B0019 D0018
COMDSK    161A'   V0196 G0008 H0012 H0039 J0006 K0029
COMDSX    162A'   V0202 V0198 V0199
COMMAD    005E    A0176 E0274 F0056 F0121 G0005 H0010 H0049 H0052 I0012 I0025
                        I0028 I0047 I0049 I0085 I0107 J0005 J0024 J0036 J0055
                        K0013 K0028 M0048 N0043 N0056 N0058 N0080 N0100 N0102
                        N0151 T0026 V0322 W0009 W0014 W0054 W0098 W0123 W0137
                        W0182
COMPAR    1576'   V0085 V0133
COMPRL    1592'   V0097 V0101
COMPRX    159E'   V0102 V0087 V0091 V0094 V0099
COMRT1    0AC0'   M0015 M0012
COMRTN    0AAC'   M0008 G0017 H0027 I0033 I0042 I0059 I0071 I0104 I0125 J0018
                        J0061 K0035 K0055 T0020
COMSTS    006A    A0183 F0080 F0126 H0021 I0020 I0040 I0056 I0068 I0070 I0093
                        I0102 I0116 I0122 J0012 J0053 K0024 K0033 M0008 M0031
                        N0148 N0171 N0174 R0016 S0026 S0111 S0115 S0123 T0019
                        V0222
COPYC     0BB2'   M0093 M0095
COPYV     0B94'   M0084 M0086
CPUACS    0A08'   L0041 D0088 M0089
CPUPAB    01D8'   D0086 D0067
CR7       001E'   B0026 U0131 U0264
D10       0EF0'   P0128 E0348 V0341
D20       0F13'   P0154 E0079
D70       0F12'   P0153 E0081
DELET1    0930'   J0053 J0050
DELET2    0938'   J0055 J0042 J0043
DELETE    08F6'   J0036 E0059
DELEXT    0954'   J0061 J0052 J0054
DELIT     0910'   J0044 J0040 J0041
DELNOP    0ECC'   P0081 J0055
DELOOP    0EC8'   P0077 J0036
DEVCOD    005D    A0175 E0024 E0079 E0081 E0106 E0165 F0023 F0129 N0041 R0011
                        U0146 V0053 V0086 V0218
DEVERR    0AF6'   M0032 M0018 M0022 N0150 P0017 S0124 U0077 U0147 U0248
DEVLEN    0058    A0172 D0064 D0087 E0085 E0109 E0141 E0200 L0010 L0042 M0038
DFORM     17E2'   W0051 B0087
DIN$E1    1A2C'   W0249 W0247
DIN$ER    1A24'   W0247 W0210
DINP      1994'   W0203 B0093
DINP1     19AE'   W0211 W0209
DINP2     19BE'   W0216 W0218
DINP3     19F2'   W0232 W0234
DINP4     19FE'   W0236 W0212
DINP5     1A12'   W0242 W0240
```

```
LABEL     VALUE   DEFN  REFERENCES

DINP6     1A20'   W0246 W0244
DINPRT    199E'   W0206 V0295 V0306 V0315
DINPV     1A34'   W0251 W0214
DINPV1    1A38'   W0252 W0254
DINPV2    1A4E'   W0259 W0262
DISABL    0012'   B0020 C0040 M0056 N0183 Q0059 R0031 U0057 W0164
DMERR     1516'   V0019 W0155
DOTRT     1A90'   W0287
DOUTP     1A86'   W0284 B0095
DOUTP1    1ACC'   W0306 W0308
DOUTP2    1AD6'   W0310 W0287
DOUTP3    1AE8'   W0316 W0309 W0322
DOUTV     1AEC'   W0317 W0289
DSK       00CA'   B0125 B0058
DSK1      00D0'   B0127 B0062
DSK2      00D6'   B0129 B0066
DSK3      00DC'   B0131 B0070
DSK4      00E2'   B0133 B0074
DSKCL1    1640'   V0224 V0221
DSKENT    02DA'   E0104 E0014
DSKEXT    163A'   V0222 V0219
DSKEXX    165A'   V0233 V0223 V0225
DSKLUN    E070    A0129 V0060 V0140 V0148 W0065
DSKXCD    1636'   V0220 G0016 J0051 M0047
DSKXOP    1630'   V0218 F0098
DSR2      0030'   B0046 B0042
DSR3      003C'   B0050 B0046
DSR4      0048'   B0054 B0050
DSR5      0054'   B0058 B0054
DSR6      005C'   B0062 B0058
DSR7      0066'   B0066 B0062
DSR8      0070'   B0070 B0066
DSR9      007A'   B0074 B0070
DSREO1    0170'   D0022 D0019
DSRENO    015E'   D0016 B0118 B0120 B0122 B0124 B0126 B0128 B0130 B0132 B0134
DSREN1    0182'   D0062 D0025
DSRLNK    0026'   B0042 B0010
ENSERV    0618'   F0119 F0101
ENSRV1    0640'   F0129 F0127
ENSVOP    0ECE'   P0083 F0121
ENTER     1526'   V0043 W0006 W0052 W0097 W0181 W0204 W0285
ENTER1    1534'   V0048 V0046
ENTERV    1556'   V0058 V0056
EOFERR    0F14'   P0156 I0122
ERSTAT    0056    A0170 D0063 D0095 M0034 M0070
EXIT      0B3E'   M0054 E0285 M0035 M0043 M0045 M0046 R0040 S0120 T0013
EXIT1     0B4E'   M0065 S0064 U0049 U0203 U0247
EXPABC    0B9E'   M0088 M0066
FAC       004A    A0160 A0161 A0162 A0163 A0164 A0165 A0166 A0167 A0168 A0169
                  A0170 A0171 A0172 A0173 A0174 A0175 A0176 A0177 A0178
                  A0179 A0180 A0181 A0182 A0183
FBYTER    18D4'   W0132 W0125 W0127 W0129
FDR$1     16E8'   V0313 V0303
FDR$2     1700'   V0320 V0316
FILERR    0AD8'   M0023 E0115 F0061 M0020 V0200 V0201
FLGSTS    004B    A0162 F0036 H0023 H0040 J0014 M0067 M0069 M0073 S0020 T0014
                  U0031
FM$10     1828'   W0070 W0068
FM$EX1    185A'   W0087 W0082
FM$EXT    185E'   W0088 W0084
```

```
LABEL     VALUE   DEFN  REFERENCES

FM$LOP    183A'   W0076 W0079
FM$XFR    1830'   W0073 W0066
FOLOOP    03F2'   E0244 E0250
FORCE     0B20'   M0047 M0044
FOREN1    043A'   E0274 E0272
FOREN2    0440'   E0275
FOREN3    0456'   E0281 R0039
FOREN4    0462'   E0284 E0282
FORENT    0426'   E0269 E0312
FORENX    0466'   E0285 E0270
FORMOP    0ED0'   P0085 E0274 I0022 I0095 K0026 W0054
FORTBL    04A8'   E0319 E0243
FPROS0    18DA'   W0134 W0121
FPROS1    18FE'   W0146 W0135
FPROS2    190A'   W0150 W0147
FPROS3    1920'   W0157 W0151 W0153
FPROS4    192C'   W0161 W0158
FPROSX    1930'   W0162 W0131 W0133 W0143 W0145 W0149 W0156 W0160
FPROT     1862'   W0096 B0089
FPROT1    1888'   W0106 W0104
FPROT2    1896'   W0110 W0195 W0316
FPROT3    1950'   W0171 W0087 W0112 W0235 W0276
FPROT4    18A0'   W0114 W0116
FPROT5    18AC'   W0118 W0043 W0172 W0246 W0250
FPROT6    189E'   W0113 W0088
GETNM0    04BE'   E0341 E0359
GETNM2    04F0'   E0360 E0354
GETNMX    04F6'   E0362 E0342 E0344 E0350 E0352 F0042
GETNUM    04B6'   E0338 E0161 E0188
GETSW     03B0'   E0219 E0160
GETSW1    03C4'   E0226 E0230
GETSW2    03DC'   E0237 E0232
GETSW3    0408'   E0252 E0239
GETSWX    0422'   E0261 E0224 E0228 E0234 E0241 E0248 E0253 E0257
H00       0EF0'   P0127 D0095 D0096 E0271 E0301 F0073 T0019
H0004     0EEC'   P0125 V0337
H0009     0EEE'   P0126 S0024 U0037
H0080     0EE6'   P0122 W0018 W0031
H00FF     0F00'   P0141 E0283 U0035
H01       0EF8'   P0134 E0076 E0269 E0303 G0014 K0040 K0051 Q0108 R0007 S0009
                  T0008 W0067
H0100     0EFE'   P0140 F0046 J0015
H02       0EF4'   P0131 S0011 T0010
H0200     0EE8'   P0123 J0049 U0155 V0123 V0224
H03       0F04'   P0144 E0073 E0299 S0013
H04       0EFB'   P0137 K0043 P0138 S0007 T0014
H0400     0EEA'   P0124 F0041 F0043 N0069 N0143 N0169
H06       0F02'   P0143 E0297
H07       0F14'   P0155 P0156
H08       0F08'   P0147 F0025
H0800     0F08'   P0148 D0062 D0086 F0049 F0051 F0088 L0056 L0071 L0088 L0103
                  L0121 L0139 L0156 M0065 W0317
H09       0EFC'   P0139 E0017
H0A       0EF2'   P0129 Q0125
H0C       0F10'   P0152 N0142
H10       0EF9'   P0135 E0015 E0040 E0071 K0053 S0020 U0031
H1000     0F0C'   P0150 E0305 F0039 F0052 F0054 H0024 H0050 I0026 I0035 I0045
H2000     0F0A'   P0149 E0038 F0009 F0048 H0007 H0036
H40       0EFA'   P0136 K0046
H4000     0F0E'   P0151 N0172 Q0105 Q0130
```

```
LABEL     VALUE   DEFN  REFERENCES

H48       0EF7'   P0133 I0037 I0100
H80       0F06'   P0145 E0005 E0007
H8000     0F06'   P0146 D0066 D0087 F0007 F0017 F0099 F0119 N0062 N0113
H88       0EF6'   P0132 I0034
HEREND    0EC6'   P0065 P0066
HERLEN    001C    P0066 M0011 W0157
HEXBUS    00B4'   B0117 B0054
HEXCTR    5FFC    A0142 C0027 C0039 D0023 M0055 N0037 N0176 N0182 N0193 O0017
                        O0089 Q0019 Q0043 Q0054 Q0058 Q0079 Q0144 R0018 R0030
                        S0039 U0042 U0045 U0056 U0124 U0130 U0151 U0164 U0170
                        U0228 U0263 W0163
HEXCTW    5FFA    A0145 C0011 C0029 C0041 M0057 N0040 N0178 N0184 N0197 N0200
                        O0020 O0091 Q0046 Q0061 Q0082 Q0146 R0020 R0032 S0042
                        U0048 U0126 U0132 U0154 U0166 U0172 U0230 U0265 W0165
HEXMTR    5FFE    A0143
HEXMTW    5FF8    A0144 O0041 U0087
HEXRDD    5FF8    A0140 O0093 U0133 U0241
HEXSTS    5FFA    A0141 C0036 O0011 O0035 O0038 O0057 O0064 O0074 O0080 Q0023
                        Q0153 R0027 S0055 S0074 U0039 U0073 U0080 U0108 U0118
                        U0158 U0167 U0233
HFE       0EF3'   P0130 Q0178
HFF       0F01'   P0142 D0084 M0031 Q0191
HLDHSK    14E8'   U0258 U0182 U0199
HOMERR    0EAA'   P0037 M0013 P0066 W0159
HSKRD     4000    A0147 O0059 O0066 O0076 O0084 S0057 U0110 U0120 U0159
HSKWT     8000    A0146 C0037 O0039 Q0154 R0028 U0082
INQSAV    0ED8'   P0093 I0012
INTEN0    0F16'   Q0019 B0105
INTEN1    0F22'   Q0023 Q0021
INTEN2    0F34'   Q0030 Q0028
INTEN3    0F9C'   Q0071 Q0057
INTEN4    0FD2'   Q0087 Q0090
INTEN5    0FAE'   Q0077
INTEN6    0F6A'   Q0049 Q0040
INTENC    0F46'   Q0037 Q0033 Q0034
INTENX    0F82'   Q0058 Q0053 Q0139
INTEX     1068'   Q0139 Q0077 Q0126 Q0128 Q0140 Q0156 Q0192
INTLN     100E'   Q0112 Q0110
INTLN1    101C'   Q0116 Q0113
INTLN2    0FF0'   Q0103 Q0123
INTLN3    1036'   Q0124 Q0097
INTLN5    1032'   Q0122 Q0185
INTLNK    00AE'   B0105 B0012
INTNUL    0F98'   Q0070 Q0073
INTX      0F8A'   Q0060 Q0056
INTXXX    0F94'   Q0063 Q0042 Q0048
LEVEL1    E008    A0120 D0020 D0022 M0096 M0097 V0065 W0166
LEVEL2    E00A    A0121 E0134 E0204 F0107 F0109 G0022 J0037 J0057 V0196 V0202
                        V0287 V0313 V0342
LEVEL3    E00C    A0122 E0219 E0259 E0338 E0360 L0120 L0127 L0138 L0145 V0273
                        V0278
LOAD      0806'   I0083 E0057
LOAD0     0856'   I0104 I0094 I0096
LOAD1     085A'   I0105 I0103
LOAD2     0880'   I0113 I0121
LOAD3     08A6'   I0122 I0117
LOADN     0844'   I0099
LOADOP    0ED6'   P0091 I0085
LOSCOM    18A8'   W0117 V0047 V0324 W0027 W0042 W0060 W0102 W0186 W0242 W0249
LRECLN    0052    A0166 D0076 D0083 E0281 E0283 F0016 F0029 F0092 H0013 H0043
```

```
LABEL     VALUE   DEFN  REFERENCES

                        M0074 M0077 M0078 S0022 S0083 T0017 U0033 U0035 U0145
LSERR     07F2'   I0068 F0128 I0057 I0123
LSRTN     0612'   F0109 F0074 G0015
LUN250    E012    A0124 C0018 R0034 V0005
LUN251    E029    A0125 C0019 R0035 V0007
LUN252    E040    A0126 C0020 R0036 V0009
LUN253    E057    A0127 C0021 R0037 V0011
LUN0      005F    A0177 D0096 E0136 E0192 M0088 N0045 V0061 V0148 V0150 V0226
LUNTBL    1504'   V0005 V0127 V0230
MAPPER    8810    A0117 C0023 M0098 Q0074 Q0141 W0168
MAPRC0    0EE4'   P0115 C0023 M0098 Q0074 Q0141 W0168
MEDFUL    0020    A0099 M0015
MEMSIZ    00FF    A0118 C0015
MFNOP     0ED5'   P0090 W0182
MFPOP     0ED3'   P0088 W0098
MPABL1    0214'   E0009 E0006
MRCOP1    0DA4'   N0214 N0217
MRECO5    0CEE'   N0138 N0213 N0235 W0248
MRECO6    0D64'   N0181 N0173 N0175
MRECO7    0D68'   N0182 N0180
MRECOP    0D9A'   N0211 N0152
MRECST    0DC4'   N0232 N0154 N0156
MRECV     0CCC'   N0128 N0079 N0114
MRECV2    0D16'   N0151 N0137
MRECV3    0D38'   N0168 N0135 N0158 N0224 N0238
MRECV5    0CF0'   N0139 N0141
MRECV6    0D46'   N0172
MRECV7    0D0A'   N0148 N0144
MRECV8    0D28'   N0157 W0241
MRECV9    0D2C'   N0159 N0162
MXMIT     0BCA'   N0034 E0279 F0071 F0124 G0012 H0018 H0054 I0016 I0031 I0054
                        I0091 I0114 J0010 J0047 J0059 K0020 K0031 M0052 V0323
                        W0026 W0059 W0101 W0185
MXMIT1    0C36'   N0062 N0059
MXMIT2    0C7C'   N0096 N0081 N0099
MXMIT3    0CAA'   N0109 N0112
MXMIT4    0CB6'   N0113 N0103
MXMIT6    0C40'   N0065 N0063
MXMIT7    0C88'   N0100 N0095
MXMIT8    0CC0'   N0116 N0119
MXMITQ    0C30'   N0060 N0057
MXMITR    0C98'   N0104 N0101
NAMLEN    0057    A0171 D0079 E0083 E0107 E0139 E0198 M0036 M0078
NOTOPN    0EFB'   P0138 J0053
NULL1     109E'   Q0153 Q0155
NULLOP    0EDB'   P0096 Q0149
OPCODE    004A    A0161 D0072 D0091 E0005 E0007 E0015 E0017 E0039 E0071 E0073
                        E0269 E0271 E0296 F0057 F0073 G0006 G0014 M0083 M0092
                        R0005 S0006 T0006 U0009 V0146
OPEN      04FA'   F0007 E0052
OPENO1    0548'   F0035 F0021 F0024 F0026
OPENO2    0564'   F0044 F0040
OPEN1     0574'   F0049 F0047
OPEN2     057E'   F0052 F0050
OPEN3     0588'   F0055 F0053 F0108
OPEN4     05A8'   F0065 F0058 F0060
OPEN5     05F8'   F0092 F0089
OPEN6     05EC'   F0088 F0084
OPEN8     05FE'   F0098 E0284 F0086
OPENBF    E00E    A0123 F0011 F0012 F0014 F0015 F0082 F0085 F0090 F0092 I0010
```

```
LABEL     VALUE   DEFN  REFERENCES
                        I0011 I0018 I0019 I0043 I0083 I0084 I0097 I0098 I0105
                        J0038 J0058 K0019 K0023 N0211 N0218 N0220 N0221 N0223
OPENC     0528'   F0020 F0018
OPENCT    051A'   F0016 F0008 F0010
OPENLS    060C'   F0107 I0039 I0101
OPENOP    0EC6'   P0075 F0056
OPENX     05E8'   F0087 F0091
OPERR     0234'   E0019 E0074
OPERR1    02FC'   E0115 E0111
OPOK      0238'   E0020 E0016 E0018
OPOK1     0242'   E0024 E0022 E0089 E0114 E0306
OPOK2     027E'   E0043 E0041
OPOK3     0270'   E0039 E0027 E0031 E0035 E0037
OPTBL     0288'   E0052 E0045
OSOMAP    5FF8    A0139 A0140 A0141 A0142 A0143 A0144 A0145
PABACS    09EA'   L0009 D0068 M0079
PABLUN    0055    A0169 M0088
PABPTR    005A    A0173 D0063 E0082 E0112 E0137 E0202 L0009 L0041 M0040
PACMOV    01F0'   D0092 D0094
PARSE     0300'   E0134 E0020
PARSE0    0344'   E0161 E0159
PARSE1    0398'   E0198 E0175 E0180 E0183
PARSE4    035C'   E0174 E0167
PAVMOV    01AC'   D0073 D0075
PRECM0    0DDC'   00011 00015
PRECM1    0DEC'   00017 00013
PRECOM    0DD8'   00010 C0026 N0036 Q0078 Q0143 R0017 S0038
PWRENT    00E8'   C0009 B0032
PWREXT    0150'   C0039 C0025
PWRLNK    0020'   B0032 B0008
R0        0000    C0014 C0017 C0018 C0019 C0020 C0021 C0022 C0030 C0032
                  E0024 E0025 E0026 E0028 E0030 E0032 E0034 E0036 E0139
                  E0140 E0141 E0145 E0155 E0174 E0184 E0187 E0201 E0203
                  E0223 E0233 E0233 E0240 E0251 E0252 E0281 E0296 E0297
                  E0299 E0301 E0303 E0353 E0353 E0357 F0016 F0019 F0020
                  F0022 F0023 F0025 F0027 F0028 F0029 F0057 F0066 F0067
                  F0080 F0083 F0085 F0090 F0126 H0013 H0014 H0021 H0023
                  H0024 H0040 H0041 H0043 H0045 H0046 I0020 I0022 I0040
                  I0056 I0093 I0095 I0102 I0116 K0015 K0016 K0024 K0026
                  K0033 K0037 K0038 K0041 K0044 K0047 K0048 K0049 K0049
                  K0051 L0058 L0060 L0158 L0160 L0172 N0041 N0043 N0045
                  N0047 N0049 N0051 N0053 N0055 N0060 N0061 N0064 N0065
                  N0067 N0080 N0088 N0090 N0092 N0104 N0116 N0131 N0133
                  N0142 N0148 N0151 N0153 N0155 N0171 N0174 N0215 N0218
                  N0219 N0220 N0221 N0222 N0223 N0237 00010 00014 00041
                  00093 P0012 P0016 P0018 Q0049 Q0062 Q0079 Q0080 Q0081
                  Q0082 Q0083 Q0084 Q0087 Q0092 Q0095 Q0114 Q0121 Q0125
                  Q0127 Q0127 Q0132 Q0138 Q0144 Q0145 Q0146 Q0147 Q0149
                  Q0153 Q0154 Q0167 Q0170 Q0172 Q0173 Q0175 Q0176 Q0179
                  Q0180 Q0182 Q0183 Q0188 Q0190 Q0191 R0005 R0007 R0011
                  R0018 R0019 R0020 R0021 R0023 S0006 S0007 S0009 S0011
                  S0013 S0022 S0024 S0086 S0089 S0110 S0111 S0115 S0117
                  S0118 S0119 S0122 S0123 T0006 T0008 T0010 T0017 U0011
                  U0033 U0037 U0039 U0040 U0042 U0043 U0045 U0046 U0047
                  U0048 U0056 U0057 U0087 U0128 U0130 U0131 U0132 U0133
                  U0151 U0152 U0153 U0154 U0155 U0162 U0164 U0165 U0166
                  U0170 U0171 U0172 U0176 U0186 U0228 U0229 U0230 U0241
                  U0258 U0259 U0261 U0262 U0263 U0264 U0265 V0048 V0049
                  V0050 V0051 V0052 V0053 V0088 V0089 V0090 V0092 V0093
                  V0093 V0100 V0130 V0131 V0137 V0143 V0146 V0222 V0226
```

LABEL	VALUE	DEFN	REFERENCES
			V0227 V0228 V0229 V0230 V0231 V0265 V0274 V0276 V0301
			V0305 V0317 V0326 V0332 V0334 V0335 W0028 W0030 W0033
			W0035 W0037 W0038 W0065 W0076 W0081 W0086 W0103 W0105
			W0111 W0117 W0118 W0122 W0123 W0124 W0126 W0128 W0130
			W0132 W0136 W0137 W0138 W0140 W0142 W0144 W0148 W0155
			W0159 W0161 W0162 W0190 W0208 W0213 W0220 W0221 W0223
			W0224 W0226 W0227 W0229 W0230 W0238 W0239 W0243 W0243
			W0260 W0266 W0267 W0268 W0269 W0273 W0274 W0275 W0288
			W0290 W0292 W0293 W0295 W0296 W0298 W0299 W0301 W0302
			W0304 W0310 W0332 W0333 W0335
R1	0001		C0015 C0027 C0028 C0029 C0039 C0040 C0041 D0023 D0024
			E0137 E0278 E0340 E0341 E0343 E0345 E0346 E0351 E0355
			E0358 F0035 F0036 F0037 F0038 F0041 F0043 F0044 F0045
			F0045 F0048 F0051 F0054 F0055 F0070 H0053 I0030 I0034
			I0037 I0053 I0089 I0100 J0058 K0018 L0009 L0010 L0011
			L0012 L0025 L0026 L0027 L0028 L0029 L0073 L0075 L0123
			L0126 L0141 L0144 M0009 M0013 M0023 M0025 M0027 M0029
			M0032 M0033 M0034 M0055 M0056 M0057 N0037 N0038 N0039
			N0040 N0106 N0128 N0176 N0177 N0178 N0182 N0183 N0184
			N0192 N0194 00017 00018 00019 00020 00035 00036 00038
			00039 00057 00058 00059 00064 00065 00066 00074 00075
			00076 00080 00081 00082 00089 00090 00091 Q0032 Q0035
			Q0036 Q0051 Q0052 Q0054 Q0055 Q0058 Q0059 Q0060 Q0061
			Q0104 Q0105 Q0107 Q0108 Q0109 Q0111 Q0112 Q0115 Q0116
			Q0117 Q0118 Q0119 Q0129 Q0130 Q0133 Q0134 Q0135 Q0136
			Q0168 Q0169 Q0170 Q0171 Q0173 Q0176 Q0179 Q0180 Q0181
			Q0182 Q0186 Q0187 Q0188 Q0189 Q0190 S0039 S0040 S0041
			S0042 S0043 S0055 S0056 S0057 S0074 S0075 S0076 S0080
			U0014 U0015 U0073 U0074 U0075 U0080 U0081 U0082 U0148
			U0233 U0234 U0235 U0239 V0049 V0055 V0095 V0098 V0178
			V0181 V0231 V0232 V0339 W0025 W0108 W0118 W0119 W0120
			W0134 W0146 W0150 W0152 W0154 W0155 W0157 W0159 W0163
			W0164 W0165 W0187 W0192 W0211 W0236 W0255 W0256 W0263
			W0264 W0270 W0271 W0312 W0319
R10	000A		B0117 B0119 B0121 B0123 B0125 B0127 B0129 B0131 B0133
			D0026 D0062 D0086 E0008 E0009 E0038 E0075 E0076 E0077
			E0104 E0305 F0007 F0009 F0017 F0039 F0088 F0099 F0119
			F0120 H0007 H0036 H0050 I0026 I0035 I0045 J0049 L0056
			L0071 L0088 L0103 L0121 L0139 L0156 M0065 N0062 N0069
			N0113 N0143 N0169 N0172 N0179 N0192 Q0072 Q0076 Q0083
			V0054 V0057 V0123 V0224 V0290 V0291 V0292 V0293 W0317
R11	000B		C0009 D0020 D0022 E0134 E0204 E0219 E0259 E0338 E0360
			F0107 F0109 G0022 L0012 L0120 L0127 L0138 L0145 L0171
			M0097 N0034 N0145 N0146 P0013 P0019 Q0041 Q0047 Q0050
			V0085 V0122 V0170 V0196 V0202 V0249 V0263 V0273 V0278
			V0287 V0295 V0306 V0315 V0342 W0005 W0010 W0011 W0014
			W0015 W0016 W0051 W0096 W0166 W0167 W0180 W0203 W0284
			W0340
R12	000C		P0012 P0016 P0018
R15	000F		D0073 L0027 L0029 L0058 L0073 L0090 L0105 L0158 M0084
			Q0121 Q0138 W0260 W0266 W0268 W0273 W0275
R2	0002		E0143 E0147 E0156 E0158 E0166 E0182 E0220 E0242 E0249
			E0254 E0256 E0340 E0347 E0348 E0349 E0349 F0037 F0046
			F0049 F0052 J0012 J0014 J0015 K0036 K0040 K0043 K0046
			K0053 K0054 L0090 L0092 M0008 M0010 M0011 M0013 M0015
			M0017 M0019 M0021 N0193 N0196 N0197 N0199 N0200 00011
			00012 00063 00070 00073 00087 Q0166 Q0167 Q0178 Q0183
			Q0184 S0054 S0061 U0009 U0010 U0013 U0014 U0074 U0078
			U0081 U0084 U0108 U0109 U0110 U0112 U0113 U0118 U0119
			U0120 U0158 U0159 U0167 U0168 V0050 V0058 V0059 V0060

```
                        V0127 V0128 V0135 V0140 V0150 V0218 V0220 V0232 V0300
R3          0003        E0162 E0164 E0165 E0177 E0179 E0189 E0191 E0192 E0220
                        E0221 E0229 E0231 E0238 E0245 E0247 E0339 E0347 E0351
                        L0105 L0107 00058 00061 00065 00068 00075 00078 00081
                        00084 S0056 S0059 S0075 S0078 U0109 U0112 U0115 U0119
                        U0122 U0124 U0125 U0126 U0234 U0237 V0086 V0090 V0098
                        V0128 V0131 V0139 V0171 V0172 V0175 V0181 W0075 W0078
R4          0004        C0010 C0011 C0012 C0013 D0016 D0017 D0063 D0063 D0064
                        D0064 D0065 D0071 D0076 D0077 D0079 D0080 D0083 D0084
                        D0087 D0090 D0095 D0096 E0005 E0007 E0015 E0017 E0024
                        E0039 E0071 E0073 E0079 E0081 E0082 E0083 E0085 E0106
                        E0107 E0109 E0112 E0136 E0137 E0139 E0141 E0165 E0192
                        E0198 E0200 E0202 E0269 E0271 E0274 E0275 E0276 E0277
                        E0281 E0283 E0296 F0016 F0020 F0023 F0028 F0029 F0036
                        F0055 F0056 F0057 F0065 F0065 F0067 F0069 F0073 F0080
                        F0082 F0083 F0092 F0121 F0122 F0123 F0126 F0129 G0005
                        G0006 G0009 G0010 G0011 G0014 H0010 H0011 H0013 H0014
                        H0015 H0015 H0016 H0017 H0020 H0020 H0021 H0023 H0026
                        H0040 H0043 H0045 H0046 H0047 H0048 H0048 H0049 H0052
                        H0053 I0012 I0013 I0014 I0015 I0020 I0025 I0028 I0029
                        I0038 I0038 I0040 I0043 I0047 I0049 I0050 I0051 I0052
                        I0052 I0053 I0056 I0068 I0068 I0070 I0070 I0085 I0086
                        I0087 I0087 I0088 I0090 I0093 I0099 I0102 I0105 I0107
                        I0108 I0109 I0110 I0110 I0111 I0111 I0112 I0113 I0116
                        I0118 I0118 I0119 I0119 I0120 I0120 I0122 J0005 J0007
                        J0007 J0008 J0009 J0012 J0014 J0017 J0024 J0036 J0044
                        J0045 J0046 J0053 J0055 J0056 J0057 K0013 K0014 K0014
                        K0016 K0017 K0022 K0024 K0028 K0030 K0033 K0037 K0054
                        L0009 L0010 L0041 L0042 M0008 M0031 M0034 M0036 M0038
                        M0040 M0048 M0049 M0050 M0051 M0067 M0069 M0070 M0073
                        M0074 M0076 M0077 M0077 M0078 M0078 M0082 M0088 M0088
                        M0091 N0041 N0043 N0045 N0047 N0049 N0051 N0053 N0055
                        N0056 N0058 N0060 N0078 N0080 N0088 N0090 N0092 N0100
                        N0102 N0106 N0108 N0131 N0133 N0134 N0136 N0148 N0151
                        N0171 N0174 N0233 Q0030 Q0031 Q0032 Q0104 Q0107 Q0109
                        Q0112 Q0115 Q0116 Q0118 Q0129 Q0133 Q0135 R0005 R0011
                        R0016 S0006 S0020 S0022 S0026 S0036 S0043 S0080 S0083
                        S0086 S0089 S0091 S0111 S0115 S0117 S0119 S0123 T0006
                        T0014 T0017 T0019 T0026 U0009 U0031 U0033 U0035 U0144
                        U0145 U0146 U0202 U0231 U0232 V0043 V0044 V0048 V0053
                        V0061 V0062 V0063 V0064 V0086 V0146 V0148 V0150 V0218
                        V0222 V0226 V0288 V0299 V0300 V0305 V0308 V0310 V0311
                        V0317 V0318 V0320 V0322 V0325 V0326 V0329 V0337 V0339
                        V0341 W0007 W0009 W0011 W0014 W0015 W0016 W0017 W0017
                        W0022 W0022 W0024 W0024 W0025 W0028 W0033 W0038 W0039
                        W0053 W0054 W0056 W0058 W0067 W0071 W0086 W0098 W0100
                        W0103 W0108 W0123 W0137 W0162 W0182 W0184 W0187 W0192
                        W0208 W0211 W0213 W0239 W0245 W0245 W0288 W0313 W0330
                        W0331 W0333 W0337 W0339
R5          0005        C0025 D0070 D0074 D0076 D0077 D0078 D0079 D0080 D0081
                        D0082 D0083 D0089 D0093 E0009 E0010 E0012 E0021 E0021
                        E0023 E0039 E0040 E0043 E0044 E0045 E0045 E0046 E0105
                        E0106 E0135 E0136 E0225 E0226 E0227 E0235 E0235 E0236
                        E0237 E0237 L0171 L0176 M0081 M0085 M0090 M0094 N0034
                        N0145 N0185 N0186 00016 00072 00092 Q0023 Q0024 Q0025
                        Q0027 Q0037 Q0039 Q0043 Q0044 Q0045 Q0046 Q0077 Q0140
                        R0015 S0037 S0082 U0127 U0147 V0085 V0102 V0173 V0174
                        V0175 V0177 V0182 V0298 V0299 V0314 V0318 V0320 W0040
                        W0240 W0247
R6          0006        C0034 C0036 C0037 D0071 D0072 D0073 D0090 D0091 D0092
```

LABEL	VALUE	DEFN	REFERENCES
			E0082 E0088 E0112 E0113 E0198 E0199 E0200 E0201 E0202
			E0227 E0229 E0243 E0247 E0278 F0011 F0014 F0070 I0010
			I0018 I0030 I0083 I0089 I0097 J0038 K0018 K0019 K0023
			L0174 M0040 M0041 M0082 M0083 M0084 M0091 M0092 M0093
			N0078 N0094 N0098 N0108 N0111 N0115 N0116 N0118 N0134
			N0136 N0138 N0140 N0157 N0157 N0161 N0212 N0216 N0234
			Q0024 Q0051 Q0086 Q0087 Q0089 Q0092 Q0094 Q0095 Q0096
			Q0096 Q0122 Q0151 R0025 R0027 R0028 R0030 R0031 R0032
			R0033 R0034 R0035 R0036 R0037 R0038 S0091 S0092 S0094
			S0096 S0097 S0104 S0104 S0107 U0144 U0148 U0150 U0177
			U0178 U0187 U0191 U0191 U0193 U0197 U0231 U0239 V0095
			V0178 V0253 V0263 V0267 V0338 W0007 W0008 W0012 W0012
			W0018 W0020 W0020 W0031 W0053 W0062 W0073 W0074 W0076
			W0081 W0083 W0099 W0100 W0107 W0111 W0113 W0115 W0183
			W0184 W0188 W0193 W0215 W0217 W0231 W0233 W0238 W0251
			W0253 W0258 W0261 W0305 W0307 W0313 W0314 W0320
R7	0007		C0009 C0042 E0083 E0084 E0085 E0087 E0107 E0108 E0109
			E0110 E0203 E0277 F0012 F0015 F0068 F0069 H0017 I0011
			I0019 I0029 I0084 I0088 I0090 I0098 I0113 J0037 K0017
			K0022 M0036 M0037 M0038 M0039 M0067 M0068 M0069 M0070
			M0071 M0072 M0073 M0074 M0075 M0076 N0128 P0013 P0019
			Q0019 Q0020 Q0026 Q0027 Q0041 Q0047 Q0050 Q0064 R0013
			R0013 S0036 S0047 S0083 S0084 S0092 S0094 S0096 S0098
			S0098 S0102 U0145 U0189 U0232 U0245 V0065 V0088 V0173
			V0249 V0255 V0288 V0289 V0291 V0302 V0308 V0310 V0330
			V0330 V0334 W0005 W0039 W0051 W0055 W0056 W0057 W0058
			W0061 W0069 W0070 W0071 W0072 W0074 W0096 W0180 W0203
			W0284
R8	0008		N0211 N0215 N0232 N0233 N0237 Q0049 Q0062 Q0063 U0146
			U0162 U0175 U0176 U0185 U0186 U0187 U0188 U0189 U0202
			V0122 V0125 V0142 V0145 V0151 V0151 V0152 V0170 V0184
			V0294 V0297 V0304 V0321 V0322 V0325 V0328 V0329 V0332
			W0062 W0063 W0221 W0224 W0227 W0230 W0236 W0255 W0263
			W0270 W0292 W0295 W0298 W0301 W0312 W0318 W0319
R9	0009		D0065 D0066 D0092 L0041 L0042 L0043 L0060 L0075 L0092
			L0107 L0126 L0144 L0160 M0093 W0064
RAMBEG	E008	A0119	A0120 A0121 A0122 A0123 A0124 A0125 A0126 A0127 A0128
			A0129 A0130 A0131
RAMTST	0054	A0168	D0065
RBUSY	0200	A0152	D0061 U0113
RDATLN	0066	A0181	H0020 I0118 I0119 I0120 N0131 N0133 N0134 S0086 S0089
			S0091 S0117 W0239 W0245
RDB0	0A22'	L0060	L0057
RDB1	0A32'	L0075	L0072
RDB2	0A42'	L0092	L0089
RDB3	0A52'	L0107	L0104
RDBRT0	0A24'	L0061	L0059
RDBRT1	0A34'	L0076	L0074
RDBRT2	0A44'	L0093	L0091
RDBRT3	0A54'	L0108	L0106
RDBYT0	0A16'	L0056	N0096 N0109 S0045 U0243 V0251
RDBYT1	0A26'	L0071	E0356 V0097 V0180
RDBYT2	0A36'	L0088	E0144 E0154 E0181 E0186 E0255 E0258
RDBYT3	0A46'	L0103	E0176 E0178 E0222 E0244 E0246
READ	0684'	H0007	E0054
READ0	068E'	H0010	H0008
READ1	0694'	H0011	T0027
READ2	06D4'	H0027	H0022 H0025
READ3	06C0'	H0021	H0056
READOP	0EC9'	P0078	H0010 I0107

```
LABEL     VALUE   DEFN  REFERENCES

REC$1     16DE'   V0310 V0307
RECLEN    0068    A0182 F0020 F0028 F0083 I0038 I0099 I0111 I0113 I0120 N0088
                        N0090
RECNO     0060    A0178 E0275 F0065 G0009 H0015 H0048 I0013 I0050 I0086 I0108
                        J0007 J0044 K0014 M0049 N0047 N0049 V0062
RECNUM    0050    A0165 F0065 F0082 H0015 H0026 H0048 I0038 I0043 I0052 I0087
                        I0105 I0110 I0112 I0118 J0007 J0017 K0014 N0060 N0108
RELHSK    14F0'   U0261 U0194 U0204 U0242
RELOC     1B00'   W0330 B0097
RELOC2    1B1A'   W0339 W0334 W0336
RELOX     1B1E'   W0340 W0338
REN       001C'   B0025 N0038 O0090 Q0044 Q0080 S0040 U0043 U0046 U0125 U0165
                        U0171
RES9      113A'   R0027 R0029
RESBA1    10FE'   R0010 R0012 R0014 S0021 S0025
RESBAD    10FA'   R0009 S0008
RESEN1    1102'   R0011 R0006
RESEN2    1144'   R0030 R0015
RESEN3    1166'   R0039 S0027
RESENT    10EE'   R0005 E0316
RESENX    116A'   R0040 R0008 S0010
RESREN    13B2'   U0124 U0117
RESTOP    116E'   R0041 C0032 R0023
RET       067A'   G0017 F0081 F0100 F0130
REWIN1    08C0'   J0007 J0025
REWIN2    08EA'   J0018 J0013 J0016
REWIND    08B6'   J0005 E0056
RNAME     1956'   W0180 B0091
RS232     00B8'   B0119 B0042
RS2321    00BE'   B0121 B0046
RS2322    00C4'   B0123 B0050
RSENT     029C'   E0071 E0013
RSENT1    02AC'   E0075 E0072
RSENT2    02BE'   E0081 E0078
RSENT3    02C4'   E0082 E0080
RSENT4    02D8'   E0089 E0086
RST9      0146'   C0036 C0038
RSTIN1    0D90'   N0199 N0195
RSTIN2    0D98'   N0201 N0198
RSTINT    0D7A'   N0192 M0054 N0149 N0181
RW$PAB    179A'   W0025 W0019 W0023
RWDOP     0ECB'   P0080 J0005
RWSECT    1752'   W0005 B0085
S$1       16B8'   V0298 V0296
S$2       1704'   V0321 V0309 V0312 V0319
S$3       173A'   V0338 V0331
S$4       1720'   V0330 V0327
S14$15    1694'   V0287 W0205 W0286
SAV$FN    15F6'   V0170 F0064
SAV$FX    1618'   V0184 V0176
SAV$LP    160E'   V0180 V0183
SAVE      0728'   I0010 E0058
SAVE0     07A2'   I0042
SAVE01    0772'   I0029 I0027
SAVE1     07A6'   I0043 I0041
SAVE2     07BC'   I0049 I0046
SAVE3     07C2'   I0050 I0048
SAVED     0760'   I0025 I0021
SAVEN     0784'   I0034 I0023
SAVEOP    0ED7'   P0092 I0025 N0056 N0100
```

```
LABEL     VALUE   DEFN  REFERENCES

SAVEV     0792'   I0038 I0036
SAVEX     07EE'   I0062 I0024 I0044
SAVEY     07EA'   I0061 I0106
SBAV      0800    A0150 Q0052 S0076 U0040 U0075 U0084 U0115 U0122 U0168 U0237
SBR$EX    0D02'   N0145 N0070 N0170
SBR0      008A'   B0087 B0085
SBR1      0090'   B0089 B0087
SBR2      0096'   B0091 B0089
SBR3      009C'   B0093 B0091
SBR4      00A2'   B0095 B0093
SBR5      00A8'   B0097 B0095
SCNOFF    0054    A0167 D0064 D0077 K0037 K0054 M0077 N0232
SECT$X    17BC'   W0034 W0032
SENDZ     0A9E'   L0171 C0035 Q0152 R0026
SENDZ1    0AA2'   L0173 L0175
SENNUL    106C'   Q0140 Q0070
SHSK      0100    A0153 D0012 D0068 D0078 D0082 S0059 S0078 U0235
SIZERR    0AEA'   M0029 D0021 F0062 I0062 M0016 S0113 U0205
SKP$RS    177C'   W0017 W0013
SLAST     E073    A0131 U0128 U0259 U0262
SLBAVL    1336'   U0045 U0041
SLINTR    0F56'   Q0043 Q0038
SLOPEX    1342'   U0048 U0044 U0058
SLRD1     1426'   U0167 U0157 U0169
SLVBAD    12FA'   U0016 U0012
SLVCLS    134A'   U0056 U0018
SLVENT    12E6'   U0009 E0317
SLVILL    00FE    A0100 M0017
SLVOP2    131C'   U0037 U0034
SLVOP3    1322'   U0039 U0036
SLVOPN    1306'   U0031 U0017
SLVRD     13D8'   U0144 U0019
SLVRDO    140C'   U0159 U0173
SLVRD1    1408'   U0158 U0156 U0160
SLVRD2    143E'   U0174 U0163 U0181
SLVRD3    1454'   U0182 U0179
SLVRD4    1478'   U0195 U0198
SLVRD6    1490'   U0202 U0192
SLVRD7    1498'   U0204 U0190
SLVTAB    12FE'   U0017 U0014
SLVWR     14A0'   U0228 U0020
SLVWR1    14D4'   U0243 U0246
SLVWR2    14E4'   U0248 U0236 U0238
SLVWRX    14E0'   U0247
SPSWTB    048E'   E0312 E0225
SRQEND    0EE4'   P0107 Q0089
SRQTBL    0EDC'   P0103 Q0086
SRVREQ    E072    A0130 C0022 F0129 Q0072 R0038
STATHC    0ED9'   P0094 K0013 N0155
STATS1    09B2'   K0036 K0034
STATS2    09C0'   K0041 K0039
STATS3    09D0'   K0047 K0045
STATS4    09E2'   K0054 K0050 K0052
STATS5    09C8'   K0044 K0042
STATSX    09E6'   K0055 K0025 K0027
STATUN    0992'   K0028
STATUS    0958'   K0013 E0061
STFNOP    05A4'   F0064 F0063
STSOP     0ECD'   P0082 K0028 N0153
SUBLNK    0084'   B0085 B0011
```

```
LABEL     VALUE   DEFN  REFERENCES

SUR1      0A6A'   L0126 L0122
SUREAD    0A56'   L0120 E0138 N0035 N0107 S0044 U0240 V0096 V0179 V0250
SURRTN    0A6C'   L0127 L0125
SUW1      0A86'   L0144 L0140
SUWRIT    0A72'   L0138 N0129 S0081 U0149 W0237
SUWRTN    0A88'   L0145 L0143
SYOUT     1354'   U0073 U0244
SYOUT1    1364'   U0078 U0076
SYOUT2    136A'   U0080 U0085
SYOUT3    137E'   U0087 U0079 U0083
SYOUTE    1360'   U0077 U0086
SYTIN     1384'   U0108 U0116 U0180 U0183 U0195 U0200
SYTIN2    13A0'   U0118 U0114 U0123
SYTIN4    13C0'   U0128 U0111 U0121 U0161
SYTIN5    13D2'   U0133 U0129
TIMOUT    0AF0'   M0031 E0280 F0072 F0125 G0013 H0019 H0055 I0017 I0032 I0055
                  I0092 I0115 J0011 J0048 J0060 K0021 K0032 M0053 S0063
                  S0121
TMOUT1    0600    A0097 D0010
TMOUT4    0300    A0098 D0063 D0073 S0054
TOFRD2    11F8'   S0073 S0077 S0079
TRANEN    0EA9'   P0029 N0118
TRANS     0E9E'   P0028 N0115
TRAP1     026C'   E0038 E0029 E0033
TRFENT    1170'   S0006 E0315
TRFOPN    118C'   S0020
TRFRD     11FC'   S0074 S0012
TRFRD1    1246'   S0096 S0093
TRFRD2    129A'   S0122 S0073
TRFRD3    124A'   S0098 S0095
TRFRD4    125E'   S0106 S0108
TRFRD5    127A'   S0114 S0105
TRFRD6    1276'   S0113
TRFRD7    1296'   S0121 S0082
TRFRD8    124E'   S0100 S0103
TRFRD9    125A'   S0104 S0099
TRFWT     11A6'   S0036 S0014
TRFWT1    11CA'   S0045 S0048
TRFWT2    11F0'   S0063 S0037
TRFWT3    11DA'   S0055 S0062
TRFWT4    11F4'   S0064 S0058
TRFWT6    11D6'   S0054 S0060
TROPN1    11A0'   S0026 S0023
TTYBRK    08EE'   J0024 E0042
VDPRD     09FE'   L0026 L0013 L0124 W0265
VDPWD     09FA'   L0025 L0142 Q0120 Q0137 W0257 W0272
VERENT    046A'   E0296 E0314
VERFYX    0486'   E0305 E0298 E0300 E0302
VEROP     0ECF'   P0084 H0052 I0047
VERPOP    0EDA'   P0095 I0028 N0058 N0102
VRD       FBFE    A0095 D0073 L0058 L0073 L0090 L0105 W0266 W0268
VWA       8C02    A0094 A0095 A0096
VWD       FFFE    A0096 L0158 M0084 Q0121 Q0138 W0260 W0273 W0275
WBO       0A9A'   L0160 L0157
WBORTN    0A9C'   L0161 L0159
WBUSY     0400    A0151 D0036 U0078
WDATLN    0064    A0180 E0277 F0069 F0123 G0011 H0016 H0046 I0014 I0029 I0052
                  I0088 I0109 J0009 J0046 J0057 K0017 K0022 K0030 M0051
                  N0055 N0078 V0064 V0299 V0305 V0310 V0311 V0317 V0320
                  V0337 V0341 W0011 W0015 W0024 W0038 W0058 W0100 W0184
```

```
LABEL     VALUE   DEFN  REFERENCES
                        W0313
WEN       001A'   B0024 C0028 N0039 Q0081 Q0145 R0019 S0041 U0153 U0229
WRDIN     1670'   V0263 W0080 W0085 W0110 W0207 W0219 W0222 W0225 W0228
WRDOUT    167E'   V0273 V0333 V0336 W0029 W0291 W0294 W0297 W0300 W0303
WRITE     06D8'   H0036 E0055
WRITE0    06E2'   H0039 H0037
WRITE1    071A'   H0053 H0051
WRITOP    0ECA'   P0079 H0049 I0049
WRITR     06FA'   H0046 H0044
WRTVAR    06F6'   H0045 H0042
WS$2$3    1794'   W0024 W0021
WSECOP    0ED4'   P0089 W0009
WTBYTO    0A8E'   L0156 N0160 S0087 S0090 S0101 S0112 S0116 U0174 U0184 U0196
                        U0201
WTPR01    18F8'   W0144 W0141
WTPROT    18F2'   W0142 W0139
XMIT3     0C52'   N0078 W0041
XT3$RT    17DA'   W0042 W0040
XVTH      165C'   V0249 V0340 W0109 W0189 W0194 W0315 W0321
XVTHLP    1662'   V0251 V0254
```