```
                    00002
                    00003   ********************************************************************************
                    00004   *
                    00005   *       BOOT67
                    00006   *
                    00007   *       This is the boot program that resides on the Master Boot Record
                    00008   *       of the H67 ("Winchester") drive.   Its purpose is to display a
                    00009   *       menu of up to 16 possible boot options and have the user select
                    00010   *       which one to boot.  It then loads the OS bootstrap off that partition.
                    00011   *
                    00012   *       Disassembled and commented by Glenn Roberts, August 2011
                    00013   *
  042.200           00014           XTEXT   HOSEQU
                    00015X


                    00017X  **      HDOS SYSTEM EQUIVALENCES.
                    00018X  *
                    00019X
  024.000           00020X  S.GRT0  EQU     24000A          SYSTEM AREA FOR  GRT0
  025.000           00021X  S.GRT1  EQU     25000A          SYSTEM AREA FOR  GRT1
  026.000           00022X  S.GRT2  EQU     26000A          SYSTEM AREA FOR  GRT2
                    00023X
  030.000           00024X  ROMBOOT EQU     30000A          ROM BOOT ENTRY
                    00025X
  040.100           00026X          ORG     40100A          FREE SPACE FROM PAM-8
                    00027X
  040.100           00028X          DS      8               JUMP TO SYSTEM EXIT
  040.110           00029X  D.CON   DS      16              DISK CONSTANTS
  040.130           00030X  SYDD    EQU     *               SYSTEM DISK ENTRY POINT
  040.130           00031X  D.VEC   DS      24*3            SYSTEM ROM ENTRY VECTORS
  040.240           00032X  D.RAM   DS      31              SYSTEM ROM WORK AREA
  040.277           00033X  S.VAL   DS      36              SYSTEM VALUES
  040.343           00034X  S.INT   DS      115             SYSTEM INTERNAL WORK AREAS
  041.126           00035X          DS      16
  041.146           00036X  S.SOVR  DS      2               STACK OVERFLOW WARNING
  041.150           00037X          DS      42200A-*        SYSTEM STACK
  001.032           00038X  STACKL  EQU     *-S.SOVR        STACK SIZE
                    00039X
  042.200           00040X  STACK   EQU     *               LWA+1 SYSTEM STACK
  042.200           00041X  USERFWA EQU     *               USER FWA
  042.200           00042           XTEXT   DIRDEF
                    00043X


                    00045X  **      DIRECTORY ENTRY FORMAT.
                    00046X
  000.000           00047X          ORG     0
                    00048X
                    00049X
  000.377           00050X  DF.EMP  EQU     377Q            FLAGS ENTRY EMPTY
  000.376           00051X  DF.CLR  EQU     376Q            FLAGS ENTRY EMPTY, REST OF DIR ALSO CLEAR
                    00052X
  000.000           00053X  DIR.NAM DS      8               NAME
```

```
000.010             00054X  DIR.EXT DS      3               EXTENSION
000.013             00055X  DIR.PRO DS      1               PROJECT
000.014             00056X  DIR.VER DS      1               VERSION
000.015             00057X  DIRIDL  EQU     *               FILE IDENTIFICATION LENGTH
                    00058X
000.015             00059X  DIR.CLU DS      1               CLUSTER FACTOR
000.016             00060X  DIR.FLG DS      1               FLAGS
000.017             00061X          DS      1               RESERVED
000.020             00062X  DIR.FGN DS      1               FIRST GROUP NUMBER
000.021             00063X  DIR.LGN DS      1               LAST GROUP NUMBER
000.022             00064X  DIR.LSI DS      1               LAST SECTOR INDEX (IN LAST GROUP)
000.023             00065X  DIR.CRD DS      2               CREATION DATE
000.025             00066X  DIR.ALD DS      2               LAST ALTERATION DATE
                    00067X
000.027             00068X  DIRELEN EQU     *               DIRECTORY ENTRY LENGTH
000.027             00069           XTEXT   MTR
                    00070X
```

```
                    00073X  **      MTR - PAM/8 EQUIVALENCES.
                    00074X  *
                    00075X  *       THIS DECK CONTAINS SYMBOLIC DEFINITIONS USED TO
                    00076X  *       MAKE USE OF THE PAM/8 CODE AND CONTROL BYTES.



                    00078X  **      IO PORTS
                    00079X
  000.360           00080X  IP.PAD  EQU     360Q            PAD INPUT PORT
  000.360           00081X  OP.CTL  EQU     360Q            CONTROL OUTPUT PORT
  000.360           00082X  OP.DIG  EQU     360Q            DIGIT SELECT OUTPUT PORT
  000.361           00083X  OP.SEG  EQU     361Q            SEGMENT SELECT OUTPUT PORT
  000.362           00084X  IP.CON  EQU     362Q            H-88/H-89/HA-8-8 Configuration  /80.07.gc/
  000.362           00085X  OP2.CTL EQU     362Q            H-88/H-89/HA-8-8 Control Port   /80.07.gc/



                    00087X  **      FRONT PANEL CONTROL BITS.                               /80.07.gc/
                    00088X  *
                    00089X  *       CB.*  set in OP.CTL
                    00090X  *       CB2.* set in OP2.CTL
                    00091X  *
                    00092X
  000.020           00093X  CB.SSI  EQU     00010000B       SINGLE STEP INTERRUPT
  000.040           00094X  CB.MTL  EQU     00100000B       MONITOR LIGHT
  000.100           00095X  CB.CLI  EQU     01000000B       CLOCK INTERRUPT ENABLE
  000.200           00096X  CB.SPK  EQU     10000000B       SPEAKER ENABLE
                    00097X
  000.001           00098X  CB2.SSI EQU     00000001B       Single Step Interrupt
  000.002           00099X  CB2.CLI EQU     00000010B       Clock Interrupt Enable
  000.040           00100X  CB2.ORG EQU     00100000B       ORG 0 Select
  000.100           00101X  CB2.SID EQU     01000000B       Side 1 Select



                    00103X  **      Secondary Control Bits
                    00104X



                    00106X  **      MONITOR MODE FLAGS.
                    00107X
  000.000           00108X  DM.MR   EQU     0               MEMORY READ
  000.001           00109X  DM.MW   EQU     1               MEMORY WRITE
  000.002           00110X  DM.RR   EQU     2               REGISTER READ
  000.003           00111X  DM.RW   EQU     3               REGISTER WRITE



                    00113X  **      USER OPTION BITS.
                    00114X  *
                    00115X  *       THESE BITS ARE SET IN CELL .MFLAG.
                    00116X
  000.200           00117X  UO.HLT  EQU     10000000B       DISABLE HALT PROCESSING
  000.100           00118X  UO.NFR  EQU     CB.CLI          NO REFRESH OF FRONT PANEL
  000.002           00119X  UO.DDU  EQU     00000010B       DISABLE DISPLAY UPDATE
```

```
   000.001             00120X  UO.CLK  EQU     00000001B       ALLOW PRIVATE INTERRUPT PROCESSING



                       00122X  **      MONITOR IDENTIFICATION FLAGS
                       00123X  *
                       00124X  *       THESE BYTES IDENTIFY THE ROM MONITOR.
                       00125X  *       THEY ARE THE VARIOUS VALUES OF LOCATION .IDENT
                       00126X
   000.021             00127X  M.PAM8  EQU     021Q            'LXI' INSTRUCTION AT 000.000 IN PAM-8
   000.303             00128X  M.FOX   EQU     303Q            'JMP' INSTRUCTION AT 000.000 IN FOX ROM



                       00130X  **      Configuration Flags                                     /80.07.gc/
                       00131X  *
                       00132X  *       These bits are read in IP.CON.
                       00133X  *
                       00134X
   000.003             00135X  CN.174M EQU     00000011B       Port 174Q Device-Type Mask
   000.014             00136X  CN.170M EQU     00001100B       Port 170Q Device-Type Mask
   000.020             00137X  CN.PRI  EQU     00010000B       Primary/Secondary:  1=>primary == 170Q
   000.040             00138X  CN.MEM  EQU     00100000B       Memory Test/Normal Switch:  0=>Test; 1=>Normal
   000.100             00139X  CN.BAU  EQU     01000000B       Baud Rate:  0=>9600; 1=>19,200
   000.200             00140X  CN.ABO  EQU     10000000B       Auto-Boot:  1=>Auto-Boot
                       00141X
   000.000             00142X  CND.H17 EQU     00B             H-17 Disk,            Valid only in CN.174M
   000.000             00143X  CND.NDI EQU     00B             No Device Installed,  Valid only in CN.170M
   000.001             00144X  CND.H47 EQU     01B             H-47 Disk



                       00146X  **      ROUTINE ENTRY POINTS.
                       00147X  *
                       00148X
   000.000             00149X  .IDENT  EQU     0000A           IDENTIFICATION LOCATION
   000.053             00150X  .DLY    EQU     0053A           DELAY
   001.267             00151X  .LOAD   EQU     1267A           TAPE LOAD
   001.374             00152X  .DUMP   EQU     1374A           TAPE DUMP
   002.136             00153X  .ALARM  EQU     2136A           ALARM ROUTINE
   002.140             00154X  .HORN   EQU     2140A           HORN
   002.172             00155X  .CTC    EQU     2172A           CHECK TAPE CHECKSUM
   002.205             00156X  .TPERR  EQU     2205A           TAPE ERROR ROUTINE
   002.264             00157X  .PCHL   EQU     2264A           PCHL INSTRUCTION
   002.265             00158X  .SRS    EQU     2265A           SCAN RECORD START
   002.325             00159X  .RNP    EQU     2325A           READ NEXT PAIR
   002.331             00160X  .RNB    EQU     2331A           READ NEXT BYTE
   002.347             00161X  .CRC    EQU     2347A           CRC-16 CALCULATOR
   003.017             00162X  .WNP    EQU     3017A           WRITE NEXT PAIR
   003.024             00163X  .WNB    EQU     3024A           WRITE NEXT BYTE
   003.122             00164X  .DOD    EQU     3122A           DECODE FOR OCTAL DISPLAY
   003.260             00165X  .RCK    EQU     3260A           READ CONSOLE KEYSET
   003.356             00166X  .DODA   EQU     3356A           SEGMENT CODE TABLE
```

```
                    00168X  **      RAM CELLS USED BY H8MTR.
                    00169X  *
                    00170X
   040.000          00171X  .START  EQU     40000A          START DUMP ADDRESS
   040.002          00172X  .IOWRK  EQU     40002A          IN OR OUT INSTRUCTION
   040.005          00173X  .REGI   EQU     40005A          DISPLAYED REGISTER INDEX
   040.006          00174X  .DSPROT EQU     40006A          PERIOD FLAG BYTE
   040.007          00175X  .DSPMOD EQU     40007A          DISPLAY MODE
   040.010          00176X  .MFLAG  EQU     40010A          USER OPTION BYTE
   040.011          00177X  .CTLFLG EQU     40011A          PANEL CONTROL BYTE
   040.013          00178X  .ALEDS  EQU     40013A          ABUSS LEDS
   040.021          00179X  .DLEDS  EQU     40021A          DBUSS LEDS
   040.024          00180X  .ABUSS  EQU     40024A          ABUSS REGISTER
   040.027          00181X  .CRCSUM EQU     40027A          CRCSUM WORD
   040.031          00182X  .TPERRX EQU     40031A          TAPE ERROR EXIT VECTOR
   040.033          00183X  .TICCNT EQU     40033A          CLOCK TICK COUNTER
   040.035          00184X  .REGPTR EQU     40035A          REGISTER POINTER
   040.037          00185X  .UIVEC  EQU     40037A          USER INTERRUPT VECTORS
   040.064          00186X  .NMIRET EQU     40064A          H88/H89 NMI Return Address      /80.07.gc/
   040.066          00187X  .CTL2FL EQU     40066A          OP2.CTL Control Byte            /80.07.gc/
   000.027          00188           XTEXT   H67DEF
                    00189X  **      H67 Disk Controller Definitions
                    00190X  *




                    00192X  **      Register addresses
                    00193X  *
   000.170          00194X  BASE    EQU     170Q            Controller base address
                    00195X
   000.000          00196X  RI.DAT  EQU     0               Data In/Out (Read/Write)
   000.001          00197X  RI.CON  EQU     1               Control (Write Only)
   000.001          00198X  RI.BST  EQU     1               Bus Status (Read Only)




                    00200X  *       Control Register Definition
                    00201X
   000.100          00202X  BC.SEL  EQU     01000000B       Select and data bit 0
   000.040          00203X  BC.IE   EQU     00100000B       Interrupt Enable
   000.020          00204X  BC.RST  EQU     00010000B       Reset
   000.002          00205X  BC.EDT  EQU     00000010B       Enable Data




                    00207X  *       Bus Status Register Definition
                    00208X
   000.200          00209X  BS.REQ  EQU     10000000B       Bus Transfer Request
   000.100          00210X  BS.DTD  EQU     01000000B       Data Transfer Direction
   000.000          00211X  BS.IN   EQU     00000000B        Data to Host
   000.100          00212X  BS.OUT  EQU     01000000B        Data to Controller
   000.040          00213X  BS.LMB  EQU     00100000B       Last byte in data/command string
   000.020          00214X  BS.MTY  EQU     00010000B       Message type
   000.000          00215X  BS.DAT  EQU     00000000B        Data
```

```
   000.020              00216X  BS.COM  EQU     00010000B        Command
   000.010              00217X  BS.BSY  EQU     00001000B       Busy
   000.004              00218X  BS.INT  EQU     00000100B       Interrupt Pending
   000.002              00219X  BS.PE   EQU     00000010B       Parity Error
   000.001              00220X  BS.HID  EQU     00000001B       Hardware Identification



                        00222X  *       Status Byte Definitions
                        00223X
   000.140              00224X  ST.LUN  EQU     01100000B       Logical Unit
   000.034              00225X  ST.SPR  EQU     00011100B       Spare
   000.002              00226X  ST.ERR  EQU     00000010B       Error
   000.001              00227X  ST.PER  EQU     00000001B       Parity Error



                        00229X  **      Commands
                        00230X  *
                        00231X
   000.340              00232X  CLASSM  EQU     11100000B       Class Mask
                        00233X
   000.000              00234X  CLASS0  EQU     00000000B       Class 0
   000.040              00235X  CLASS1  EQU     00100000B       Class 1
   000.300              00236X  CLASS6  EQU     11000000B       Class 6
                        00237X
   000.037              00238X  OPCODM  EQU     00011111B       Op-code Mask
   000.140              00239X  LUNM    EQU     01100000B       Logical Unit Mask
   000.037              00240X  LSA.2   EQU     00011111B       Logical Sector Address (2)



                        00242X  *       Class 0 Commands
                        00243X
   000.000              00244X  D.TDR   EQU     CLASS0+0        Test drive ready
   000.001              00245X  D.REC   EQU     CLASS0+1        Recalibrate drive
   000.002              00246X  D.RSY   EQU     CLASS0+2        Request Syndrome
   000.003              00247X  D.RSE   EQU     CLASS0+3        Request Sense
   000.004              00248X  D.FOR   EQU     CLASS0+4        Format Drive
   000.005              00249X  D.CTF   EQU     CLASS0+5        Check track format
   000.006              00250X  D.FT    EQU     CLASS0+6        Format Track
   000.007              00251X  D.FBS   EQU     CLASS0+7        Format bad sector
   000.010              00252X  D.REA   EQU     CLASS0+8        Read
   000.011              00253X  D.WPS   EQU     CLASS0+9        Write protect the sector
   000.012              00254X  D.WRI   EQU     CLASS0+10       Write
   000.013              00255X  D.SEK   EQU     CLASS0+11       Seek
```

```
                    00257X  *       Class 1 Commands
                    00258X
000.040             00259X  D.CP3   EQU     CLASS1+0        Copy block



                    00261X  *       Class 6 Commands
                    00262X
000.300             00263X  D.FFD   EQU     CLASS6+0        Format floppy disk



                    00265X  *       Type 0 error codes (Drive error Codes)
                    00266X
000.000             00267X  T0.NST  EQU     0               No status
000.001             00268X  T0.NIS  EQU     1               No Index signal
000.002             00269X  T0.NSC  EQU     2               No seek complete
000.003             00270X  T0.WFT  EQU     3               Write fault
000.004             00271X  T0.DNR  EQU     4               Drive not ready
000.005             00272X  T0.DNS  EQU     5               Drive not selected
000.006             00273X  T0.NT0  EQU     6               No track zero
000.007             00274X  TO.MDS  EQU     7               Mult-drive selected



                    00276X  *       Type 1 error codes (data error codes)
                    00277X
000.000             00278X  T1.ID   EQU     0               ID Read Error
000.001             00279X  T1.UDE  EQU     1               Uncorrectable data error
000.002             00280X  T1.IDNF EQU     2               ID Address Mark not found
000.003             00281X  T1.DMNF EQU     3               Data Address Mark Not FOund
000.004             00282X  T1.RNF  EQU     4               Record Not Found
000.005             00283X  T1.SKE  EQU     5               Seek Error
000.006             00284X  T1.DTE  EQU     6               DMA Time-out Error (not used)
000.007             00285X  T1.WP   EQU     7               Write protected
000.010             00286X  T1.CDE  EQU     8               Correctable Data field Error
000.011             00287X  T1.BBF  EQU     9               Bad Block Found
000.012             00288X  T1.FE   EQU     10              Format Error



                    00290X  *       Type 2 Error Codes (Command error codes)
                    00291X
000.000             00292X  T2.ILC  EQU     0               Illegal Command
000.001             00293X  T2.IDA  EQU     1               Illegal Disk Address
000.002             00294X  T2.IFN  EQU     2               Illegal Function
000.027             00295           XTEXT   ASCII
                    00296X
```

```
                    00298X  **      ASCII CHARACTER EQUIVALENCES.
                    00299X
   000.015          00300X  CR      EQU     13              CARRIAGE RETURN
   000.012          00301X  LF      EQU     10              LINE FEED
   000.200          00302X  NULL    EQU     200Q            PAD CHARACTER
   000.000          00303X  NUL2    EQU     0
   000.007          00304X  BELL    EQU     7               BELL CHARACTER
   000.177          00305X  RUBOUT  EQU     177Q
   000.010          00306X  BKSP    EQU     10Q             CTL-H
   000.026          00307X  C.SYN   EQU     26Q             SYNC
   000.002          00308X  C.STX   EQU     2               STX
   000.047          00309X  QUOTE   EQU     47Q
   000.011          00310X  TAB     EQU     11Q
   000.033          00311X  ESC     EQU     33Q
   000.012          00312X  NL      EQU     12Q             NEW LINE (HDOS SYSTEMS)
   000.212          00313X  ENL     EQU     NL+200Q         NL + END-OF-LINE-FLAG
   000.014          00314X  FF      EQU     14Q             FORM FEED
   000.001          00315X  CTLA    EQU     01Q             CTL-A
   000.002          00316X  CTLB    EQU     02Q             CTL-B
   000.003          00317X  CTLC    EQU     03Q             CTL-C
   000.004          00318X  CTLD    EQU     04Q             CTL-D
   000.017          00319X  CTLO    EQU     17Q             CTL-O
   000.020          00320X  CTLP    EQU     20Q             CTL-P
   000.021          00321X  CTLQ    EQU     21Q             CTL-Q
   000.023          00322X  CTLS    EQU     23Q             CTL-S
   000.032          00323X  CTLZ    EQU     32Q             CTL-Z
   000.027          00324           XTEXT   U8250
                    00325X



                    00327X  **      8250 UART CONTROL AND BIT DEFINITIONS.
                    00328X
   000.350          00329X  SC.ACE  EQU     350Q            SYSTEM CONSOLE PORT IF 8250 ACE
   000.156          00330X  AC.DLY  EQU     110             220 MIL. SEC. DELAY FOR 8250
                    00331X
   000.000          00332X  UR.RBR  EQU     0               RECEIVER BUFFER REGISTER (READ ONLY)
                    00333X
   000.000          00334X  UR.THR  EQU     0               TRANSMITTER HOLDING REGISTER (WRITE ONLY)
                    00335X
   000.000          00336X  UR.DLL  EQU     0               DIVISOR LATCH (LEAST SIGNIFICANT)
                    00337X
   000.001          00338X  UR.DLM  EQU     1               DIVISOR LATCH (MOST SIGNIFICANT)
                    00339X
   000.001          00340X  UR.IER  EQU     1               INTERRUPT ENABLE REGISTER
   000.001          00341X  UC.EDA  EQU     00000001B        ENABLE RECEIVED DATA AVAILABLE INTERRUPT
   000.002          00342X  UC.TRE  EQU     00000010B        ENABLE TRANSMIT HOLD REGISTER EMPTY INTERRUPT
   000.004          00343X  UC.RSI  EQU     00000100B        ENABLE RECEIVE STATUS INTERRUPT
   000.010          00344X  UC.MSI  EQU     00001000B        ENABLE MODEM STATUS INTERRUPT
                    00345X
   000.002          00346X  UR.IIR  EQU     2               INTERRUPT IDENTIFICATION REGISTER
   000.001          00347X  UC.IIP  EQU     00000001B        INVERTED INTERRUPT PENDING (0 MEANS PENDING)
   000.006          00348X  UC.IID  EQU     00000110B        INTERRUPT ID
                    00349X
   000.003          00350X  UR.LCR  EQU     3               LINE CONTROL REGISTER
   000.000          00351X  UC.5BW  EQU     00000000B        5 BIT WORDS
```

```
   000.001              00352X  UC.6BW  EQU     00000001B       6 BIT WORDS
   000.002              00353X  UC.7BW  EQU     00000010B       7 BIT WORDS
   000.003              00354X  UC.8BW  EQU     00000011B       8 BIT WORDS
   000.004              00355X  UC.2SB  EQU     00000100B       TWO STOP BITS SELECTED
   000.010              00356X  UC.PEN  EQU     00001000B       PARITY COMPUTATION ENABLED
   000.020              00357X  UC.EPS  EQU     00010000B       EVEN PARITY SELECT
   000.040              00358X  UC.SKP  EQU     00100000B       STICK PARITY
   000.100              00359X  UC.SB   EQU     01000000B       SET BREAK
   000.200              00360X  UC.DLA  EQU     10000000B       DIVISOR LATCH ACCESS
                        00361X
   000.004              00362X  UR.MCR  EQU     4               MODEM CONTROL REGISTER
   000.001              00363X  UC.DTR  EQU     00000001B       DATA TERMINAL READY
   000.002              00364X  UC.RTS  EQU     00000010B       REQUEST TO SEND
   000.004              00365X  UC.OU1  EQU     00000100B       OUT 1
   000.010              00366X  UC.OU2  EQU     00001000B       OUT 2
   000.020              00367X  UC.LOO  EQU     00010000B       LOOP
                        00368X
   000.005              00369X  UR.LSR  EQU     5               LINE STATUS REGISTER
   000.001              00370X  UC.DR   EQU     00000001B       DATA READY
   000.002              00371X  UC.OR   EQU     00000010B       OVERRUN
   000.004              00372X  UC.PE   EQU     00000100B       PARITY ERROR
   000.010              00373X  UC.FE   EQU     00001000B       FRAMING ERROR
   000.020              00374X  UC.BI   EQU     00010000B       BREAK INTERRUPT
   000.040              00375X  UC.THE  EQU     00100000B       TRANSMITTER HOLDING REGISTER EMPTY
   000.100              00376X  UC.TSE  EQU     01000000B       TRANSMITTER SHIFT REGISTER EMPTY
                        00377X
   000.006              00378X  UR.MSR  EQU     6               MODEM STATUS REGISTER
   000.001              00379X  UC.DCS  EQU     00000001B       DELTA CLEAR TO SEND
   000.002              00380X  UC.DDR  EQU     00000010B       DELTA DATA SET READY
   000.004              00381X  UC.TER  EQU     00000100B       TRAILING EDGE OF RING
   000.010              00382X  UC.DRL  EQU     00001000B       DELTA RECEIVE LINE SIGNAL DETECT
   000.020              00383X  UC.CTS  EQU     00010000B       CLEAR TO SEND
   000.040              00384X  UC.DSR  EQU     00100000B       DATA SET READY
   000.100              00385X  UC.RI   EQU     01000000B       RING INDICATOR
   000.200              00386X  UC.RLS  EQU     10000000B       RECEIVED LINE SIGNAL DETECT
   000.027              00387           XTEXT   U8251
                        00388X
```

```
                    00391X  **      8251 USART BIT DEFINITIONS.
                    00392X  *
                    00393X
                    00394X  **      PORT ADDRESSES
                    00395X
  000.000           00396X  UDR     EQU     0               DATA REGISTER IS EVEN
  000.001           00397X  USR     EQU     1               STATUS REGISTER IS NEXT
                    00398X
  000.372           00399X  SC.UART EQU     372Q            CONSOLE USART ADDRESS (IFF 8251)
                    00400X
                    00401X
                    00402X  **      MODE INSTRUCTION CONTROL BITS.
                    00403X
  000.100           00404X  UMI.1B  EQU     01000000B       1 STOP BIT
  000.200           00405X  UMI.HB  EQU     10000000B       1 1/2 STOP BITS
  000.300           00406X  UMI.2B  EQU     11000000B       2 STOP BITS
  000.040           00407X  UMI.PE  EQU     00100000B       EVEN PARITY
  000.020           00408X  UMI.PA  EQU     00010000B       USE PARITY
  000.000           00409X  UMI.L5  EQU     00000000B       5 BIT CHARACTERS
  000.004           00410X  UMI.L6  EQU     00000100B       6 BIT CHARACTERS
  000.010           00411X  UMI.L7  EQU     00001000B       7 BIT CHARACTERS
  000.014           00412X  UMI.L8  EQU     00001100B       8 BIT CHARACTERS
  000.001           00413X  UMI.1X  EQU     00000001B       CLOCK X 1
  000.002           00414X  UMI.16X EQU     00000010B       CLOCK X 16
  000.003           00415X  UMI.64X EQU     00000011B       CLOCK X 64
                    00416X
                    00417X  **      COMMAND INSTRUCTION BITS.
                    00418X
  000.100           00419X  UCI.IR  EQU     01000000B       INTERNAL RESET
  000.040           00420X  UCI.RO  EQU     00100000B       READER-ON CONTROL FLAG
  000.020           00421X  UCI.ER  EQU     00010000B       ERROR RESET
  000.004           00422X  UCI.RE  EQU     00000100B       RECEIVE ENABLE
  000.002           00423X  UCI.IE  EQU     00000010B       ENABLE INTERRUPTS FLAG
  000.001           00424X  UCI.TE  EQU     00000001B       TRANSMIT ENABLE
                    00425X
                    00426X  **      STATUS READ COMMAND BITS.
                    00427X
  000.100           00428X  USR.BD  EQU     01000000B       Break Detect                    /80.08.gc/
  000.040           00429X  USR.FE  EQU     00100000B       FRAMING ERROR
  000.020           00430X  USR.OE  EQU     00010000B       OVERRUN ERROR
  000.010           00431X  USR.PE  EQU     00001000B       PARITY ERROR
  000.004           00432X  USR.TXE EQU     00000100B       TRANSMITTER EMPTY
  000.002           00433X  USR.RXR EQU     00000010B       RECEIVER READY
  000.001           00434X  USR.TXR EQU     00000001B       TRANSMITTER READY
  000.027           00435           XTEXT   ESVAL
                    00436X



                    00438X  **      S.VAL - SYSTEM VALUE DEFINTIONS.
                    00439X  *
                    00440X  *       THESE VALUES ARE SET AND MAINTAINED BY THE SYSTEM.
                    00441X  *
                    00442X  *       THE DECK HOSEQU MUST BE MODIFIED WHEN THIS IS MODIFIED.
                    00443X
                    00444X
```

```
   040.277              00445X          ORG     S.VAL
                        00446X
   040.277              00447X  S.DATE  DS      9               SYSTEM DATE (IN ASCII)
   040.310              00448X  S.DATC  DS      2               CODED DATE
   040.312              00449X  S.TIME  DS      4               TIME FROM MIDNIGHT (IN TICS)
   040.316              00450X  S.HIMEM DS      2               HARDWARE HIGH MEMORY ADRESS+1
                        00451X
   040.320              00452X  S.SYSM  DS      2               FWA RESIDENT SYSTEM
                        00453X
   040.322              00454X  S.USRM  DS      2               LWA USER MEMORY
                        00455X
   040.324              00456X  S.OMAX  DS      2               MAX OVERLAY SIZE FOR SYSTEM
                        00457X
                        00458X
                        00459X  **      THE FOLLOWING FIVE CELLS SHOULD BE MODIFIED/READ ONLY VIA THE .CONSL SYSCALL
                        00460X
   000.200              00461X  CSL.ECH EQU     10000000B       SUPPRESS ECHO
   000.004              00462X  CSL.RAW EQU     00000100B       Raw Mode I/O                    /80.09.gc/
   000.002              00463X  CSL.WRP EQU     00000010B       WRAP LINES AT WIDTH
   000.001              00464X  CSL.CHR EQU     00000001B       OPERATE IN CHARACTER MODE
                        00465X
   000.000              00466X  I.CSLMD EQU     0               S.CSLMD IS FIRST BYTE
   040.326              00467X  S.CSLMD DS      1               CONSOLE MODE
                        00468X
   000.200              00469X  CTP.BKS EQU     10000000B       TERMINAL PROCESSES BACKSPACES
   000.100              00470X  CTP.FF  EQU     01000000B       Terminal Processes Form-Feed    /80.09.gc/
   000.040              00471X  CTP.MLI EQU     00100000B       MAP LOWER CASE TO UPPER ON INPUT
   000.020              00472X  CTP.MLO EQU     00010000B       MAP LOWER CASE TO UPPER ON OUTPUT
   000.010              00473X  CTP.2SB EQU     00001000B       TERMINAL NEEDS TWO STOP BITS
   000.002              00474X  CTP.BKM EQU     00000010B       MAP BKSP (UPON INPUT) TO RUBOUT
   000.001              00475X  CTP.TAB EQU     00000001B       TERMINAL SUPPORTS TAB CHARACTERS
                        00476X
   000.001              00477X  I.CONTY EQU     1               S.CONTY IS 2ND BYTE
   000.000              00478X          ERRNZ   *-S.CSLMD-I.CONTY
   040.327              00479X  S.CONTY DS      1               CONSOLE TYPE FLAGS
   000.002              00480X  I.CUSOR EQU     2               S.CUSOR IS 3RD BYTE
   000.000              00481X          ERRNZ   *-S.CSLMD-I.CUSOR
   040.330              00482X  S.CUSOR DS      1               CURRENT CURSOR POSITION
   000.003              00483X  I.CONWI EQU     3               S.CONWI IS 4TH BYTE
   000.000              00484X          ERRNZ   *-S.CSLMD-I.CONWI
   040.331              00485X  S.CONWI DS      1               CONSOLE WIDTH
                        00486X
   000.001              00487X  CO.FLG  EQU     00000001B       CTL-O FLAG
   000.200              00488X  CS.FLG  EQU     10000000B       CTL-S FLAG
                        00489X
   000.004              00490X  I.CONFL EQU     4               S.CONFL IS 5TH BYTE
   000.000              00491X          ERRNZ   *-S.CSLMD-I.CONFL
   040.332              00492X  S.CONFL DS      1               CONSOLE FLAGS
                        00493X
   040.333              00494X  S.CAADR DS      2               ADDRESS FOR ABORT PROCESSING (>256 IF VALID)
   040.335              00495X  S.CCTAB DS      6               ADDR FOR CTL-A, CTL-B, CTL-C PROCESSING
   040.343              00496           XTEXT   ESINT
```

```
                    00498X  **      S.INT - SYSTEM INTERNAL WORKAREA DEFINITIONS.
                    00499X  *
                    00500X  *       THESE CELLS ARE REFERENCED BY OVERLAYS AND MAIN CODE, AND
                    00501X  *       MUST THEREFORE RESIDE IN FIXED LOW MEMORY.
                    00502X
                    00503X
  040.343           00504X          ORG     S.INT
                    00505X
                    00506X  **      CONSOLE STATUS FLAGS
                    00507X
  040.343           00508X  S.CDB   DS      1               CONSOLE DESCRIPTOR BYTE
  000.000           00509X  CDB.H85 EQU     00000000B
  000.001           00510X  CDB.H84 EQU     00000001B       =0 IF H8-5, =1 IF H8-4
  040.344           00511X  S.BAUD  DS      2               [0-14]  H8-4 BAUD RATE, =0 IF H8-5
                    00512X  *                               [15]    =1 IF BAUD RATE => 2 STOP BITS
                    00513X
                    00514X  **      TABLE ADDRESS WORDS
                    00515X
  040.346           00516X  S.DLINK DS      2               ADDRESS OF DATA IN HDOS CODE
  040.350           00517X  S.OFWA  DS      2               FWA  OVERLAY  TABLE
  040.352           00518X  S.CFWA  DS      2               FWA  CHANNEL  TABLE
  040.354           00519X  S.DFWA  DS      2               FWA  DEVICE   TABLE
  040.356           00520X  S.RFWA  DS      2               FWA  RESIDENT HDOS CODE
                    00521X
                    00522X  **      DEVICE DRIVER DELAYED LOAD FLAGS
                    00523X
  040.360           00524X  S.DDLDA DS      2               DRIVER LOAD ADDRESS (HIGH BYTE=0 IF NO LOAD PENDING)
  040.362           00525X  S.DDLEN DS      2               CODE LENGTH IN BYTES
  040.364           00526X  S.DDGRP DS      1               GROUP NUMBER FOR DRIVER
  040.365           00527X          DS      1               HOLD PLACE
                    00528X  *S.DDSEC        DS      2               SECTOR NUMBER FOR DRIVER ( * OBSOLETE ! * )
  040.366           00529X  S.DDDTA DS      2               DEVICE'S ADDRESS IN DEVLST +DEV.RES
  040.370           00530X  S.DDOPC DS      1               OPEN OPCODE PENDEDING
                    00531X
                    00532X  **      OVERLAY MANAGEMENT FLAGS
                    00533X
  000.001           00534X  OVL.IN  EQU     00000001B       IN MEMORY
  000.002           00535X  OVL.RES EQU     00000010B       PERMINANTLY RESIDENT
  000.014           00536X  OVL.NUM EQU     00001100B       OVERLAY NUMBER MASK
  000.200           00537X  OVL.UCS EQU     10000000B       USER CODE SWAPPED FOR OVERLAY
                    00538X
  040.371           00539X  S.OVLFL DS      1               OVERLAY FLAG
  040.372           00540X  S.UCSF  DS      2               FWA SWAPPED USER CODE
  040.374           00541X  S.UCSL  DS      2               LENGTH SWAPPED USER CODE
  040.376           00542X  S.OVLS  DS      2               SIZE OF OVERLAY CODE
  041.000           00543X  S.OVLE  DS      2               ENTRY POINT OF OVERLAY CODE
                    00544X
  041.002           00545X  S.SSN   DS      2               SWAP AREA SECTOR NUMBER
  041.004           00546X  S.OSN   DS      2               OVERLAY SECTOR NUMBER
                    00547X
                    00548X  *       SYSCALL PROCESSING WORK AREAS
                    00549X
  041.006           00550X  S.CACC  DS      1               (ACC) UPON SYSCALL
  041.007           00551X  S.CODE  DS      1               SYSCALL INDEX IN PROGRESS
                    00552X
                    00553X  *       JUMPS TO ROUTINES IN RESIDENT HDOS CODE
                    00554X
```

```
041.010             00555X  S.JUMPS DS      0               START OF DUMP VECTORS
041.010             00556X  S.SDD   DS      3               JUMP TO STAND-IN DEVICE DRIVER
041.013             00557X  S.FASER DS      3               JUMP TO FATSERR (FATAL SYSTEM ERROR)
041.016             00558X  S.DIREA DS      3               JUMP TO DIREAD (DISK FILE READ)
041.021             00559X  S.FCI   DS      3               JUMP TO FCI (FETCH CHANNEL INFO)
041.024             00560X  S.SCI   DS      3               JUMP TO SCI (STORE CHANNEL INFO)
041.027             00561X  S.GUP   DS      3               JUMP TO GUP (GET UNIT POINTER)
                    00562X
041.032             00563X  S.MOUNT DS      1               <>0 IF THE SYSTEM DISK IS MOUNTED
041.033             00564X  S.DCS   DS      1               DEFAULT CLUSTER SIZE-1
                    00565X
041.034             00566X  S.BOOTF DS      1               BOOT FLAGS
000.001             00567X  BOOT.P  EQU     00000001B       EXECUTE PROLOGUE UPON BOOTUP
                    00568X
                    00569X  *       STACK VALUE SAVED FOR OVERLAY SYSCALLS
                    00570X
041.035             00571X  S.OVSTK DS      2               VALUE OF SP UPON SYSCALLS USING OVERLAY
                    00572X
041.037             00573X          DS      1               RESERVED



                    00575X  **      ACTIVE I/O AREA.
                    00576X  *
                    00577X  *       THE AIO.XXX AREA CONTAINS INFORMATION ABOUT THE I/O OPERATION
                    00578X  *       CURRENTLY BEING PERFORMED. THE INFORMATION IS OBTAINED FROM
                    00579X  *       THE CHANNEL TABLE, AND WILL BE RESTORED THERE WHEN DONE.
                    00580X  *
                    00581X  *       NORMALLY, THE AIO.XXX INFORMATION WOULD BE OBTAINED DIRECTLY
                    00582X  *       FROM VARIOUS SYSTEM TABLES VIA POINTER REGISTERS. SINCE THE
                    00583X  *       8080 HAS NO GOOD INDEXED ADDRESSING, THE DATA IS MANUALLY
                    00584X  *       COPIED INTO THE AIO.XXX CELLS BEFORE PROCESSING, AND
                    00585X  *       BACKDATED AFTER PROCESSING.
                    00586X
041.040             00587X  AIO.VEC DS      3               JUMP INSTRUCTION
041.041             00588X  AIO.DDA EQU     *-2             DEVICE DRIVER ADDRESS
041.043             00589X  AIO.FLG DS      1               FLAG BYTE
041.044             00590X  AIO.GRT DS      2               ADDRESS OF GROUP RESERV TABLE
041.046             00591X  AIO.SPG DS      1               SECTORS PER GROUP
041.047             00592X  AIO.CGN DS      1               CURRENT GROUP NUMBER
041.050             00593X  AIO.CSI DS      1               CURRENT SECTOR INDEX
041.051             00594X  AIO.LGN DS      1               LAST GROUP NUMBER
041.052             00595X  AIO.LSI DS      1               LAST SECTOR INDEX
041.053             00596X  AIO.DTA DS      2               DEVICE TABLE ADDRESS
041.055             00597X  AIO.DES DS      2               DIRECTORY SECTOR
041.057             00598X  AIO.DEV DS      2               DEVICE CODE
041.061             00599X  AIO.UNI DS      1               UNIT NUMBER (0-9)
                    00600X
041.062             00601X  AIO.DIR DS      DIRELEN         DIRECTORY ENTRY
                    00602X
041.111             00603X  AIO.CNT DS      1               SECTOR COUNT
041.112             00604X  AIO.EOM DS      1               END OF MEDIA FLAG
041.113             00605X  AIO.EOF DS      1               END OF FILE FLAG
041.114             00606X  AIO.TFP DS      2               TEMP FILE POINTERS
041.116             00607X  AIO.CHA DS      2               ADDRESS OF CHANNEL BLOCK (IOC.DDA)
```

```
   041.120              00609X  S.BDA   DS      1               Boot Device Address (Setup by ROM) /80.09.gc/
   041.121              00610X  S.SCR   DS      2               SYSTEM SCRATCH AREA ADDRESS
   041.123              00611           XTEXT   UDD



                        00613X  **      $UDD - UNPACK DECIMAL DIGITS.
                        00614X  *
                        00615X  *       UDD CONVERTS A 16 BIT VALUE INTO A SPECIFIED NUMBER OF
                        00616X  *       DECIMAL DIGITS. THE RESULT IS ZERO FILLED.
                        00617X  *
                        00618X  *       ENTRY   (B,C) = ADDRESS VALUE
                        00619X  *               (A) = DIGIT COUNT
                        00620X  *               (H,L) = MEMORY ADDRESS
                        00621X  *       EXIT    (HL) = (HL) + (A)
                        00622X  *       USES    ALL
                        00623X
                        00624X
   031.157              00625X  $UDD    EQU     31157A          IN H17 ROM
                        00626
   011.000              00627   BOOLEN  EQU     9*256                   Length of boot code
   053.200              00628   EXORG   EQU     USERFWA+BOOLEN          load point for Extended SBC
   000.014              00629   BAU.96  EQU     000014A                 BAUD divisor for 9600
   000.000              00630   WARMB   EQU     0                       Jump to 0 for warm boot
   001.000              00631   SBCXLN  EQU     256                     Length or Extended SBC
   262.355              00632   MI.INIR EQU     262355A                 Z80 "INIR" instruction
                        00633   *
                        00634   *       SYSINT storage
                        00635   *
                        00636   *       These storage areas follow the Active I/O (AIO.xxx) definitions in the
                        00637   *       system internals (ESINT) include file.  They are only set by this code
                        00638   *       and appear to be here to provide pointers for other utilities to use.
                        00639   *
   041.126              00640   S.OSID  EQU     041126A                 OS ID no. (in OS table)
   041.127              00641   S.OCCR  EQU     041127A                 Occurrence no. (partition)
   041.130              00642   S.BLOW  EQU     041130A                 OS boot sector (low)
   041.131              00643   S.BMID  EQU     041131A                 (middle)
   041.132              00644   S.BHI   EQU     041132A                 (high)
   041.133              00645   S.ALOW  EQU     041133A                 Next OS boot sector (low)
   041.134              00646   S.AMID  EQU     041134A                 (middle)
   041.135              00647   S.AHI   EQU     041135A                 (high)
                        00648
   042.200              00649           ORG     USERFWA
                        00650
   042.200  303 000 043 00651   ENTRY   JMP     RELOC                   Relocate the code
                        00652
                        00654   *
                        00655   *       Software Boot Code (SBC)
                        00656   *
                        00657   *       When the user attempts to boot a partition of the Winchester Disk, the
                        00658   *       Monitor 90 boot ROM (see the MTR90 manual) loads the SBC. The SBC will
                        00659   *       then perform secondary boot of the hard disk.
                        00660   *
                        00661   *       The first 128 bytes of sector 0 contain critical information on the
                        00662   *       structure of the disk.
                        00663   *
```

```
   042.203  001          00664           DB      1                       Major version (v 1.1)
   042.204  001          00665           DB      1                       Minor version
   042.205  040 040 040 00666   DFLBOO   DB      '                   '   Default boot string (19)
   042.230  024 000 000 00667           DB      20,0,0                  Beginning sector # of bad sector table A
   042.233  170 000 000 00668           DB      120,0,0                 Beginning sector # of bad sector table B
   042.236  050 000 000 00669   SSBA    DB      40,0,0                  Beginning sector # of superblock A
   042.241  120 000 000 00670   SSBB    DB      80,0,0                  Beginning sector # of superblock B
   042.244  000 001      00671           DW      256                     Sector size
   042.246  050 000      00672           DW      40                      Sectors per track
   042.250  004 000      00673           DW      4                       Tracks per cylinder
   042.252  364 000      00674           DW      244                     Cylinders per volume
   042.254  240 000      00675           DW      160                     Sectors per region
   042.256  200 230      00676           DW      39040                   Number of sectors (cyl/vol * sectors/rgn)
   042.260  000          00677           DB      0                       (byte 3 of # sectors)
   042.261  363          00678           DB      243                     Number of regions (0 based)
   042.262  345 277      00679   CS.SBA  DW      277345A                 Checksum for superblock copy A
   042.264  345 277      00680   CS.SBB  DW      277345A                 Checksum for superblock copy B
   042.266  000 000      00681           DW      0                       Checksum for bad sector table copy A
   042.270  000 000      00682           DW      0                       Checksum for bad sector table copy B
   042.272  001 377      00683           DW      377001A                 ?
                         00684
   042.274  000 000 000 00685           DB      0,0,0,0,0,0,0,0,0,0     Reserved
   042.306  000 000 000 00686           DB      0,0,0,0,0,0,0,0,0,0
   042.320  000 000 000 00687           DB      0,0,0,0,0,0,0,0,0,0
   042.332  000 000 000 00688           DB      0,0,0,0,0,0,0,0,0,0
   042.344  000 000 000 00689           DB      0,0,0,0,0,0,0,0,0,0
   042.356  000 000 000 00690           DB      0,0,0,0,0,0,0,0,0,0
   042.370  000 000 000 00691           DB      0,0,0,0,0,0,0,0
                         00692



                         00695   *
                         00696   *       Entry point
                         00697   *
                         00698   *       Step 1: relocate the code (last to first) to make room for
                         00699   *       loading the boot code from the selected partition.  Essentiall
                         00700   *       need room for two boot loads - this one plus the OS-specific one.
                         00701   *
   043.000  001 001 011 00702   RELOC   LXI     B,BOOLEN+1      byte count
   043.003  041 201 064 00703           LXI     H,USERFWA+BOOLEN+BOOLEN+1       To: leave room for TWO boot loads)
   043.006  021 053 054 00704           LXI     D,ENDREL+BOOLEN From: skip over relocation code
   043.011  032          00705   RELOC1  LDAX    D               fetch a byte
   043.012  053          00706           DCX     H               decrement To
   043.013  033          00707           DCX     D               decrement From
   043.014  167          00708           MOV     M,A             store it
   043.015  013          00709           DCX     B               loop 'til count is zero
   043.016  170          00710           MOV     A,B
   043.017  261          00711           ORA     C
   043.020  302 011 043 00712           JNZ     RELOC1          Exits with (HL) pointing to START!
                         00713
   043.023  072 120 041 00714           LDA     S.BDA           Boot Device Address (set by ROM)
   043.026  267          00715           ORA     A               if not set then skip
   043.027  312 052 043 00716           JZ      RELOC2          otherwise overwrite port numbers...
                         00717
   043.032  062 304 056 00718           STA     INPORT1         save port numbers...
```

```
   043.035  062 047 057 00719           STA     INPORT2         data port...
   043.040  062 060 057 00720           STA     OUPORT1
   043.043  074         00721           INR     A               increment (command port)
   043.044  062 052 057 00722           STA     INPORT3
   043.047  062 055 057 00723           STA     OUPORT2
   043.052  351         00724   RELOC2  PCHL                    now jump to START!
                        00725
   043.053               00726   ENDREL  EQU     *               End of relocation routine
                        00727   *
                        00728   *       The following code is relocated up by BOOLEN to make
                        00729   *       room for loading the OS boot code off the partition
                        00730   *




                        00733   *
                        00734   *       Step 2: initialize console and controller
                        00735   *
   053.200               00736           ORG     USERFWA+BOOLEN
                        00737
   053.200  041 000 000 00738   START   LXI     H,0
   053.203  071         00739           DAD     SP              HL = SP
   053.204  042 006 065 00740           SHLD    SPSAVE          Store SP
   053.207  061 250 072 00741           LXI     SP,EXT245       set up our own stack
                        00742
   053.212  315 245 060 00743           CALL    FCU             set up console
   053.215  315 137 057 00744           CALL    INITC           initialize controller
                        00745




                        00748   *
                        00749   *       Step 3: Load SBC extension
                        00750   *
   053.220  001 000 001 00751           LXI     B,SBCXLN        Extended SBC length
   053.223  170         00752           MOV     A,B             A = 0
   053.224  261         00753           ORA     C
   053.225  312 335 053 00754           JZ      GETSBA          if 0 skip
                        00755
   053.230  170         00756           MOV     A,B
   053.231  007         00757           RLC
   053.232  332 335 053 00758           JC      GETSBA
                        00759   *
                        00760   *       Attempt to read extended SBC
                        00761   *
   053.235  041 142 056 00762           LXI     H,EXTSBC        HL = sector address of extended SBC
   053.240  021 325 063 00763           LXI     D,EXORG+EXORG-ENDREL    where to load it
   053.243  315 145 056 00764           CALL    DOREAD          read it!
   053.246  322 335 053 00765           JNC     GETSBA          success, next step...
                        00766   *
                        00767   *       Unable to load extended SBC, abort.
                        00768   *
   053.251  315 126 060 00769           CALL    $TYPET
   053.254  007         00770           DB      BELL
   053.255  000 106 101 00771           DB      0,'FATAL ERROR - EXTENDED SBC LOAD ABORTED.',200Q
```

```
   053.327  315 272 061 00772          CALL    WAITCR          prompt user
   053.332  303 000 000 00773          JMP     WARMB           warm boot...



                       00776   *
                       00777   *       Step 4: Attempt to read superblock A
                       00778   *
                       00779   *       (Superblock is three 256-byte tables)
                       00780   *
   053.335  041 236 042 00781   GETSBA  LXI     H,SSBA          sector address on disk
   053.340  021 104 065 00782           LXI     D,SUPBLK
   053.343  001 000 003 00783           LXI     B,256*3
   053.346  315 145 056 00784           CALL    DOREAD          read it!
   053.351  322 031 054 00785           JNC     CHKSBA          success, next step
                       00786   *
                       00787   *       Error reading superblock A
                       00788   *
   053.354  315 126 060 00789           CALL    $TYPET
   053.357  007         00790           DB      BELL
   053.360  000 105 122 00791           DB      0,'ERROR - CANNOT READ SUPERBLOCK A.',200Q
   054.023  315 272 061 00792           CALL    WAITCR
   054.026  303 056 054 00793           JMP     GETSBB          try superblock B...
                       00794   *
                       00795   *       verify checksum for Superblock A
                       00796   *
   054.031  021 000 000 00797   CHKSBA  LXI     D,0
   054.034  001 000 003 00798           LXI     B,256*3
   054.037  041 104 065 00799           LXI     H,SUPBLK
   054.042  315 014 062 00800           CALL    $BCRC
   054.045  052 262 042 00801           LHLD    CS.SBA
   054.050  315 154 062 00802           CALL    HLCPDE
   054.053  312 210 054 00803           JZ      GETBS           checksum OK, proceed
                       00804   *
                       00805   *       Step 4a: Read or checksum failed on superblock A...
                       00806   *
                       00807   *       Attempt to read superblock B
                       00808   *
   054.056  041 241 042 00809   GETSBB  LXI     H,SSBB
   054.061  021 104 065 00810           LXI     D,SUPBLK
   054.064  001 000 003 00811           LXI     B,256*3
   054.067  315 145 056 00812           CALL    DOREAD
   054.072  322 100 054 00813           JNC     CHKSBB          OK, next step
   054.075  303 125 054 00814           JMP     SBERR           else fatal error
                       00815   *
                       00816   *       verify checksum?
                       00817   *
   054.100  021 000 000 00818   CHKSBB  LXI     D,0
   054.103  001 000 003 00819           LXI     B,256*3
   054.106  041 104 065 00820           LXI     H,SUPBLK
   054.111  315 014 062 00821           CALL    $BCRC
   054.114  052 264 042 00822           LHLD    CS.SBB
   054.117  315 154 062 00823           CALL    HLCPDE
   054.122  312 210 054 00824           JZ      GETBS           checksum OK, proceed
                       00825   *
                       00826   *       Error both superblock A and B failed, abort!
```

```
                        00827   *
   054.125  315 126 060 00828   SBERR   CALL    $TYPET
   054.130  007         00829           DB      BELL
   054.131  000 106 101 00830           DB      0,'FATAL ERROR - CANNOT READ SUPERBLOCK B.',200Q
   054.202  315 272 061 00831           CALL    WAITCR
   054.205  303 000 000 00832           JMP     WARMB           fatal error, warm boot



                        00835   *
                        00836   *       Step 5: Look for default Boot String, otherwise prompt user
                        00837   *
   054.210  052 006 065 00838   GETBS   LHLD    SPSAVE
   054.213  021 200 042 00839           LXI     D,USERFWA
   054.216  315 154 062 00840           CALL    HLCPDE
   054.221  312 243 054 00841           JZ      GETBS1          OK, continue!
                        00842
   054.224  315 052 064 00843           CALL    SKIP0           Bump HL to point to next NULL
   054.227  052 006 065 00844           LHLD    SPSAVE
                        00845
   054.232  315 063 061 00846           CALL    PARSEB          parse boot string
   054.235  312 146 055 00847           JZ      OSLOOK          parse OK, look it up now
   054.240  303 066 055 00848           JMP     ERRBS           else ERROR in boot string
                        00849
   054.243  016 000     00850   GETBS1  MVI     C,0             count up...
   054.245  006 023     00851           MVI     B,19            count down...
   054.247  041 205 042 00852           LXI     H,DFLBOO        Default boot
   054.252  176         00853   GETBS2  MOV     A,M             fetch a char
   054.253  376 040     00854           CPI     ' '
   054.255  312 266 054 00855           JZ      GETBS3          break out on ' '
   054.260  014         00856           INR     C               bump counters and pointer
   054.261  005         00857           DCR     B
   054.262  043         00858           INX     H
   054.263  302 252 054 00859           JNZ     GETBS2          and loop
                        00860   *
                        00861   *       C = length of boot string
                        00862   *
   054.266  171         00863   GETBS3  MOV     A,C
   054.267  267         00864           ORA     A
   054.270  312 373 054 00865           JZ      ASKBS           Zero length (no default) - prompt user for Boot string...
                        00866   *
                        00867   *       Have a default boot string
                        00868   *
   054.273  006 000     00869           MVI     B,0
   054.275  041 205 042 00870           LXI     H,DFLBOO        Default boot
   054.300  315 063 061 00871           CALL    PARSEB          parse boot string
   054.303  312 146 055 00872           JZ      OSLOOK          OK, look it up
                        00873
   054.306  315 126 060 00874           CALL    $TYPET
   054.311  007         00875           DB      BELL
   054.312  000 105 122 00876           DB      0,'ERROR - Syntax error in Default Boot String.',200Q
   054.370  315 272 061 00877           CALL    WAITCR
                        00878   *
                        00879   *       Step 5a: Ask user for boot string
                        00880   *
   054.373  315 210 062 00881   ASKBS   CALL    DOMENU
```

```
   054.376  315 126 060 00882           CALL    $TYPET
   055.001  000 000 000 00883           DB      0,0,0,'Boot String?.............>',' '+200Q
                        00884   *
                        00885   *       Read boot string from console
                        00886   *
   055.037  041 325 064 00887           LXI     H,INPSTR
   055.042  076 023     00888           MVI     A,19            Boot string can be up to
   055.044  315 162 056 00889           CALL    RDSTR           19 characters
   055.047  072 326 064 00890           LDA     STRLEN          how many were read?
   055.052  117         00891           MOV     C,A
   055.053  006 000     00892           MVI     B,0             BC = string length
   055.055  041 327 064 00893           LXI     H,STRTXT        HL = string text
   055.060  315 063 061 00894           CALL    PARSEB          parse it
   055.063  312 146 055 00895           JZ      OSLOOK
                        00896   *
                        00897   *       Syntax error in boot string
                        00898   *
   055.066  315 126 060 00899   ERRBS   CALL    $TYPET
   055.071  007         00900           DB      BELL
   055.072  000 105 162 00901           DB      0,'Error - Syntax error in Boot String.',200Q
   055.140  315 272 061 00902           CALL    WAITCR
   055.143  303 373 054 00903           JMP     ASKBS           Go back and ask again for boot string




                        00906   *
                        00907   *       Step 6: look for OSNAME in the OS ID Table
                        00908   *
                        00909   *       EXIT:   B = index of matching entry in OS ID table
                        00910   *
   055.146  041 104 065 00911   OSLOOK  LXI     H,SB.IDT        Point to OS ID table
   055.151  006 001     00912           MVI     B,1             count = 1
   055.153  305         00913   OSLK1   PUSH    B
   055.154  345         00914           PUSH    H
   055.155  001 020 000 00915           LXI     B,16            string length
   055.160  021 365 064 00916           LXI     D,OSNAME
   055.163  315 006 064 00917           CALL    STRCMP          compare the two strings
   055.166  341         00918           POP     H
   055.167  301         00919           POP     B
   055.170  312 265 055 00920           JZ      HAVEOS          found it! next step...
   055.173  004         00921           INR     B
   055.174  021 020 000 00922           LXI     D,16            offset to next entry
   055.177  031         00923           DAD     D               HL = HL + 16
   055.200  170         00924           MOV     A,B
   055.201  376 021     00925           CPI     16+1            done them all?
   055.203  302 153 055 00926           JNZ     OSLK1           no, keep looking
                        00927   *
                        00928   *       no match to specified OS
                        00929   *
   055.206  315 126 060 00930           CALL    $TYPET
   055.211  007         00931           DB      BELL
   055.212  000 105 162 00932           DB      0,'Error - Operating System not found.',200Q
   055.257  315 272 061 00933           CALL    WAITCR
   055.262  303 373 054 00934           JMP     ASKBS           go back and ask again for Boot string...
```

```
                        00937   *       Step 7: find the desired occurrence of this OS
                        00938   *
                        00939   *               (HL) points to entry
                        00940   *
   055.265  072 005 065 00941   HAVEOS  LDA     OCCNUM          occurrence number
   055.270  315 353 063 00942           CALL    FINDO           find specified occurrence
   055.273  312 346 055 00943           JZ      HAVEOC          OK, got it...
                        00944   *
                        00945   *       error no such partition
                        00946   *
   055.276  315 126 060 00947           CALL    $TYPET
   055.301  007         00948           DB      BELL
   055.302  000 105 162 00949           DB      0,'Error - Partition not found.',200Q
   055.340  315 272 061 00950           CALL    WAITCR
   055.343  303 373 054 00951           JMP     ASKBS           go back and ask again for Boot string...



                        00954   *
                        00955   *       Step 8: Fill in system internals area with key data.
                        00956   *
   055.346  170         00957   HAVEOC  MOV     A,B
   055.347  062 126 041 00958           STA     S.OSID          OS ID
   055.352  072 005 065 00959           LDA     OCCNUM          user partition #
   055.355  062 127 041 00960           STA     S.OCCR          occurrence number
   055.360  176         00961           MOV     A,M
   055.361  062 130 041 00962           STA     S.BLOW          low byte of sector no.
   055.364  043         00963           INX     H               next entry
   055.365  176         00964           MOV     A,M
   055.366  062 131 041 00965           STA     S.BMID          middle byte of sector no.
   055.371  043         00966           INX     H               next entry
   055.372  176         00967           MOV     A,M
   055.373  062 132 041 00968           STA     S.BHI           high byte of sector no.
   055.376  043         00969           INX     H               skip ID byte 1
   055.377  043         00970           INX     H               next
   056.000  176         00971           MOV     A,M
   056.001  062 133 041 00972           STA     S.ALOW          low byte of NEXT entry
   056.004  043         00973           INX     H               next
   056.005  176         00974           MOV     A,M
   056.006  062 134 041 00975           STA     S.AMID          middle byte
   056.011  043         00976           INX     H               next
   056.012  176         00977           MOV     A,M
   056.013  062 135 041 00978           STA     S.AHI           high byte
   056.016  053         00979           DCX     H
   056.017  053         00980           DCX     H
   056.020  053         00981           DCX     H
   056.021  053         00982           DCX     H
   056.022  053         00983           DCX     H
   056.023  053         00984           DCX     H               restore HL to low sector byte
```

```
                        00987   *
                        00988   *       Step 9: read in the boot code
                        00989   *
   056.024  001 000 011 00990           LXI     B,BOOLEN        how many bytes to read
   056.027  021 200 042 00991           LXI     D,USERFWA       where to load it
   056.032  315 145 056 00992           CALL    DOREAD          load the boot code from partition
   056.035  322 134 056 00993           JNC     HAVEO1
                        00994   *
                        00995   *       Problem reading boot code?
                        00996   *
   056.040  315 126 060 00997           CALL    $TYPET
   056.043  007         00998           DB      BELL
   056.044  000 105 162 00999           DB      0,'Error - Unable to read Boot Code from Partition.',200Q
   056.126  315 272 061 01000           CALL    WAITCR
   056.131  303 373 054 01001           JMP     ASKBS           Go back and ask again for Boot string...
                        01002   *
                        01003   *       Now jump to the boot code!
                        01004   *
   056.134  061 200 042 01005   HAVEO1  LXI     SP,USERFWA
   056.137  303 200 042 01006           JMP     USERFWA
                        01007
   056.142  011 000 000 01008   EXTSBC  DB      9,0,0           sector id of extended SBC




                        01011   ***     DOREAD - read from disk
                        01012   *
                        01013   *       ENTRY:  BC = byte count
                        01014   *               DE = destination in memory
                        01015   *               HL = sector address (3 bytes: low, mid, high)
                        01016   *
   056.145  325         01017   DOREAD  PUSH    D
   056.146  136         01018           MOV     E,M             E = low
   056.147  043         01019           INX     H
   056.150  126         01020           MOV     D,M             D = mid
   056.151  353         01021           XCHG                    DE <=> HL (HL = low, mid)
   056.152  321         01022           POP     D               DE = destination
   056.153  315 242 056 01023           CALL    EXT133
   056.156  315 315 057 01024           CALL    UNK67
   056.161  311         01025           RET




                        01028   *
                        01029   *       RDSTR - Read a string from the console. Exit on
                        01030   *               Carriage Return.
                        01031   *
                        01032   *       ENTRY:  (HL) = address of string structure
                        01033   *               A = maximum string length
                        01034   *
                        01035   *       String format:
                        01036   *               BYTE:   max
                        01037   *               BYTE:   len
                        01038   *               CHAR[]: string
                        01039   *
```

```
   056.162  167          01040   RDSTR   MOV     M,A             store max
   056.163  043          01041           INX     H               point to len
   056.164  066 000      01042           MVI     M,0             set to 0
                          01043
   056.166  345          01044           PUSH    H
   056.167  321          01045           POP     D
   056.170  023          01046           INX     D               (DE) = first string character
   056.171  053          01047           DCX     H               (HL) = max
                          01048
   056.172  345          01049   RDSTR1  PUSH    H
   056.173  325          01050           PUSH    D
   056.174  315 044 060  01051           CALL    $ICTT           read a character from the console
   056.177  315 163 060  01052           CALL    $TYPEC.         echo it
   056.202  321          01053           POP     D
   056.203  341          01054           POP     H
   056.204  376 015      01055           CPI     CR              Carriage Return?
   056.206  312 234 056  01056           JZ      RDSTR3          Yes - done!
   056.211  376 040      01057           CPI     ' '             control character?
   056.213  322 222 056  01058           JNC     RDSTR2          no
   056.216  043          01059           INX     H               point to len
   056.217  066 000      01060           MVI     M,0             set len to 0!
   056.221  311          01061           RET                     and exit!
                          01062
   056.222  022          01063   RDSTR2  STAX    D               store the character
   056.223  023          01064           INX     D               point to next location
   056.224  043          01065           INX     H               point to counter
   056.225  064          01066           INR     M               increment it
   056.226  176          01067           MOV     A,M             get it
   056.227  053          01068           DCX     H               point to max
   056.230  276          01069           CMP     M               count = max?
   056.231  302 172 056  01070           JNZ     RDSTR1          no, keep reading
   056.234  311          01071   RDSTR3  RET                     exit!
                          01072   *
                          01073   *       Data area
                          01074   *
   056.235  000          01075   S.LUN   DB      0               LUN = 0
   056.236  000          01076   SS.CDB  DB      0               Console descriptor 1 = 8250, 0 = 8251
   056.237  000          01077   X.CONTY DB      0               UART config
   056.240  000 000      01078   X.BAUD  DW      0               BAUD
                          01079   *
                          01080   *       BC = byte count
                          01081   *       DE = destination
                          01082   *       HL = sector address
                          01083   *
   056.242  315 051 057  01084   EXT133  CALL    RDSTAT
   056.245  346 216      01085           ANI     BS.REQ+BS.BSY+BS.INT+BS.PE
   056.247  067          01086           STC                     prepare for possible error return
   056.250  300          01087           RNZ                     return if error
                          01088
   056.251  305          01089           PUSH    B
   056.252  315 102 057  01090           CALL    UNK59
   056.255  076 010      01091           MVI     A,D.REA         Read
   056.257  315 227 057  01092           CALL    BLDCDR
   056.262  301          01093           POP     B
   056.263  315 156 057  01094           CALL    GETCON          get controller's attention and send command byte
   056.266  315 365 056  01095           CALL    DOCMD           complete the command
   056.271  353          01096           XCHG                    HL <=> DE
```

```
   056.272  121          01097           MOV     D,C
   056.273  130          01098           MOV     E,B
   056.274  315 303 056  01099           CALL    UNK49
   056.277  103          01100           MOV     B,E
   056.300  112          01101           MOV     C,D
   056.301  353          01102           XCHG
   056.302  311          01103           RET
                         01104   *
                         01105   *       Process read of data
                         01106   *
                         01107   *       ENTRY:  HL = buffer to read
                         01108   *               DE = byte count?
                         01109   *
   056.303  016 170      01110   UNK49   MVI     C,BASE
   056.304               01111   INPORT1 EQU     *-1             port no
                         01112
   056.305  173          01113           MOV     A,E
   056.306  247          01114           ANA     A
   056.307  312 324 056  01115           JZ      UNK49B
   056.312  315 351 056  01116   UNK49A  CALL    RDHALF          read half
   056.315  315 351 056  01117           CALL    RDHALF          other half...
   056.320  035          01118           DCR     E
   056.321  302 312 056  01119           JNZ     UNK49A
   056.324  172          01120   UNK49B  MOV     A,D
   056.325  247          01121           ANA     A
   056.326  310          01122           RZ
   056.327  376 201      01123           CPI     129
   056.331  332 342 056  01124           JC      UNK49C          jump if 0..128
   056.334  315 351 056  01125           CALL    RDHALF
   056.337  172          01126           MOV     A,D
   056.340  326 200      01127           SUI     128
   056.342  107          01128   UNK49C  MOV     B,A
   056.343  315 353 056  01129           CALL    EXT140
   056.346  026 000      01130           MVI     D,0
   056.350  311          01131           RET
                         01132   *
                         01133   *       RDHALF - read half a sector (128 bytes)
                         01134   *
   056.351  006 200      01135   RDHALF  MVI     B,128
   056.353  315 117 057  01136   EXT140  CALL    WAITR           read a byte
   056.356  167          01137           MOV     M,A             store it
   056.357  043          01138           INX     H               up pointer
   056.360  005          01139           DCR     B               down counter
   056.361  310          01140           RZ
                         01141   *
                         01142   *       Z80 "INIR": input to memory from port (C); loop 'til B=0
                         01143   *
   056.362  355 262      01144           DW      MI.INIR
   056.364  311          01145           RET
                         01146   *
                         01147   *       DOCMD - complete sending the command block
                         01148   *               (send last 5 bytes of CDR block)
                         01149   *
   056.365  305          01150   DOCMD   PUSH    B
   056.366  345          01151           PUSH    H
   056.367  315 375 056  01152           CALL    DOCMD1
   056.372  341          01153           POP     H
```

```
   056.373  301          01154           POP     B
   056.374  311          01155           RET
   056.375  041 015 057  01156   DOCMD1  LXI     H,CDRBLK+1
   057.000  016 005      01157           MVI     C,5
   057.002  176          01158   DOCMD2  MOV     A,M
   057.003  315 373 057  01159           CALL    SENDB           send a byte
   057.006  043          01160           INX     H
   057.007  015          01161           DCR     C
   057.010  302 002 057  01162           JNZ     DOCMD2
   057.013  311          01163           RET
                         01164   *
                         01165   *       Command Descriptor Block (Class 0 commands)
                         01166   *
   057.014  136          01167   CDRBLK  DB      94              000 | opcode
   057.015  000          01168           DB      0               LUN | log addr2
   057.016  000          01169           DB      0               log addr 1
   057.017  000          01170           DB      0               log addr 0
   057.020  000          01171           DB      0               num blocks
   057.021  000          01172           DB      0               control
                         01173
                         01174   *
                         01175   *       READ2 = Read two bytes from data port
                         01176   *
                         01177   *       EXIT:   B = first byte
                         01178   *               C = second byte
                         01179   *
   057.022  363          01180   READ2   DI
   057.023  315 062 057  01181           CALL    RDWAIT          Wait for data available, then read it
   057.026  107          01182           MOV     B,A             save in B
   057.027  315 051 057  01183   READ2A  CALL    RDSTAT          read status port
   057.032  346 360      01184           ANI     BS.REQ+BS.OUT+BS.LMB+BS.COM
   057.034  376 260      01185           CPI     BS.REQ+BS.LMB+BS.COM    Wait for no OUT
   057.036  302 027 057  01186           JNZ     READ2A          no, loop...
   057.041  315 046 057  01187           CALL    RDDATA          read from data port
   057.044  117          01188           MOV     C,A             save in C
   057.045  311          01189           RET
                         01190   *
                         01191   *       Controller I/O routines
                         01192   *
   057.046  333 170      01193   RDDATA  IN      BASE+RI.DAT
   057.047               01194   INPORT2 EQU     *-1
   057.050  311          01195           RET
                         01196
   057.051  333 171      01197   RDSTAT  IN      BASE+RI.BST
   057.052               01198   INPORT3 EQU     *-1
   057.053  311          01199           RET
                         01200
   057.054  323 171      01201   WRCTRL  OUT     BASE+RI.CON
   057.055               01202   OUPORT2 EQU     *-1
   057.056  311          01203           RET
                         01204
   057.057  323 170      01205   WRDATA  OUT     BASE+RI.DAT
   057.060               01206   OUPORT1 EQU     *-1
   057.061  311          01207           RET
                         01208
                         01209   *
                         01210   *       Wait for data-to-host mode, then read from data in port
```

```
                    01211   *
  057.062  315 051 057 01212   RDWAIT  CALL    RDSTAT
  057.065  346 320     01213           ANI     BS.REQ+BS.OUT+BS.COM
  057.067  362 062 057 01214           JP      RDWAIT
  057.072  376 220     01215           CPI     BS.REQ+BS.COM           Wait for Data to Host mode
  057.074  302 062 057 01216           JNZ     RDWAIT
  057.077  303 046 057 01217           JMP     RDDATA          now do the read operation
                    01218   *
                    01219   *       If C is not zero increment B and reset C to 0
                    01220   *
  057.102  365         01221   UNK59   PUSH    PSW
  057.103  171         01222           MOV     A,C
  057.104  247         01223           ANA     A
  057.105  302 112 057 01224           JNZ     UNK59A
  057.110  361         01225           POP     PSW
  057.111  311         01226           RET
  057.112  004         01227   UNK59A  INR     B
  057.113  016 000     01228           MVI     C,0
  057.115  361         01229           POP     PSW
  057.116  311         01230           RET
                    01231   *
                    01232   *       WAITR - wait for buss ready then read data
                    01233   *
  057.117  315 051 057 01234   WAITR   CALL    RDSTAT
  057.122  346 320     01235           ANI     BS.REQ+BS.OUT+BS.COM
  057.124  362 117 057 01236           JP      WAITR
  057.127  376 200     01237           CPI     BS.REQ
  057.131  302 117 057 01238           JNZ     WAITR
  057.134  303 046 057 01239           JMP     RDDATA
                    01240   *
                    01241   *       Initialize controller
                    01242   *
  057.137  315 272 057 01243   INITC   CALL    SEEK0           seek track 0
  057.142  320         01244           RNC
  057.143  076 020     01245           MVI     A,BC.RST        reset
  057.145  315 054 057 01246           CALL    WRCTRL
  057.150  315 336 057 01247           CALL    WAITS           wait for result
  057.153  330         01248           DB      BS.REQ+BS.OUT+BS.COM+BS.BSY
  057.154  100         01249           DB      BS.OUT          Wait for REQ and BSY to clear
  057.155  311         01250           RET
                    01251   *
                    01252   *       GETCON - Get controller's attention (and
                    01253   *                send first byte of command)
                    01254   *
  057.156  305         01255   GETCON  PUSH    B
  057.157  315 164 057 01256           CALL    GETCN1
  057.162  301         01257           POP     B
  057.163  311         01258           RET
                    01259
  057.164  315 336 057 01260   GETCN1  CALL    WAITS           Wait for BUSY bit to clear
  057.167  010         01261           DB      BS.BSY          AND mask
  057.170  000         01262           DB      0               expected result...
  057.171  330         01263           RC                      if 'C' - error!
                    01264
  057.172  363         01265           DI
  057.173  076 100     01266           MVI     A,BC.SEL        Assert SEL and DATA0
  057.175  315 054 057 01267           CALL    WRCTRL
```

```
   057.200  315 051 057 01268   GETCN2  CALL    RDSTAT
   057.203  346 010     01269           ANI     BS.BSY          wait for BUSY bit to clear
   057.205  312 200 057 01270           JZ      GETCN2          wait for arrival at "controller attention"
   057.210  076 002     01271           MVI     A,BC.EDT        Allow enable data
   057.212  315 054 057 01272           CALL    WRCTRL
   057.215  072 014 057 01273           LDA     CDRBLK          Get command byte from CDR block
   057.220  315 373 057 01274           CALL    SENDB           send command byte
   057.223  303 035 060 01275           JMP     UNK72
                        01276
   057.226  000         01277           DB      0
                        01278   *
                        01279   *       BLDCDR - Build 6-byte command descriptor block
                        01280   *
                        01281   *       ENTRY:  A = command byte
                        01282   *               H = logical address byte 1
                        01283   *               L = logical address byte 0
                        01284   *               B = number of blocks
                        01285   *
   057.227  305         01286   BLDCDR  PUSH    B
   057.230  325         01287           PUSH    D
   057.231  345         01288           PUSH    H
   057.232  062 014 057 01289           STA     CDRBLK          A contains the command
   057.235  072 235 056 01290           LDA     S.LUN           Logical Unit Number
   057.240  017         01291           RRC
   057.241  017         01292           RRC
   057.242  017         01293           RRC                     LUN in top 3 bits
   057.243  062 015 057 01294           STA     CDRBLK+1        store LUN
   057.246  174         01295           MOV     A,H
   057.247  062 016 057 01296           STA     CDRBLK+2        logical address 1
   057.252  175         01297           MOV     A,L
   057.253  062 017 057 01298           STA     CDRBLK+3        logical address 0
   057.256  170         01299           MOV     A,B
   057.257  062 020 057 01300           STA     CDRBLK+4        number of blocks
   057.262  257         01301           XRA     A
   057.263  062 021 057 01302           STA     CDRBLK+5        clear control byte
   057.266  341         01303           POP     H
   057.267  321         01304           POP     D
   057.270  301         01305           POP     B
   057.271  311         01306           RET
                        01307   *
                        01308   *       SEEK0 - seek track address 0
                        01309   *
   057.272  076 013     01310   SEEK0   MVI     A,D.SEK         Command: Seek
   057.274  001 000 000 01311           LXI     B,0
   057.277  140         01312           MOV     H,B
   057.300  151         01313           MOV     L,C             seek address 0
   057.301  315 227 057 01314           CALL    BLDCDR          build the command block
   057.304  315 156 057 01315           CALL    GETCON          Get controller attention and send command byte
   057.307  315 365 056 01316           CALL    DOCMD           Send last 5 bytes of CDR block
   057.312  303 315 057 01317           JMP     UNK67
                        01318
   057.315  305         01319   UNK67   PUSH    B
   057.316  315 022 057 01320   EXT171  CALL    READ2           read two bytes
   057.321  171         01321           MOV     A,C             second byte
   057.322  247         01322           ANA     A
   057.323  302 316 057 01323           JNZ     EXT171          wait for 0
   057.326  373         01324           EI                      interrupts back on!
```

```
   057.327  170          01325           MOV     A,B             first byte
   057.330  346 002      01326           ANI     2
   057.332  301          01327           POP     B
   057.333  310          01328           RZ
   057.334  067          01329           STC
   057.335  311          01330           RET
                         01331   *
                         01332   *       WAITS - Check and wait for status
                         01333   *
                         01334   *       ENTRY:  (SP+1) = mask byte
                         01335   *               (SP+2) = desired result
                         01336   *
   057.336  343          01337   WAITS   XTHL                    Get stack pointer
   057.337  106          01338           MOV     B,M             first byte in B
   057.340  043          01339           INX     H
   057.341  116          01340           MOV     C,M             Second byte in C
   057.342  043          01341           INX     H               fix stack and
   057.343  343          01342           XTHL                    put it back
                         01343
   057.344  325          01344           PUSH    D
   057.345  315 352 057  01345           CALL    WAITS1
   057.350  321          01346           POP     D
   057.351  311          01347           RET
                         01348   *
                         01349   *       WAITS1 - do the status check
                         01350   *
                         01351   *       ENTRY:  B = mask
                         01352   *               C = expected result
                         01353   *
   057.352  021 000 000  01354   WAITS1  LXI     D,0
   057.355  033          01355   WAITS2  DCX     D               Really big loop timer...
   057.356  172          01356           MOV     A,D
   057.357  263          01357           ORA     E
   057.360  067          01358           STC                     Set 'C' in case we time out...
   057.361  310          01359           RZ                      Return if time is up
   057.362  315 051 057  01360           CALL    RDSTAT          Read from status port
   057.365  240          01361           ANA     B               AND with the mask
   057.366  271          01362           CMP     C               CoMPare with exptected result
   057.367  302 355 057  01363           JNZ     WAITS2          loop 'til obtained.
   057.372  311          01364           RET
                         01365   *
                         01366   *       SENDB - send a byte from the CDR block
                         01367   *
                         01368   *       ENTRY:  A = data byte
                         01369   *
   057.373  365          01370   SENDB   PUSH    PSW             Save data byte
   057.374  315 051 057  01371   SENDB1  CALL    RDSTAT          Read status
   057.377  346 320      01372           ANI     BS.REQ+BS.OUT+BS.COM    Wait for ready
   060.001  362 374 057  01373           JP      SENDB1
   060.004  376 320      01374           CPI     BS.REQ+BS.OUT+BS.COM
   060.006  302 374 057  01375           JNZ     SENDB1
   060.011  361          01376           POP     PSW             restore data byte
   060.012  303 057 057  01377           JMP     WRDATA          OK, now write to data port
                         01378   *
                         01379   *       Now wait for completion...
                         01380   *
   060.015  315 051 057  01381   SENDB2  CALL    RDSTAT          Read status
```

```
060.020  346 320     01382           ANI     BS.REQ+BS.OUT+BS.COM    Wait for ready
060.022  362 015 060 01383           JP      SENDB2
060.025  346 120     01384           ANI     BS.OUT+BS.COM
060.027  376 020     01385           CPI     BS.COM
060.031  302 015 060 01386           JNZ     SENDB2
060.034  311         01387           RET                     OK done
                     01388
060.035  315 051 057 01389   UNK72   CALL    RDSTAT          ??? how does 'M' flag ever get cleared???
060.040  372 035 060 01390           JM      UNK72
060.043  311         01391           RET
                     01392
                     01393   ***     Console Routines
                     01394   *
                     01395   *
060.044              01396           XTEXT   XICTT



                     01398X  **      $ICTT - INPUT FROM CONSOLE TASK TIME.
                     01399X  *
                     01400X  *       $ICTT IS A TASK-TIME CONSOLE INPUT ROUTINE, WHICH
                     01401X  *       PERFORMS SIMPLE SINGLE CHARACTER INPUTS.
                     01402X  *
                     01403X  *       IT IS CALLED DURING BOOT OPERATIONS, AND BY SPECIAL ROUTINES
                     01404X  *       WHICH MAY BE RUNNING IN ENVIRONMENTS WHERE KEYBOARD INTERRUPTS
                     01405X  *       ARE UNDESIRABLE.
                     01406X  *
                     01407X  *       Modified to handle H8-4 ports by G. Chandler, 1-SEP-78
                     01408X  *       This routine assumes that the ports have been previously initialized,
                     01409X  *       and that S.CDB has been previously initialized.
                     01410X  *
                     01411X  *       ENTRY   NONE
                     01412X  *       EXIT    (A) = CHARACTER
                     01413X  *       USES    A,F
                     01414X
                     01415X
060.044  315 056 060 01416X  $ICTT   CALL    $ICTT.
060.047  332 044 060 01417X          JC      $ICTT
060.052  315 104 060 01418X          CALL    $ICTT..
060.055  311         01419X          RET
                     01420X
060.056  072 236 056 01421X  $ICTT.  LDA     SS.CDB
060.061  376 001     01422X          CPI     CDB.H84
060.063  312 075 060 01423X          JZ      ICTT2           IF  H8-4 PORT
                     01424X
                     01425X  *       HAVE 8251 FOR CONSOLE
                     01426X
060.066  333 373     01427X  ICTT1   IN      SC.UART+USR
060.070  346 002     01428X          ANI     USR.RXR
060.072  300         01429X          RNZ                     READY
                     01430X
060.073  067         01431X          STC                     FLAG NOT READY
060.074  311         01432X          RET
                     01433X
                     01434X  *       HAVE 8250 PORT FOR CONSOLE
                     01435X
```

```
   060.075  333 355     01436X  ICTT2   IN      SC.ACE+UR.LSR
   060.077  346 001     01437X          ANI     UC.DR
   060.101  300         01438X          RNZ                     READY
                        01439X
   060.102  067         01440X          STC                     FLAG NOT READY
   060.103  311         01441X          RET
                        01442X
   060.104  072 236 056 01443X  $ICTT.. LDA     SS.CDB
   060.107  376 001     01444X          CPI     CDB.H84
   060.111  312 121 060 01445X          JZ      ICTT3
                        01446X
                        01447X  *       HAVE 8251 FOR CONSOLE
                        01448X
   060.114  333 372     01449X          IN      SC.UART+UDR
   060.116  346 177     01450X          ANI     177Q
   060.120  311         01451X          RET
                        01452X
                        01453X  *       HAVE 8250 FOR CONSOLE
                        01454X
   060.121  333 350     01455X  ICTT3   IN      SC.ACE+UR.RBR
   060.123  346 177     01456X          ANI     177Q
   060.125  311         01457X          RET
   060.126              01458           XTEXT   XTYPET



                        01460X  **      $TYPET - TYPE TEXT.
                        01461X  *
                        01462X  *       $TYPET IS CALLED TO TYPE A BLOCK OF TEXT ON THE SYSTEM CONSOLE
                        01463X  *       AT TASK TIME RATHER THAN AT INTERRUPT TIME.
                        01464X  *
                        01465X  *       IMBEDDED ZERO BYTES INDICATE A CARRIAGE RETURN LINE FEED,
                        01466X  *       A BYTE WITH THE 200Q BIT SET IS THE LAST BYTE OF THE MESSAGE.
                        01467X  *
                        01468X  *       This routine modified to accomodate H8-4 ports by G.Chandler, 1-SEP-78
                        01469X  *       This routine assumes that the ports have been previously initialized,
                        01470X  *       and that S.CDB has been previously initialized.
                        01471X  *
                        01472X  *       Modified to use local version of S.CDB (SS.CDB)
                        01473X  *                                       gfr 8/23/2011
                        01474X  *
                        01475X  *       ENTRY   (RET) = TEXT
                        01476X  *       EXIT    TO (RET+LENGTH)
                        01477X  *       USES    A,F
                        01478X
                        01479X
   060.126  343         01480X  $TYPET  XTHL                    (HL) = TEXT ADDRESS
   060.127  315 134 060 01481X          CALL    $TYPET.         TYPE IT
   060.132  343         01482X          XTHL
   060.133  311         01483X          RET
                        01484X
   060.134  176         01485X  $TYPET. MOV     A,M
   060.135  346 177     01486X          ANI     177Q
   060.137  304 163 060 01487X          CNZ     $TYPEC.         IF NOT CRLF
   060.142  247         01488X          ANA     A
   060.143  314 154 060 01489X          CZ      $TYPET1         IS CRLF
```

```
060.146  276          01490X           CMP     M
060.147  043          01491X           INX     H
060.150  300          01492X           RNE                     WAS 200 BIT SET
060.151  303 134 060  01493X           JMP     $TYPET.
                      01494X
                      01495X  *        TYPE CRLF
                      01496X
060.154  315 126 060  01497X  $TYPET1  CALL    $TYPET
060.157  015 212      01498X           DB      CR,LF+200Q
060.161  257          01499X           XRA     A               RESTORE (A)
060.162  311          01500X           RET



                      01502X  **       $TYPEC. - TYPE SINGLE CHARACTER.
                      01503X  *
                      01504X  *        IF CR, PADD WITH 4 ZERO BYTES
                      01505X  *
                      01506X  *        ENTRY   (A) = CHARACTER
                      01507X  *        EXIT    (A) = CHARACTER
                      01508X  *        USES    A,F
                      01509X
                      01510X
060.163  365          01511X  $TYPEC.  PUSH    PSW             SAVE CHAR
060.164  072 236 056  01512X           LDA     SS.CDB
060.167  376 001      01513X           CPI     CDB.H84
060.171  312 211 060  01514X           JZ      TYPEC2          IF H8-4 PORT
                      01515X
                      01516X  *        HAVE 8251 PORT FOR CONSOLE
                      01517X
060.174  333 373      01518X  TYPEC1   IN      SC.UART+USR
060.176  346 001      01519X           ANI     USR.TXR
060.200  312 174 060  01520X           JZ      TYPEC1          NOT READY
060.203  361          01521X           POP     PTT
060.204  323 372      01522X           OUT     SC.UART+UDR
060.206  303 223 060  01523X           JMP     TYPEC3
                      01524X
                      01525X  *        HAVE 8250 PORT FOR CONSOLE
                      01526X
060.211  333 355      01527X  TYPEC2   IN      SC.ACE+UR.LSR
060.213  346 040      01528X           ANI     UC.THE
060.215  312 211 060  01529X           JZ      TYPEC2          NOT READY
060.220  361          01530X           POP     PSW
060.221  323 350      01531X           OUT     SC.ACE+UR.THR
                      01532X
060.223  376 015      01533X  TYPEC3   CPI     CR
060.225  300          01534X           RNE                     NOT CR
                      01535X
                      01536X  *        IS CR, PADD 4 TIMES
                      01537X
060.226  076 004      01538X           MVI     A,4
060.230  365          01539X  TYPEC4   PUSH    PSW
060.231  257          01540X           XRA     A
060.232  315 163 060  01541X           CALL    $TYPEC.
060.235  361          01542X           POP     PSW
060.236  075          01543X           DCR     A
```

```
060.237  302 230 060 01544X          JNZ     TYPEC4
060.242  076 015     01545X          MVI     A,CR
060.244  311         01546X          RET



                     01548   **      FCU - FIND CONSOLE USART.
                     01549   *
                     01550   *       FCU FINDS AND CONFIGURES THE CONSOLE USART.
                     01551   *
                     01552   *       ENTRY   NONE
                     01553   *       EXIT    NONE
                     01554   *       USES    A,F,(BC),(HL)
                     01555   *
                     01556
060.245  257         01557   FCU     XRA     A
060.246  323 351     01558           OUT     SC.ACE+UR.IER   interrupts off (8250)
060.250  323 373     01559           OUT     SC.UART+USR     and also (8251)
060.252  076 003     01560           MVI     A,UC.8BW        8 bits
060.254  323 353     01561           OUT     SC.ACE+UR.LCR   try setting it (assuming 8250)
060.256  333 353     01562           IN      SC.ACE+UR.LCR   try to read it back
060.260  376 003     01563           CPI     UC.8BW          see if unchanged
060.262  076 000     01564           MVI     A,CDB.H85
060.264  041 000 000 01565           LXI     H,0
060.267  302 315 060 01566           JNZ     FCU1            readback failed, must be 8251?
                     01567
060.272  041 014 000 01568           LXI     H,BAU.96        BAUD divisor hard wired for 9600
                     01569
060.275  174         01570           MOV     A,H
060.276  346 200     01571           ANI     10000000B
060.300  312 313 060 01572           JZ      FCU0
060.303  072 237 056 01573           LDA     X.CONTY
060.306  366 010     01574           ORI     CTP.2SB
060.310  062 237 056 01575           STA     X.CONTY
060.313  076 001     01576   FCU0    MVI     A,CDB.H84       must be 8251
                     01577
                     01578   *       Have type and BAUD rate.
                     01579   *       (A) = S.CDB value
                     01580   *       (HL) = BAUD rate (0 if 8251)
                     01581
060.315  042 240 056 01582   FCU1    SHLD    X.BAUD
060.320  062 236 056 01583           STA     SS.CDB
060.323  303 326 060 01584           JMP     SCU
                     01585
060.326              01586           XTEXT   XSCU            Setup console USART



                     01588X  **      SCU - SETUP CONSOLE USART.
                     01589X  *
                     01590X  *       SCU CONFIGURES THE CONSOLE USART.
                     01591X  *
                     01592X  *       IF  8250
                     01593X  *         THEN  PORT = 372-3Q
                     01594X  *         ELSE  PORT = 340-7Q
```

```
                    01595X  *
                    01596X  *
                    01597X  *       ENTRY   NONE
                    01598X  *       EXIT    NONE
                    01599X  *       USES    A,F,(BC),(HL)
                    01600X  *
                    01601X  *       Modified to use local copy of S.CDB (SS.CDB),
                    01602X  *       S.CONTY (X.CONTY), and S.BAUD (X.BAUD)
                    01603X  *                                       gfr 8/23/2011
                    01604X  *
                    01605X
060.326  072 236 056 01606X  SCU     LDA     SS.CDB
060.331  376 001     01607X          CPI     CDB.H84
060.333  312 376 060 01608X          JZ      SCU1                    IF  8250
                    01609X
                    01610X  *       PRESET 8251
                    01611X
060.336  076 201     01612X          MVI     A,201Q
060.340  323 373     01613X          OUT     SC.UART+USR             GET USART IN KNOWN STATE
060.342  323 373     01614X          OUT     SC.UART+USR
060.344  323 373     01615X          OUT     SC.UART+USR
060.346  323 373     01616X          OUT     SC.UART+USR
060.350  076 100     01617X          MVI     A,UCI.IR                RESET
060.352  323 373     01618X          OUT     SC.UART+USR
060.354  072 237 056 01619X          LDA     X.CONTY
060.357  346 010     01620X          ANI     CTP.2SB
000.000              01621X          ERRNZ   CTP.2SB*16+UMI.1B-UMI.2B
060.361  007         01622X          RLC
060.362  007         01623X          RLC
060.363  007         01624X          RLC
060.364  007         01625X          RLC
060.365  366 116     01626X          ORI     UMI.1B+UMI.L8+UMI.16X
060.367  323 373     01627X          OUT     SC.UART+USR
060.371  076 025     01628X          MVI     A,UCI.ER+UCI.RE+UCI.TE
060.373  323 373     01629X          OUT     SC.UART+USR
060.375  311         01630X          RET
                    01631X
                    01632X  *       IS 8250
                    01633X
060.376  333 355     01634X  SCU1    IN      SC.ACE+UR.LSR                                   /80.01.GC/
061.000  346 100     01635X          ANI     UC.TSE          CHECK FOR SHIFT EMPTY           /80.01.GC/
061.002  312 376 060 01636X          JZ      SCU1
                    01637X
061.005  257         01638X          XRA     A                                               /79.01.GC/
061.006  323 351     01639X          OUT     SC.ACE+UR.IER   TURN OFF ANY INTERRUPTS         /79.01.GC/
061.010  076 020     01640X          MVI     A,UC.LOO                                        /79.01.GC/
061.012  323 354     01641X          OUT     SC.ACE+UR.MCR                                   /79.01.GC/
061.014  052 240 056 01642X          LHLD    X.BAUD
061.017  076 200     01643X          MVI     A,UC.DLA
061.021  323 353     01644X          OUT     SC.ACE+UR.LCR   ACCESS DIVISOR LATCHES
061.023  175         01645X          MOV     A,L
061.024  323 350     01646X          OUT     SC.ACE+UR.DLL   SET LEAST SIGNIFICANT
061.026  174         01647X          MOV     A,H
061.027  346 177     01648X          ANI     177Q            TRIM STOP BITS
061.031  323 351     01649X          OUT     SC.ACE+UR.DLM   SET MOST SIGNIFICANT
061.033  072 237 056 01650X          LDA     X.CONTY
061.036  346 010     01651X          ANI     CTP.2SB
```

```
   061.040  017          01652X          RRC
   000.000               01653X          ERRNZ   CTP.2SB/2-UC.2SB
   000.000               01654X          ERRNZ   UC.2SB-4        (A) = UC.2SB IF 2 STOP BITS
   061.041  366 003      01655X          ORI     UC.8BW          8 BIT WORDS
   061.043  323 353      01656X          OUT     SC.ACE+UR.LCR
   061.045  076 156      01657X          MVI     A,AC.DLY                                        /79.01.GC/
   061.047  315 053 000  01658X          CALL    .DLY                                            /79.01.GC/
   061.052  333 350      01659X          IN      SC.ACE+UR.RBR  GOBBLE ANY TRASH                 /79.01.GC/
   061.054  333 354      01660X          IN      SC.ACE+UR.MCR                                   /79.01.GC/
   061.056  346 357      01661X          ANI     377Q-UC.LOO                                     /79.01.GC/
   061.060  323 354      01662X          OUT     SC.ACE+UR.MCR                                   /79.01.GC/
   061.062  311          01663X          RET
                         01664   *
                         01665   *       Read/parse boot string
                         01666   *
                         01667   *       ENTRY:  BC = string length
                         01668   *               (HL) = string text
                         01669
   061.063  171          01670   PARSEB  MOV     A,C
   061.064  260          01671           ORA     B
   061.065  312 263 061  01672           JZ      PARSB2          done?
                         01673   *
                         01674   *       Skip leading colon if present
                         01675   *
   061.070  176          01676           MOV     A,M             fetch a char
   061.071  376 072      01677           CPI     ':'
   061.073  302 105 061  01678           JNZ     PARSB1          jump if not ':'
   061.076  013          01679           DCX     B               else skip over ':'
   061.077  043          01680           INX     H
                         01681
   061.100  171          01682           MOV     A,C
   061.101  260          01683           ORA     B
   061.102  312 263 061  01684           JZ      PARSB2          done?
                         01685   *
                         01686   *       now parse the string
   061.105  042 266 061  01687   PARSB1  SHLD    BSPTR           store pointer to boot string
   061.110  305          01688           PUSH    B
   061.111  341          01689           POP     H               HL = BC (length?)
   061.112  042 270 061  01690           SHLD    BSLEN           store boot string length
   061.115  257          01691           XRA     A
   061.116  062 005 065  01692           STA     OCCNUM          default user partition = 0
                         01693
   061.121  076 040      01694           MVI     A,' '
   061.123  001 020 000  01695           LXI     B,16            length
   061.126  021 365 064  01696           LXI     D,OSNAME
   061.131  315 175 062  01697           CALL    FILLC           blank out OS name entry
                         01698
   061.134  052 270 061  01699           LHLD    BSLEN
   061.137  345          01700           PUSH    H
   061.140  301          01701           POP     B               BC = length
   061.141  052 266 061  01702           LHLD    BSPTR           HL = string
   061.144  315 076 062  01703           CALL    ISANUM          see if it's alphanumeric
   061.147  173          01704           MOV     A,E
   061.150  262          01705           ORA     D
   061.151  312 263 061  01706           JZ      PARSB2          not alphauneric - abort
                         01707
   061.154  041 020 000  01708           LXI     H,16
```

```
   061.157  315 154 062 01709           CALL    HLCPDE
   061.162  332 263 061 01710           JC      PARSB2          < 16 alphanumeric characters? - abort
                        01711   *
                        01712   *       have valid 16 character alphanumeric boot string
                        01713   *
   061.165  305         01714           PUSH    B
   061.166  341         01715           POP     H               HL=BC
   061.167  042 270 061 01716           SHLD    BSLEN           save it
   061.172  325         01717           PUSH    D
   061.173  301         01718           POP     B               BC=DE
   061.174  052 266 061 01719           LHLD    BSPTR
   061.177  021 365 064 01720           LXI     D,OSNAME
   061.202  315 162 062 01721           CALL    MOVEB
   061.205  042 266 061 01722           SHLD    BSPTR
   061.210  052 270 061 01723           LHLD    BSLEN
   061.213  345         01724           PUSH    H
   061.214  301         01725           POP     B
   061.215  170         01726           MOV     A,B
   061.216  261         01727           ORA     C
   061.217  312 261 061 01728           JZ      EXT200
   061.222  052 266 061 01729           LHLD    BSPTR
   061.225  176         01730           MOV     A,M
   061.226  043         01731           INX     H
   061.227  013         01732           DCX     B
   061.230  376 073     01733           CPI     ';'             Optionally followed by occurrence number
   061.232  302 263 061 01734           JNZ     PARSB2          only ';' allowed!
   061.235  170         01735           MOV     A,B
   061.236  261         01736           ORA     C
   061.237  312 261 061 01737           JZ      EXT200
   061.242  315 344 061 01738           CALL    GETNUM          get the numeric portion (occurrence number)
   061.245  302 263 061 01739           JNZ     PARSB2          error in user-defined partition # - abort!
   061.250  173         01740           MOV     A,E
   061.251  376 100     01741           CPI     64
   061.253  322 263 061 01742           JNC     PARSB2          must be less than 64
   061.256  062 005 065 01743           STA     OCCNUM          save user-defined occurrence number
   061.261  257         01744   EXT200  XRA     A
   061.262  311         01745           RET
                        01746
   061.263  366 001     01747   PARSB2  ORI     1
   061.265  311         01748           RET
                        01749
   061.266  000 000     01750   BSPTR   DW      0               pointer to boot string
   061.270  000 000     01751   BSLEN   DW      0               length of boot string
                        01752
                        01753   *
                        01754   *       WAITCR - prompt user for CR and wait
                        01755   *
   061.272  315 126 060 01756   WAITCR  CALL    $TYPET
   061.275  000 000 160 01757           DB      0,0,'press return to continue...',200Q
   061.333  041 325 064 01758           LXI     H,INPSTR
   061.336  076 001     01759           MVI     A,1             read a single character
   061.340  315 162 056 01760           CALL    RDSTR
   061.343  311         01761           RET
                        01762   *
                        01763   *       Get numeric value from ASCII string
                        01764   *
                        01765   *       ENTRY:  BC = count
```

```
                        01766   *               (HL) = string
                        01767   *
                        01768   *       EXIT:   DE = number
                        01769   *
   061.344  021 000 000 01770   GETNUM  LXI     D,0             zero the value
                        01771
   061.347  170         01772   GETNM1  MOV     A,B
   061.350  261         01773           ORA     C               count = 0?
   061.351  310         01774           RZ                      yes, done
   061.352  176         01775           MOV     A,M             get character
   061.353  043         01776           INX     H               point to next
   061.354  326 060     01777           SUI     '0'             convert to decimal
   061.356  332 366 061 01778           JC      GETNM2
   061.361  376 012     01779           CPI     10
   061.363  332 371 061 01780           JC      GETNM3
   061.366  366 001     01781   GETNM2  ORI     1               exit, non numeric found...
   061.370  311         01782           RET
                        01783   *
                        01784   *       have 0-9
                        01785   *
   061.371  345         01786   GETNM3  PUSH    H
   061.372  041 000 000 01787           LXI     H,0             HL = 0
   061.375  031         01788           DAD     D               HL = DE
   061.376  051         01789           DAD     H               HL = DE*2
   061.377  051         01790           DAD     H               HL = DE*4
   062.000  031         01791           DAD     D               HL = DE*5
   062.001  051         01792           DAD     H               HL = DE*10
   062.002  137         01793           MOV     E,A
   062.003  026 000     01794           MVI     D,0             DE = number
   062.005  031         01795           DAD     D               HL = DE*10+number
   062.006  353         01796           XCHG                    DE = DE*10+number
   062.007  341         01797           POP     H
   062.010  013         01798           DCX     B               count down
   062.011  303 347 061 01799           JMP     GETNM1          and loop
                        01800
   062.014              01801           XTEXT   BCRC



                        01803X  **      $BCRC - GENERATE CRC16 ON A BLOCK OF DATA.
                        01804X  *
                        01805X  *       *** WARNING ***
                        01806X  *
                        01807X  *       THIS CRC-16 IS NOT COMPATIBLE WITH THE ONE
                        01808X  *       PRODUCED BY PAM-8, AND THE DECK CRC.COM!
                        01809X  *
                        01810X  *       ENTRY   (BC) = BYTE COUNT
                        01811X  *               (HL) = ADDRESS
                        01812X  *               (DE) = CRC ACCUMULATOR
                        01813X  *       EXIT    (HL) = (HL)+(BC)
                        01814X  *               (DE) = NEW CRC
                        01815X  *       USES    ALL
                        01816X
                        01817X
   062.014  170         01818X  $BCRC   MOV     A,B
   062.015  261         01819X          ORA     C
```

```
   062.016  310          01820X          RZ                      NO MORE
   062.017  176          01821X          MOV     A,M             (A) = NEW BYTE
   062.020  345          01822X          PUSH    H
   062.021  305          01823X          PUSH    B               SAVE REGISTERS
   062.022  253          01824X          XRA     E
   062.023  107          01825X          MOV     B,A
   062.024  017          01826X          RRC
   062.025  017          01827X          RRC
   062.026  017          01828X          RRC
   062.027  017          01829X          RRC
   062.030  117          01830X          MOV     C,A
   062.031  250          01831X          XRA     B
   062.032  346 360      01832X          ANI     0F0H
   062.034  252          01833X          XRA     D
   062.035  157          01834X          MOV     L,A
   062.036  171          01835X          MOV     A,C
   062.037  007          01836X          RLC
   062.040  346 037      01837X          ANI     1FH
   062.042  255          01838X          XRA     L
   062.043  157          01839X          MOV     L,A
   062.044  170          01840X          MOV     A,B
   062.045  007          01841X          RLC
   062.046  346 001      01842X          ANI     1
   062.050  252          01843X          XRA     D
   062.051  255          01844X          XRA     L
   062.052  127          01845X          MOV     D,A
   062.053  171          01846X          MOV     A,C
   062.054  346 360      01847X          ANI     0F0H
   062.056  250          01848X          XRA     B
   062.057  137          01849X          MOV     E,A
   062.060  171          01850X          MOV     A,C
   062.061  250          01851X          XRA     B
   062.062  007          01852X          RLC
   062.063  346 340      01853X          ANI     0E0H
   062.065  253          01854X          XRA     E
   062.066  137          01855X          MOV     E,A
   062.067  301          01856X          POP     B
   062.070  341          01857X          POP     H
   062.071  043          01858X          INX     H
   062.072  013          01859X          DCX     B
   062.073  303 014 062  01860X          JMP     $BCRC
                         01861
                         01862   *
                         01863   *       ISANUM - Test if string is alphanumeric
                         01864   *
                         01865   *       ENTRY:  BC = length
                         01866   *               (HL) = string
                         01867   *
                         01868   *       EXIT:   DE = length processed
                         01869   *               'Z' set if alphanumeric
                         01870   *
   062.076  021 000 000  01871   ISANUM  LXI     D,0             length = 0
                         01872
   062.101  170          01873   ISANM1  MOV     A,B
   062.102  261          01874           ORA     C
   062.103  310          01875           RZ                      count = 0? (exit)
                         01876
```

```
062.104  176          01877           MOV     A,M             fetch a byte
062.105  376 060      01878           CPI     '0'
062.107  332 151 062  01879           JC      ISANM3          < 0 -> non alpha
062.112  376 072      01880           CPI     '9'+1
062.114  332 143 062  01881           JC      ISANM2          0-9 ... OK!
062.117  376 101      01882           CPI     'A'
062.121  332 151 062  01883           JC      ISANM3          < 'A' -> non alpha
062.124  376 133      01884           CPI     'Z'+1
062.126  332 143 062  01885           JC      ISANM2          A-Z ... OK!
062.131  376 141      01886           CPI     'a'
062.133  332 151 062  01887           JC      ISANM3          < 'a' -> non alpha
062.136  376 173      01888           CPI     'z'+1
062.140  322 151 062  01889           JNC     ISANM3          > 'z' -> non alpha
                      01890   *
                      01891   *       isalpha [0..9; A..Z; a..z]
                      01892   *
062.143  023          01893   ISANM2  INX     D               bump length count
062.144  043          01894           INX     H               point to next storage location
062.145  013          01895           DCX     B               count down
062.146  303 101 062  01896           JMP     ISANM1          and loop...
                      01897
062.151  366 001      01898   ISANM3  ORI     1
062.153  311          01899           RET
                      01900
062.154               01901           XTEXT   HLCPDE
                      01902X
                      01903X  **      HLCPDE  - (HL) COMPARED TO (DE)
                      01904X  *
                      01905X  *       THIS ROUTINE IS DOUBLE WORD COMPARE OF REGISTER PAIRS (DE) AND (HL).
                      01906X  *
                      01907X  *       ENTRY:  (HL)&(DE) SET UP
                      01908X  *
                      01909X  *       EXIT:   (PSW)   =
                      01910X  *                        'Z' SET   IF (HL) =  (DE)
                      01911X  *                        'C' SET   IF (HL) <  (DE)
                      01912X  *                        'C' CLEAR IF (HL) >= (DE)
                      01913X  *
                      01914X  *
                      01915X  *       USES:   (PSW)
                      01916X  *
                      01917X
062.154  174          01918X  HLCPDE  MOV     A,H
062.155  272          01919X          CMP     D       'C' SET => (A) < (D)
062.156  300          01920X          RNZ
062.157  175          01921X          MOV     A,L
062.160  273          01922X          CMP     E       'C' SET => (L) < (E)
062.161  311          01923X          RET
                      01924   *
                      01925   *       MOVEB - Move pre-determined number of bytes from one location to another
                      01926   *
                      01927   *       ENTRY:  BC = count
                      01928   *               (HL) = source
                      01929   *               (DE) = destination
                      01930   *
                      01931   *       EXIT:   'Z' set if count == 0
                      01932   *
062.162  170          01933   MOVEB   MOV     A,B
```

```
   062.163  261          01934           ORA     C
   062.164  310          01935           RZ                      if BC==0 return
   062.165  176          01936           MOV     A,M             fetch a byte
   062.166  022          01937           STAX    D               store a byte
   062.167  043          01938           INX     H               next source
   062.170  023          01939           INX     D               next destination
   062.171  013          01940           DCX     B               count down...
   062.172  303 162 062  01941           JMP     MOVEB           loop
                         01942   *
                         01943   *       FILLC - Fills a location with a character
                         01944   *
                         01945   *       ENTRY:  A = fill character
                         01946   *               BC = count
                         01947   *               DE = location
                         01948   *
   062.175  157          01949   FILLC   MOV     L,A
   062.176  170          01950           MOV     A,B
   062.177  261          01951           ORA     C
   062.200  175          01952           MOV     A,L
   062.201  310          01953           RZ
   062.202  022          01954           STAX    D
   062.203  023          01955           INX     D
   062.204  013          01956           DCX     B
   062.205  303 175 062  01957           JMP     FILLC
                         01958   *
                         01959   *       DOMENU - print screen menu
                         01960   *
                         01961   *       (uses H19/H89 escape sequences)
                         01962   *
   062.210  315 126 060  01963   DOMENU  CALL    $TYPET
   062.213  033 110      01964           DB      ESC,'H'         Cursor home
   062.215  033 112      01965           DB      ESC,'J'         Erase to end of page
   062.217  000 040 040  01966           DB      0,'                                 '
   062.257  110 105 101  01967           DB      'HEATH/ZENITH H/Z67'
   062.301  000          01968           DB      0
   062.302  040 040 040  01969           DB      '                         '
   062.331  123 157 146  01970           DB      'Software Boot Code (SBC) vers '
   062.367  061 056 061  01971           DB      '1.1',200Q
   062.373  315 126 060  01972           CALL    $TYPET
   062.376  000 040 040  01973           DB      0,'                                '
   063.035  102 157 157  01974           DB      'Boot Option(s) Menu',0
   063.061  000          01975           DB      0
                         01976
   063.062  117 160 145  01977           DB      'Operating systems'
   063.103  072 040 040  01978           DB      ':     Maximum occurrence number'
   063.142  072 000      01979           DB      ':',0
   063.144  055 055 055  01980           DB      '------------------     --'
   063.175  055 055 055  01981           DB      '-------------------------',200Q
                         01982
   063.226  041 104 065  01983           LXI     H,SB.IDT        OS ID Table
   063.231  006 020      01984           MVI     B,16            loop over 16 entries (count down)
   063.233  016 001      01985           MVI     C,1             OS count (count up)
   063.235  305          01986   EXT218  PUSH    B
   063.236  345          01987           PUSH    H
   063.237  176          01988           MOV     A,M
   063.240  376 040      01989           CPI     ' '             is it blank?
   063.242  312 336 063  01990           JZ      EXT220          yep, next...
```

```
                     01991   *
                     01992   *       non blank OK entry
                     01993   *
   063.245  315 065 064 01994           CALL    UNK103
   063.250  170         01995           MOV     A,B
   063.251  267         01996           ORA     A
   063.252  312 336 063 01997           JZ      EXT220
   063.255  075         01998           DCR     A
   063.256  062 352 063 01999           STA     UNK98
                     02000
   063.261  076 040     02001           MVI     A,' '
   063.263  001 074 000 02002           LXI     B,60
   063.266  021 010 065 02003           LXI     D,BUFFER
   063.271  315 175 062 02004           CALL    FILLC           blank out 60 bytes
                     02005
   063.274  341         02006           POP     H               Source
   063.275  345         02007           PUSH    H
   063.276  021 010 065 02008           LXI     D,BUFFER        Dest
   063.301  001 020 000 02009           LXI     B,16            16 bytes
   063.304  315 162 062 02010           CALL    MOVEB           move 'em!
   063.307  041 024 000 02011           LXI     H,20
   063.312  031         02012           DAD     D
   063.313  072 352 063 02013           LDA     UNK98
   063.316  117         02014           MOV     C,A
   063.317  006 000     02015           MVI     B,0
   063.321  076 002     02016           MVI     A,2
   063.323  315 157 031 02017           CALL    $UDD
   063.326  066 200     02018           MVI     M,200Q
   063.330  041 010 065 02019           LXI     H,BUFFER
   063.333  315 134 060 02020           CALL    $TYPET.
                     02021   *
                     02022   *       next...
                     02023   *
   063.336  341         02024   EXT220  POP     H
   063.337  301         02025           POP     B
   063.340  021 020 000 02026           LXI     D,16            jump 16 bytes to next entry
   063.343  031         02027           DAD     D
   063.344  014         02028           INR     C               increment counter
   063.345  005         02029           DCR     B               done all of them?
   063.346  302 235 063 02030           JNZ     EXT218          nope, loop
   063.351  311         02031           RET
                     02032
                     02033
   063.352  000         02034   UNK98   DB      0
                     02035   *
                     02036   *       FINDO - Find an OS occurrence in the partition table
                     02037   *
                     02038   *       ENTRY:  A = OS occurrence number (zero-based)
                     02039   *               B = OS number from OS ID table
                     02040   *
   063.353  117         02041   FINDO   MOV     C,A
   063.354  014         02042           INR     C               Use C as count-down counter
   063.355  041 104 066 02043           LXI     H,SB.SAT        Sector allocation table
   063.360  176         02044   FINDO1  MOV     A,M             fetch a byte
   063.361  043         02045           INX     H               and point to next...
   063.362  376 037     02046           CPI     1FH             end of table?
   063.364  312 003 064 02047           JZ      FINDO3          yes return
```

```
063.367  270          02048           CMP     B               is it the OS we want?
063.370  302 375 063  02049           JNZ     FINDO2          no
063.373  015          02050           DCR     C               yes! count down
063.374  310          02051           RZ                      when zero we've found the desired occurrence no.
063.375  043          02052   FINDO2  INX     H               elxe skip to next entry
063.376  043          02053           INX     H
063.377  043          02054           INX     H
064.000  303 360 063  02055           JMP     FINDO1          and loop...
                      02056   *
                      02057   *       end reached without finding match.
                      02058   *
064.003  366 001      02059   FINDO3  ORI     1               clear 'Z'
064.005  311          02060           RET
                      02061   *
                      02062   *       STRCMP - Compare two strings (case insensitive)
                      02063   *
                      02064   *       ENTRY:  BC = string length
                      02065   *               (HL) = string1
                      02066   *               (DE) = string2
                      02067   *
                      02068   *       EXIT:   'Z' set if same, otherwise different
                      02069   *
064.006  170          02070   STRCMP  MOV     A,B
064.007  261          02071           ORA     C               check counter
064.010  310          02072           RZ                      return if done
064.011  176          02073           MOV     A,M             character from string1
064.012  376 141      02074           CPI     'a'
064.014  332 021 064  02075           JC      STRCP1          < 'a'?
064.017  346 337      02076           ANI     337Q            map to UPPER case
064.021  062 051 064  02077   STRCP1  STA     STRTMP          save it
064.024  032          02078           LDAX    D               character from string2
064.025  376 141      02079           CPI     'a'
064.027  332 034 064  02080           JC      STRCP2          < 'a'?
064.032  346 337      02081           ANI     337Q            map to UPPER case
064.034  345          02082   STRCP2  PUSH    H
064.035  041 051 064  02083           LXI     H,STRTMP
064.040  276          02084           CMP     M               compare the two
064.041  341          02085           POP     H
064.042  300          02086           RNZ                     not equal? return with 'Z' clear!
064.043  043          02087           INX     H               else next string1
064.044  023          02088           INX     D               next string2
064.045  013          02089           DCX     B               decrement the count
064.046  303 006 064  02090           JMP     STRCMP          and loop...
064.051  000          02091   STRTMP  DB      0               temporary storage
                      02092
                      02093   *
                      02094   *       SKIP0 - Skip to first null
                      02095   *
                      02096   *       ENTRY:  (HL) = character array
                      02097   *
                      02098   *       EXIT:   HL points to next NULL entry
                      02099   *
                      02100   *       USES:   A, BC
                      02101   *
064.052  001 000 000  02102   SKIP0   LXI     B,0
064.055  176          02103   SKIP1   MOV     A,M             fetch a byte
064.056  267          02104           ORA     A               test it
```

```
   064.057  310          02105           RZ                      if 0 -> done!
   064.060  003          02106           INX     B               else up the count
   064.061  043          02107           INX     H               point to next
   064.062  303 055 064  02108           JMP     SKIP1           and loop
                         02109   *
                         02110   *       Scan Sector allocation Table
                         02111   *
   064.065  171          02112   UNK103  MOV     A,C
   064.066  006 000      02113           MVI     B,0
   064.070  041 104 066  02114           LXI     H,SB.SAT        Sector allocation table
   064.073  176          02115   EXT233  MOV     A,M
   064.074  271          02116           CMP     C
   064.075  302 101 064  02117           JNZ     EXT234
   064.100  004          02118           INR     B
   064.101  376 037      02119   EXT234  CPI     1FH             end of table?
   064.103  310          02120           RZ
   064.104  043          02121           INX     H               each entry is 4 bytes
   064.105  043          02122           INX     H               skip to next
   064.106  043          02123           INX     H
   064.107  043          02124           INX     H
   064.110  303 073 064  02125           JMP     EXT233
                         02126
   064.113  310          02127           DB      200             not referenced ??
                         02128
   064.114                02129           DS      137             pad to 10 sectors...
                         02130   *
                         02131   *       String storage
                         02132   *
   064.325                02133   INPSTR  DS      1
   064.326                02134   STRLEN  DS      1
   064.327                02135   STRTXT  DS      30
   064.365                02136   OSNAME  DS      16              OS ID Table Entry
   065.005                02137   OCCNUM  DS      1               User-defined occurrence no to boot from
   065.006                02138   SPSAVE  DS      2
   065.010                02139   BUFFER  DS      60
                         02140
   065.104                02141   SUPBLK  EQU     *               Superblock
   065.104                02142   SB.IDT  DS      256             OS Ident table
   066.104                02143   SB.SAT  DS      256             Sector allocation table
   067.104                02144           DS      256             Region control table
                         02145
   070.104                02146           DS      612             stack area
   072.250                02147   EXT245  EQU     *
                         02148
   072.250  000          02149           END     ENTRY


02149 Statements Assembled
34964 Bytes Free
 No Errors Detected
```

$BCRC	062014	$ICTT	060044	$ICTT.	060056	$ICTT..	060104	$TYPEC.	060163	$TYPET	060126
$TYPET.	060134	$TYPET1	060154	$UDD	031157	.ABUSS	040024	.ALARM	002136	.ALEDS	040013
.CRC	002347	.CRCSUM	040027	.CTC	002172	.CTL2FL	040066	.CTLFLG	040011	.DLEDS	040021
.DLY	000053	.DOD	003122	.DODA	003356	.DSPMOD	040007	.DSPROT	040006	.DUMP	001374
.HORN	002140	.IDENT	000000	.IOWRK	040002	.LOAD	001267	.MFLAG	040010	.NMIRET	040064
.PCHL	002264	.RCK	003260	.REGI	040005	.REGPTR	040035	.RNB	002331	.RNP	002325
.SRS	002265	.START	040000	.TICCNT	040033	.TPERR	002205	.TPERRX	040031	.UIVEC	040037
.WNB	003024	.WNP	003017	AC.DLY	000156	AIO.CGN	041047	AIO.CHA	041116	AIO.CNT	041111
AIO.CSI	041050	AIO.DDA	041041	AIO.DES	041055	AIO.DEV	041057	AIO.DIR	041062	AIO.DTA	041053
AIO.EOF	041113	AIO.EOM	041112	AIO.FLG	041043	AIO.GRT	041044	AIO.LGN	041051	AIO.LSI	041052
AIO.SPG	041046	AIO.TFP	041114	AIO.UNI	041061	AIO.VEC	041040	ASKBS	054373	BASE	000170
BAU.96	000014	BC.EDT	000002	BC.IE	000040	BC.RST	000020	BC.SEL	000100	BELL	000007
BKSP	000010	BLDCDR	057227	BOOLEN	011000	BOOT.P	000001	BS.BSY	000010	BS.COM	000020
BS.DAT	000000	BS.DTD	000100	BS.HID	000001	BS.IN	000000	BS.INT	000004	BS.LMB	000040
BS.MTY	000020	BS.OUT	000100	BS.PE	000002	BS.REQ	000200	BSLEN	061270	BSPTR	061266
BUFFER	065010	C.STX	000002	C.SYN	000026	CB.CLI	000100	CB.MTL	000040	CB.SPK	000200
CB.SSI	000020	CB2.CLI	000002	CB2.ORG	000040	CB2.SID	000100	CB2.SSI	000001	CDB.H84	000001
CDB.H85	000000	CDRBLK	057014	CHKSBA	054031	CHKSBB	054100	CLASS0	000000	CLASS1	000040
CLASS6	000300	CLASSM	000340	CN.170M	000014	CN.174M	000003	CN.ABO	000200	CN.BAU	000100
CN.MEM	000040	CN.PRI	000020	CND.H17	000000	CND.H47	000001	CND.NDI	000000	CO.FLG	000001
CR	000015	CS.FLG	000200	CS.SBA	042262	CS.SBB	042264	CSL.CHR	000001	CSL.ECH	000200
CSL.RAW	000004	CSL.WRP	000002	CTLA	000001	CTLB	000002	CTLC	000003	CTLD	000004
CTLO	000017	CTLP	000020	CTLQ	000021	CTLS	000023	CTLZ	000032	CTP.2SB	000010
CTP.BKM	000002	CTP.BKS	000200	CTP.FF	000100	CTP.MLI	000040	CTP.MLO	000020	CTP.TAB	000001
D.CON	040110	D.CP3	000040	D.CTF	000005	D.FBS	000007	D.FFD	000300	D.FOR	000004
D.FT	000006	D.RAM	040240	D.REA	000010	D.REC	000001	D.RSE	000003	D.RSY	000002
D.SEK	000013	D.TDR	000000	D.VEC	040130	D.WPS	000011	D.WRI	000012	DF.CLR	000376
DF.EMP	000377	DFLBOO	042205	DIR.ALD	000025	DIR.CLU	000015	DIR.CRD	000023	DIR.EXT	000010
DIR.FGN	000020	DIR.FLG	000016	DIR.LGN	000021	DIR.LSI	000022	DIR.NAM	000000	DIR.PRO	000013
DIR.VER	000014	DIRELEN	000027	DIRIDL	000015	DM.MR	000000	DM.MW	000001	DM.RR	000002
DM.RW	000003	DOCMD	056365	DOCMD1	056375	DOCMD2	057002	DOMENU	062210	DOREAD	056145
ENDREL	043053	ENL	000212	ENTRY	042200	ERRBS	055066	ESC	000033	EXORG	053200
EXT133	056242	EXT140	056353	EXT171	057316	EXT200	061261	EXT218	063235	EXT220	063336
EXT233	064073	EXT234	064101	EXT245	072250	EXTSBC	056142	FCU	060245	FCU0	060313
FCU1	060315	FF	000014	FILLC	062175	FINDO	063353	FINDO1	063360	FINDO2	063375
FINDO3	064003	GETBS	054210	GETBS1	054243	GETBS2	054252	GETBS3	054266	GETCN1	057164
GETCN2	057200	GETCON	057156	GETNM1	061347	GETNM2	061366	GETNM3	061371	GETNUM	061344
GETSBA	053335	GETSBB	054056	HAVEO1	056134	HAVEOC	055346	HAVEOS	055265	HLCPDE	062154
I.CONFL	000004	I.CONTY	000001	I.CONWI	000003	I.CSLMD	000000	I.CUSOR	000002	ICTT1	060066
ICTT2	060075	ICTT3	060121	INITC	057137	INPORT1	056304	INPORT2	057047	INPORT3	057052
INPSTR	064325	IP.CON	000362	IP.PAD	000360	ISANM1	062101	ISANM2	062143	ISANM3	062151
ISANUM	062076	LF	000012	LSA.2	000037	LUNM	000140	M.FOX	000303	M.PAM8	000021
MI.INIR	262355	MOVEB	062162	NL	000012	NUL2	000000	NULL	000200	OCCNUM	065005
OP.CTL	000360	OP.DIG	000360	OP.SEG	000361	OP2.CTL	000362	OPCODM	000037	OSLK1	055153
OSLOOK	055146	OSNAME	064365	OUPORT1	057060	OUPORT2	057055	OVL.IN	000001	OVL.NUM	000014
OVL.RES	000002	OVL.UCS	000200	PARSB1	061105	PARSB2	061263	PARSEB	061063	QUOTE	000047
RDDATA	057046	RDHALF	056351	RDSTAT	057051	RDSTR	056162	RDSTR1	056172	RDSTR2	056222
RDSTR3	056234	RDWAIT	057062	READ2	057022	READ2A	057027	RELOC	043000	RELOC1	043011
RELOC2	043052	RI.BST	000001	RI.CON	000001	RI.DAT	000000	ROMBOOT	030000	RUBOUT	000177
S.AHI	041135	S.ALOW	041133	S.AMID	041134	S.BAUD	040344	S.BDA	041120	S.BHI	041132
S.BLOW	041130	S.BMID	041131	S.BOOTF	041034	S.CAADR	040333	S.CACC	041006	S.CCTAB	040335
S.CDB	040343	S.CFWA	040352	S.CODE	041007	S.CONFL	040332	S.CONTY	040327	S.CONWI	040331
S.CSLMD	040326	S.CUSOR	040330	S.DATC	040310	S.DATE	040277	S.DCS	041033	S.DDDTA	040366
S.DDGRP	040364	S.DDLDA	040360	S.DDLEN	040362	S.DDOPC	040370	S.DFWA	040354	S.DIREA	041016
S.DLINK	040346	S.FASER	041013	S.FCI	041021	S.GRT0	024000	S.GRT1	025000	S.GRT2	026000
S.GUP	041027	S.HIMEM	040316	S.INT	040343	S.JUMPS	041010	S.LUN	056235	S.MOUNT	041032
S.OCCR	041127	S.OFWA	040350	S.OMAX	040324	S.OSID	041126	S.OSN	041004	S.OVLE	041000

```
S.OVLFL 040371   S.OVLS  040376   S.OVSTK 041035   S.RFWA  040356   S.SCI   041024   S.SCR   041121
S.SDD   041010   S.SOVR  041146   S.SSN   041002   S.SYSM  040320   S.TIME  040312   S.UCSF  040372
S.UCSL  040374   S.USRM  040322   S.VAL   040277   SB.IDT  065104   SB.SAT  066104   SBCXLN  001000
SBERR   054125   SC.ACE  000350   SC.UART 000372   SCU     060326   SCU1    060376   SEEK0   057272
SENDB   057373   SENDB1  057374   SENDB2  060015   SKIP0   064052   SKIP1   064055   SPSAVE  065006
SS.CDB  056236   SSBA    042236   SSBB    042241   ST.ERR  000002   ST.LUN  000140   ST.PER  000001
ST.SPR  000034   STACK   042200   STACKL  001032   START   053200   STRCMP  064006   STRCP1  064021
STRCP2  064034   STRLEN  064326   STRTMP  064051   STRTXT  064327   SUPBLK  065104   SYDD    040130
T0.DNR  000004   T0.DNS  000005   T0.NIS  000001   T0.NSC  000002   T0.NST  000000   T0.NT0  000006
T0.WFT  000003   T1.BBF  000011   T1.CDE  000010   T1.DMNF 000003   T1.DTE  000006   T1.FE   000012
T1.ID   000000   T1.IDNF 000002   T1.RNF  000004   T1.SKE  000005   T1.UDE  000001   T1.WP   000007
T2.IDA  000001   T2.IFN  000002   T2.ILC  000000   TAB     000011   TO.MDS  000007   TYPEC1  060174
TYPEC2  060211   TYPEC3  060223   TYPEC4  060230   UC.2SB  000004   UC.5BW  000000   UC.6BW  000001
UC.7BW  000002   UC.8BW  000003   UC.BI   000020   UC.CTS  000020   UC.DCS  000001   UC.DDR  000002
UC.DLA  000200   UC.DR   000001   UC.DRL  000010   UC.DSR  000040   UC.DTR  000001   UC.EDA  000001
UC.EPS  000020   UC.FE   000010   UC.IID  000006   UC.IIP  000001   UC.LOO  000020   UC.MSI  000010
UC.OR   000002   UC.OU1  000004   UC.OU2  000010   UC.PE   000004   UC.PEN  000010   UC.RI   000100
UC.RLS  000200   UC.RSI  000004   UC.RTS  000002   UC.SB   000100   UC.SKP  000040   UC.TER  000004
UC.THE  000040   UC.TRE  000002   UC.TSE  000100   UCI.ER  000020   UCI.IE  000002   UCI.IR  000100
UCI.RE  000004   UCI.RO  000040   UCI.TE  000001   UDR     000000   UMI.16X 000002   UMI.1B  000100
UMI.1X  000001   UMI.2B  000300   UMI.64X 000003   UMI.HB  000200   UMI.L5  000000   UMI.L6  000004
UMI.L7  000010   UMI.L8  000014   UMI.PA  000020   UMI.PE  000040   UNK103  064065   UNK49   056303
UNK49A  056312   UNK49B  056324   UNK49C  056342   UNK59   057102   UNK59A  057112   UNK67   057315
UNK72   060035   UNK98   063352   UO.CLK  000001   UO.DDU  000002   UO.HLT  000200   UO.NFR  000100
UR.DLL  000000   UR.DLM  000001   UR.IER  000001   UR.IIR  000002   UR.LCR  000003   UR.LSR  000005
UR.MCR  000004   UR.MSR  000006   UR.RBR  000000   UR.THR  000000   USERFWA 042200   USR     000001
USR.BD  000100   USR.FE  000040   USR.OE  000020   USR.PE  000010   USR.RXR 000002   USR.TXE 000004
USR.TXR 000001   WAITCR  061272   WAITR   057117   WAITS   057336   WAITS1  057352   WAITS2  057355
WARMB   000000   WRCTRL  057054   WRDATA  057057   X.BAUD  056240   X.CONTY 056237
```

```
  $BCRC   062014        800       821      1818L     1860
  $ICTT   060044       1051      1416L     1417
  $ICTT.  060056       1416      1421L
  $ICTT.. 060104       1418      1443L
  $TYPEC. 060163       1052      1487      1511L     1541
  $TYPET  060126        769       789       828       874       882       899       930       947       997      1480L
                       1497      1756      1963      1972
  $TYPET. 060134       1481      1485L     1493      2020
  $TYPET1 060154       1489      1497L
  $UDD    031157        625E     2017
  .ABUSS  040024        180E
  .ALARM  002136        153E
  .ALEDS  040013        178E
  .CRC    002347        161E
  .CRCSUM 040027        181E
  .CTC    002172        155E
  .CTL2FL 040066        187E
  .CTLFLG 040011        177E
  .DLEDS  040021        179E
  .DLY    000053        150E     1658
  .DOD    003122        164E
  .DODA   003356        166E
  .DSPMOD 040007        175E
  .DSPROT 040006        174E
  .DUMP   001374        152E
  .HORN   002140        154E
  .IDENT  000000        149E
  .IOWRK  040002        172E
  .LOAD   001267        151E
  .MFLAG  040010        176E
  .NMIRET 040064        186E
  .PCHL   002264        157E
  .RCK    003260        165E
  .REGI   040005        173E
  .REGPTR 040035        184E
  .RNB    002331        160E
  .RNP    002325        159E
  .SRS    002265        158E
  .START  040000        171E
  .TICCNT 040033        183E
  .TPERR  002205        156E
  .TPERRX 040031        182E
  .UIVEC  040037        185E
  .WNB    003024        163E
  .WNP    003017        162E
  AC.DLY  000156        330E     1657
  AIO.CGN 041047        592L
  AIO.CHA 041116        607L
  AIO.CNT 041111        603L
  AIO.CSI 041050        593L
  AIO.DDA 041041        588E
  AIO.DES 041055        597L
  AIO.DEV 041057        598L
  AIO.DIR 041062        601L
  AIO.DTA 041053        596L
  AIO.EOF 041113        605L
  AIO.EOM 041112        604L
```

```
  AIO.FLG 041043        589L
  AIO.GRT 041044        590L
  AIO.LGN 041051        594L
  AIO.LSI 041052        595L
  AIO.SPG 041046        591L
  AIO.TFP 041114        606L
  AIO.UNI 041061        599L
  AIO.VEC 041040        587L
  ASKBS   054373        865     881L     903      934      951     1001
  BASE    000170        194E    1110     1193     1197     1201    1205
  BAU.96  000014        629E    1568
  BC.EDT  000002        205E    1271
  BC.IE   000040        203E
  BC.RST  000020        204E    1245
  BC.SEL  000100        202E    1266
  BELL    000007        304E     770      790      829      875     900      931      948      998
  BKSP    000010        306E
  BLDCDR  057227       1092    1286L    1314
  BOOLEN  011000        627E     628      702      703      703     704      736      990
  BOOT.P  000001        567E
  BS.BSY  000010        217E    1085     1248     1261     1269
  BS.COM  000020        216E    1184     1185     1213     1215    1235     1248     1372     1374     1382
                       1384    1385
  BS.DAT  000000        215E
  BS.DTD  000100        210E
  BS.HID  000001        220E
  BS.IN   000000        211E
  BS.INT  000004        218E    1085
  BS.LMB  000040        213E    1184     1185
  BS.MTY  000020        214E
  BS.OUT  000100        212E    1184     1213     1235     1248    1249     1372     1374     1382     1384
  BS.PE   000002        219E    1085
  BS.REQ  000200        209E    1085     1184     1185     1213    1215     1235     1237     1248     1372
                       1374    1382
  BSLEN   061270       1690    1699     1716     1723     1751L
  BSPTR   061266       1687    1702     1719     1722     1729    1750L
  BUFFER  065010       2003    2008     2019     2139L
  C.STX   000002        308E
  C.SYN   000026        307E
  CB.CLI  000100         95E     118
  CB.MTL  000040         94E
  CB.SPK  000200         96E
  CB.SSI  000020         93E
  CB2.CLI 000002         99E
  CB2.ORG 000040        100E
  CB2.SID 000100        101E
  CB2.SSI 000001         98E
  CDB.H84 000001        510E    1422     1444     1513     1576    1607
  CDB.H85 000000        509E    1564
  CDRBLK  057014       1156    1167L    1273     1289     1294    1296     1298     1300     1302
  CHKSBA  054031        785     797L
  CHKSBB  054100        813     818L
  CLASS0  000000        234E     244      245      246      247     248      249      250      251      252
                        253     254      255
  CLASS1  000040        235E     259
  CLASS6  000300        236E     263
  CLASSM  000340        232E
```

```
  CN.170M 000014        136E
  CN.174M 000003        135E
  CN.ABO  000200        140E
  CN.BAU  000100        139E
  CN.MEM  000040        138E
  CN.PRI  000020        137E
  CND.H17 000000        142E
  CND.H47 000001        144E
  CND.NDI 000000        143E
  CO.FLG  000001        487E
  CR      000015        300E     1055      1498      1533      1545
  CS.FLG  000200        488E
  CS.SBA  042262        679L      801
  CS.SBB  042264        680L      822
  CSL.CHR 000001        464E
  CSL.ECH 000200        461E
  CSL.RAW 000004        462E
  CSL.WRP 000002        463E
  CTLA    000001        315E
  CTLB    000002        316E
  CTLC    000003        317E
  CTLD    000004        318E
  CTLO    000017        319E
  CTLP    000020        320E
  CTLQ    000021        321E
  CTLS    000023        322E
  CTLZ    000032        323E
  CTP.2SB 000010        473E     1574      1620      1621      1651      1653
  CTP.BKM 000002        474E
  CTP.BKS 000200        469E
  CTP.FF  000100        470E
  CTP.MLI 000040        471E
  CTP.MLO 000020        472E
  CTP.TAB 000001        475E
  D.CON   040110         29L
  D.CP3   000040        259E
  D.CTF   000005        249E
  D.FBS   000007        251E
  D.FFD   000300        263E
  D.FOR   000004        248E
  D.FT    000006        250E
  D.RAM   040240         32L
  D.REA   000010        252E     1091
  D.REC   000001        245E
  D.RSE   000003        247E
  D.RSY   000002        246E
  D.SEK   000013        255E     1310
  D.TDR   000000        244E
  D.VEC   040130         31L
  D.WPS   000011        253E
  D.WRI   000012        254E
  DF.CLR  000376         51E
  DF.EMP  000377         50E
  DFLBOO  042205        666L      852       870
  DIR.ALD 000025         66L
  DIR.CLU 000015         59L
  DIR.CRD 000023         65L
```

```
  DIR.EXT 000010         54L
  DIR.FGN 000020         62L
  DIR.FLG 000016         60L
  DIR.LGN 000021         63L
  DIR.LSI 000022         64L
  DIR.NAM 000000         53L
  DIR.PRO 000013         55L
  DIR.VER 000014         56L
  DIRELEN 000027         68E      601
  DIRIDL  000015         57E
  DM.MR   000000        108E
  DM.MW   000001        109E
  DM.RR   000002        110E
  DM.RW   000003        111E
  DOCMD   056365       1095      1150L     1316
  DOCMD1  056375       1152      1156L
  DOCMD2  057002       1158L     1162
  DOMENU  062210        881      1963L
  DOREAD  056145        764       784       812       992      1017L
  ENDREL  043053        704       726E      763
  ENL     000212        313E
  ENTRY   042200        651L     2149
  ERRBS   055066        848       899L
  ESC     000033        311E     1964      1965
  EXORG   053200        628E      763       763
  EXT133  056242       1023      1084L
  EXT140  056353       1129      1136L
  EXT171  057316       1320L     1323
  EXT200  061261       1728      1737      1744L
  EXT218  063235       1986L     2030
  EXT220  063336       1990      1997      2024L
  EXT233  064073       2115L     2125
  EXT234  064101       2117      2119L
  EXT245  072250        741      2147E
  EXTSBC  056142        762      1008L
  FCU     060245        743      1557L
  FCU0    060313       1572      1576L
  FCU1    060315       1566      1582L
  FF      000014        314E
  FILLC   062175       1697      1949L     1957      2004
  FINDO   063353        942      2041L
  FINDO1  063360       2044L     2055
  FINDO2  063375       2049      2052L
  FINDO3  064003       2047      2059L
  GETBS   054210        803       824       838L
  GETBS1  054243        841       850L
  GETBS2  054252        853L      859
  GETBS3  054266        855       863L
  GETCN1  057164       1256      1260L
  GETCN2  057200       1268L     1270
  GETCON  057156       1094      1255L     1315
  GETNM1  061347       1772L     1799
  GETNM2  061366       1778      1781L
  GETNM3  061371       1780      1786L
  GETNUM  061344       1738      1770L
  GETSBA  053335        754       758       765       781L
  GETSBB  054056        793       809L
```

```
  HAVEO1  056134        993      1005L
  HAVEOC  055346        943       957L
  HAVEOS  055265        920       941L
  HLCPDE  062154        802       823       840      1709      1918L
  I.CONFL 000004        490E      491
  I.CONTY 000001        477E      478
  I.CONWI 000003        483E      484
  I.CSLMD 000000        466E
  I.CUSOR 000002        480E      481
  ICTT1   060066       1427L
  ICTT2   060075       1423      1436L
  ICTT3   060121       1445      1455L
  INITC   057137        744      1243L
  INPORT1 056304        718      1111E
  INPORT2 057047        719      1194E
  INPORT3 057052        722      1198E
  INPSTR  064325        887      1758      2133L
  IP.CON  000362         84E
  IP.PAD  000360         80E
  ISANM1  062101       1873L     1896
  ISANM2  062143       1881      1885      1893L
  ISANM3  062151       1879      1883      1887      1889      1898L
  ISANUM  062076       1703      1871L
  LF      000012        301E     1498
  LSA.2   000037        240E
  LUNM    000140        239E
  M.FOX   000303        128E
  M.PAM8  000021        127E
  MI.INIR 262355        632E     1144
  MOVEB   062162       1721      1933L     1941      2010
  NL      000012        312E      313
  NUL2    000000        303E
  NULL    000200        302E
  OCCNUM  065005        941       959      1692      1743      2137L
  OP.CTL  000360         81E
  OP.DIG  000360         82E
  OP.SEG  000361         83E
  OP2.CTL 000362         85E
  OPCODM  000037        238E
  OSLK1   055153        913L      926
  OSLOOK  055146        847       872       895       911L
  OSNAME  064365        916      1696      1720      2136L
  OUPORT1 057060        720      1206E
  OUPORT2 057055        723      1202E
  OVL.IN  000001        534E
  OVL.NUM 000014        536E
  OVL.RES 000002        535E
  OVL.UCS 000200        537E
  PARSB1  061105       1678      1687L
  PARSB2  061263       1672      1684      1706      1710      1734      1739      1742      1747L
  PARSEB  061063        846       871       894      1670L
  QUOTE   000047        309E
  RDDATA  057046       1187      1193L     1217      1239
  RDHALF  056351       1116      1117      1125      1135L
  RDSTAT  057051       1084      1183      1197L     1212      1234      1268      1360      1371      1381      1389
  RDSTR   056162        889      1040L     1760
  RDSTR1  056172       1049L     1070
```

```
  RDSTR2  056222       1058      1063L
  RDSTR3  056234       1056      1071L
  RDWAIT  057062       1181      1212L     1214      1216
  READ2   057022       1180L     1320
  READ2A  057027       1183L     1186
  RELOC   043000        651       702L
  RELOC1  043011        705L      712
  RELOC2  043052        716       724L
  RI.BST  000001        198E      1197
  RI.CON  000001        197E      1201
  RI.DAT  000000        196E      1193      1205
  ROMBOOT 030000         24E
  RUBOUT  000177        305E
  S.AHI   041135        647E       978
  S.ALOW  041133        645E       972
  S.AMID  041134        646E       975
  S.BAUD  040344        511L
  S.BDA   041120        609L       714
  S.BHI   041132        644E       968
  S.BLOW  041130        642E       962
  S.BMID  041131        643E       965
  S.BOOTF 041034        566L
  S.CAADR 040333        494L
  S.CACC  041006        550L
  S.CCTAB 040335        495L
  S.CDB   040343        508L
  S.CFWA  040352        518L
  S.CODE  041007        551L
  S.CONFL 040332        492L
  S.CONTY 040327        479L
  S.CONWI 040331        485L
  S.CSLMD 040326        467L      478       481       484       491
  S.CUSOR 040330        482L
  S.DATC  040310        448L
  S.DATE  040277        447L
  S.DCS   041033        564L
  S.DDDTA 040366        529L
  S.DDGRP 040364        526L
  S.DDLDA 040360        524L
  S.DDLEN 040362        525L
  S.DDOPC 040370        530L
  S.DFWA  040354        519L
  S.DIREA 041016        558L
  S.DLINK 040346        516L
  S.FASER 041013        557L
  S.FCI   041021        559L
  S.GRT0  024000         20E
  S.GRT1  025000         21E
  S.GRT2  026000         22E
  S.GUP   041027        561L
  S.HIMEM 040316        450L
  S.INT   040343         34L       504
  S.JUMPS 041010        555L
  S.LUN   056235       1075L     1290
  S.MOUNT 041032        563L
  S.OCCR  041127        641E       960
  S.OFWA  040350        517L
```

```
  S.OMAX  040324        456L
  S.OSID  041126        640E      958
  S.OSN   041004        546L
  S.OVLE  041000        543L
  S.OVLFL 040371        539L
  S.OVLS  040376        542L
  S.OVSTK 041035        571L
  S.RFWA  040356        520L
  S.SCI   041024        560L
  S.SCR   041121        610L
  S.SDD   041010        556L
  S.SOVR  041146         36L       38
  S.SSN   041002        545L
  S.SYSM  040320        452L
  S.TIME  040312        449L
  S.UCSF  040372        540L
  S.UCSL  040374        541L
  S.USRM  040322        454L
  S.VAL   040277         33L      445
  SB.IDT  065104        911      1983      2142L
  SB.SAT  066104       2043      2114      2143L
  SBCXLN  001000        631E      751
  SBERR   054125        814       828L
  SC.ACE  000350        329E     1436      1455      1527      1531      1558      1561      1562      1634      1639
                       1641      1644      1646      1649      1656      1659      1660      1662
  SC.UART 000372        399E     1427      1449      1518      1522      1559      1613      1614      1615      1616
                       1618      1627      1629
  SCU     060326       1584      1606L
  SCU1    060376       1608      1634L     1636
  SEEK0   057272       1243      1310L
  SENDB   057373       1159      1274      1370L
  SENDB1  057374       1371L     1373      1375
  SENDB2  060015       1381L     1383      1386
  SKIP0   064052        843      2102L
  SKIP1   064055       2103L     2108
  SPSAVE  065006        740       838       844      2138L
  SS.CDB  056236       1076L     1421      1443      1512      1583      1606
  SSBA    042236        669L      781
  SSBB    042241        670L      809
  ST.ERR  000002        226E
  ST.LUN  000140        224E
  ST.PER  000001        227E
  ST.SPR  000034        225E
  STACK   042200         40E
  STACKL  001032         38E
  START   053200        738L
  STRCMP  064006        917      2070L     2090
  STRCP1  064021       2075      2077L
  STRCP2  064034       2080      2082L
  STRLEN  064326        890      2134L
  STRTMP  064051       2077      2083      2091L
  STRTXT  064327        893      2135L
  SUPBLK  065104        782       799       810       820      2141E
  SYDD    040130         30E
  T0.DNR  000004        271E
  T0.DNS  000005        272E
  T0.NIS  000001        268E
```

```
  T0.NSC  000002        269E
  T0.NST  000000        267E
  T0.NT0  000006        273E
  T0.WFT  000003        270E
  T1.BBF  000011        287E
  T1.CDE  000010        286E
  T1.DMNF 000003        281E
  T1.DTE  000006        284E
  T1.FE   000012        288E
  T1.ID   000000        278E
  T1.IDNF 000002        280E
  T1.RNF  000004        282E
  T1.SKE  000005        283E
  T1.UDE  000001        279E
  T1.WP   000007        285E
  T2.IDA  000001        293E
  T2.IFN  000002        294E
  T2.ILC  000000        292E
  TAB     000011        310E
  TO.MDS  000007        274E
  TYPEC1  060174       1518L     1520
  TYPEC2  060211       1514      1527L     1529
  TYPEC3  060223       1523      1533L
  TYPEC4  060230       1539L     1544
  UC.2SB  000004        355E     1653      1654
  UC.5BW  000000        351E
  UC.6BW  000001        352E
  UC.7BW  000002        353E
  UC.8BW  000003        354E     1560      1563      1655
  UC.BI   000020        374E
  UC.CTS  000020        383E
  UC.DCS  000001        379E
  UC.DDR  000002        380E
  UC.DLA  000200        360E     1643
  UC.DR   000001        370E     1437
  UC.DRL  000010        382E
  UC.DSR  000040        384E
  UC.DTR  000001        363E
  UC.EDA  000001        341E
  UC.EPS  000020        357E
  UC.FE   000010        373E
  UC.IID  000006        348E
  UC.IIP  000001        347E
  UC.LOO  000020        367E     1640      1661
  UC.MSI  000010        344E
  UC.OR   000002        371E
  UC.OU1  000004        365E
  UC.OU2  000010        366E
  UC.PE   000004        372E
  UC.PEN  000010        356E
  UC.RI   000100        385E
  UC.RLS  000200        386E
  UC.RSI  000004        343E
  UC.RTS  000002        364E
  UC.SB   000100        359E
  UC.SKP  000040        358E
  UC.TER  000004        381E
```

```
  UC.THE  000040        375E     1528
  UC.TRE  000002        342E
  UC.TSE  000100        376E     1635
  UCI.ER  000020        421E     1628
  UCI.IE  000002        423E
  UCI.IR  000100        419E     1617
  UCI.RE  000004        422E     1628
  UCI.RO  000040        420E
  UCI.TE  000001        424E     1628
  UDR     000000        396E     1449      1522
  UMI.16X 000002        414E     1626
  UMI.1B  000100        404E     1621      1626
  UMI.1X  000001        413E
  UMI.2B  000300        406E     1621
  UMI.64X 000003        415E
  UMI.HB  000200        405E
  UMI.L5  000000        409E
  UMI.L6  000004        410E
  UMI.L7  000010        411E
  UMI.L8  000014        412E     1626
  UMI.PA  000020        408E
  UMI.PE  000040        407E
  UNK103  064065       1994      2112L
  UNK49   056303       1099      1110L
  UNK49A  056312       1116L     1119
  UNK49B  056324       1115      1120L
  UNK49C  056342       1124      1128L
  UNK59   057102       1090      1221L
  UNK59A  057112       1224      1227L
  UNK67   057315       1024      1317      1319L
  UNK72   060035       1275      1389L     1390
  UNK98   063352       1999      2013      2034L
  UO.CLK  000001        120E
  UO.DDU  000002        119E
  UO.HLT  000200        117E
  UO.NFR  000100        118E
  UR.DLL  000000        336E     1646
  UR.DLM  000001        338E     1649
  UR.IER  000001        340E     1558      1639
  UR.IIR  000002        346E
  UR.LCR  000003        350E     1561      1562      1644      1656
  UR.LSR  000005        369E     1436      1527      1634
  UR.MCR  000004        362E     1641      1660      1662
  UR.MSR  000006        378E
  UR.RBR  000000        332E     1455      1659
  UR.THR  000000        334E     1531
  USERFWA 042200         41E      628       649       703       736       839       991      1005      1006
  USR     000001        397E     1427      1518      1559      1613      1614      1615      1616      1618      1627
                       1629
  USR.BD  000100        428E
  USR.FE  000040        429E
  USR.OE  000020        430E
  USR.PE  000010        431E
  USR.RXR 000002        433E     1428
  USR.TXE 000004        432E
  USR.TXR 000001        434E     1519
  WAITCR  061272        772       792       831       877       902       933       950      1000      1756L
```

```
  WAITR   057117       1136      1234L     1236      1238
  WAITS   057336       1247      1260      1337L
  WAITS1  057352       1345      1354L
  WAITS2  057355       1355L     1363
  WARMB   000000        630E      773       832
  WRCTRL  057054       1201L     1246      1267      1272
  WRDATA  057057       1205L     1377
  X.BAUD  056240       1078L     1582      1642
  X.CONTY 056237       1077L     1573      1575      1619      1650
```

36273 Bytes Free