UNIVERSAL ASSEMBLER VERSION 1.2 JANUARY 4, 1978 (IN-HOUSE)

```
COMMAND LINE WAS: SNAP3 1800MACR,,,1800MACR/APLX

INCLUSION A: 1800/MAC
INCLUSION B: MDEF1800/TXT
INCLUSION C: BDEF1800/TXT
INCLUSION D: ODEF1800/TXT
INCLUSION E: STRUCT/TXT
INCLUSION F: 1800REST/TXT
INCLUSION G: 1800BOOT/TXT
INCLUSION H: 1800DIAG/TXT
INCLUSION I: 1800DSKR/TXT
INCLUSION J: 1800DBUG/TXT
INCLUSION K: 1800DSPR/TXT
INCLUSION L: 1800MOVI/TXT

PROGRAM NAME: 1800MACR

PROGRAM ADDRESS BLOCKS:      170000    /ABSOLUTE/  SIZE=000000  (ABS)
                             167400    /SYSIVR/    SIZE=000400  (ABS)
                             170000    /SYSROM/    SIZE=000047  (ABS)
                             153600    /SDBGWS/    SIZE=000051  (ABS)
                             156600    /$DATA$/    SIZE=000063  (ABS)

PRIMARY TRANSFER ADDRESS:    170000

FIXED ENTRY POINTS:

$MACROM   000011    $MACVER   000040    $DISKBUF  154000    $DOSEXT   156000
$DATA$    156600    $LOGDRV   156617    $SECTOR   156620    $TRACK    156621
$DEVADR   156622    $DRVNUM   156623    $DRVTAB   156624    $KEYCHAR  170055
$KBDSINI  170060    $CHARLOD  170063    $DSPINIT  170066    $DISPLAY  170071
$DISPDOS  170074    D$CBL     170077    D$CURSES  170102    D$BLNKDE  170105
D$BLINK   170110    DO$FNC60  170113    DO$FNC61  170116    DO$FNC62  170121
DO$FNC63  170124    DO$FNC64  170127    DO$FNC65  170132    DO$FNC66  170135
DO$FNC67  170140    DO$FNC68  170143    DO$FNC69  170146    DO$FNC6A  170151
DO$FNC6B  170154    DO$FNC6C  170157    $SEEK     170162    $STEP     170165
$READ     170170    $WRITE    170173    $ONLINE   170176    $RESTORE  170201
$DOSDRIV  170204    $DOIO     170207    $BUFIO    170212    $STATUS   170215
$FDSTAT   170220    $NS       000203    $ES       000003    $HD       000224
$VA       000213    $CK       000207    $EEOL     000022    $BP       000007
$OI       000200    $H        000011    $V        000013    $EL       000015
$EEOF     000021    $RU       000023    $RD       000024    $F        000033
$HA       000211    $NL       000215    $HU       000223    $O        000233
$OS       000100    $OW       000040    $MACVRS   177777

UNUSED LABELS:

KEYINF
```

```
                        .
                        . 1800 MACRO ROM PROGRAM - MAIN MODULE       JUN 14, 1978    12:05    HSP/HJS
                        .
000011  $MACROM*  EQU     9               MACRO ROM VERSION NUMBER
000040  $MACVER*  EQU     ' '             MACRO ROM REVISION LETTER
                        *
                        . 9       HJS     20 JUL 78       VERSION 9 FINAL RELEASE
                        . 9.J     HJS     21 JUN 78       FIX TWO RESTART BUGS (STL & BETA)
                        . 9.I     HJS     14 JUN 78       FIX BUGS & ADD MORE SPACE IN CODE
                        . 9.H     HJS     10 MAY 78       CORRECT BUGS & SETUP CRC FOR RELEASE
                        . 9.G     HJS     24 MAR 78       ADD RIM BUFFER TEST
                        . 9.F     HJS     20 MAR 78       ALLOW ROM TYPE 2 (3800) CODE GENERATION
                        . 9.E     HJS     16 MAR 78       ADD NEW VECTOR, DISKETTE LOCKUP CONDITION
                        . 9.C     HJS     13 FEB 78       FIX 'UNI-KEY' UP & COMMENTS
                        . 9.B     HJS      4 FEB 78       CORRECT AND INSERT 'UNIVERSAL KEYBOARD'
                        . 9.A     HJS     30 JAN 78       DISPLAY AND POR FIXUPS
                        .
000002  PN1800    EQU     2               1800 PROCESSOR NUMBER (FROM INFO)
                        .
                  SNAPOPT 6               INSURE 6600 CODE AVAILABLE
                  LIST    F
                        .
000000  ASCII     EQU     0               NON-ZERO => ASCII KEYIN IN DEBUG
000000  ORIGINI   EQU     0               NON-ZERO => ORIGIN CAPABILITIES IN DEBUG
000000  STACKP    EQU     0               NON-ZERO => STACK PUSH/POP OP'S IN DEBUG
                        .
000002  RAMROM    EQU     2               ZERO => ROM IN RAM AT 0010000
                        .                 ONE  => ROM IN RAM AT 0120000
                        .                 TWO  => ROM IN ROM AT 0170000
                        .
000000  ROMTYPE   EQU     0               ZERO => 1800 STANDARD MACRO-ROM
                        .                 ONE  => APF SPECIAL VERSION
                        .                 TWO  => RIMDLL VERSION (3800)
                        .                 THREE=> CYNTHIA &/OR RIM BOOT
                        .
000000  ROMCRC    EQU     0               SET THIS EQU TO THE CRC OF BYTES 0170000
                        .                 TO 0177774, STARTING FROM AN INITIAL -1
                        .                 PUT THE CRC IN BYTES 0177775 & 6 (MSB/LSB)
                        .
000013  BL        EQU     11              NUMBER OF BOTTOM LINE FOR DOS FUNCTION 6
000000  LC        EQU     0               NUMBER OF LEFT COLUMN FOR DOS FUNCTION 6
                        .
000106  NECP      EQU     70              'NO ENTRY' CURSOR POSITION FOR DEBUG
000052  OPCODEBP  EQU     052             BREAK POINT OPCODE
```

```
45:
46:                     *
47:                     * GET ALL THE DEFINITIONS
48:                     *
49:                             INC     1800/MAC        INSTRUCTION DEFINITIONS
50:                             INC     MDEF1800        MEMORY AND BIT DEFINITIONS
51:                             INC     ODEF1800        DISKETTE DEFINITIONS
52:                             LIST    -I
53:                             INC     STRUCT          STRUCTURED ASSEMBLER MACROS (FOR RIMDLL)
                                LIST    I
```

- WORKING STORAGE DEFINITIONS -

```
                        *
                        * DEFINE THE DEBUG WORKING STORAGE LOCATIONS
                        *
153600  SDBGWS  ORG     SDBGWS
153600          USE     SDBGWS
                *
153600  CURADR  SK      2               LAST MEMORY LOCATION ADDRESSED
153602  CUROUT  SK      1               LAST CHARACTER WRITTEN TO DEVICE
153603  CURSTA  SK      1               LAST STATUS FROM A DEVICE
153604  MODVAL  SK      2               LAST 'MODIFY' VALUE GIVEN
153606  KEYINF  SK      1               ASCII KEYIN MODE FLAG (0=>NORMAL)
                *
153607  OLDTOS  SKIP    2               OLD TOP OF STACK
153611  OLDREGS SKIP    2               POINTER TO ALPHA OR BETA REGISTERS
                *
                        IFC     ORIGINI
000012  BPTLN   EQU     10              MAXIMUM NUMBER OF BREAKPOINTS
                        XIF
                        IFS     ORIGINI
                BPTLN   EQU     6       MAXIMUM NUMBER OF BREAKPOINTS
                        XIF
000003  BPTES   EQU     3               THERE ARE (3) BYTES PER ENTRY
153613  BPTABL  SK      BPTLN*BPTES     BREAK POINT TABLE
153651  BPTABE                          END OF BREAKPOINT TABLE
                *
                        IFS     ORIGINI
                OTABLN  EQU     6       MAXIMUM NUMBER OF ORIGINS
                CURORG  SKIP    2       CURRENT ORIGIN VALUE
                CUROSN  SKIP    1       CURRENT ORIGIN SELECT NUMBER
                OTABL   SKIP    OTABLN*2        ORIGIN TABLE
                OPTABE                  END OF ORIGIN TABLE
                        XIF
                *
                * THE REST OF THE PAGE IS USED FOR DEBUG'S REGISTERS AND STACK
                *
                        IFGE    $,SDBGWS+(SESAVB.AND.0177)
                        ERR     DEBUG WORKING STORAGE AREA TOO LARGE!
                        XIF
                *
                * PUT THE DATA AREA DEFINITIONS INTO THE ENTRY POINT TABLE
                *
154000  $DISKBUF* EQU   DOSBUF          DOS LF0-LF3
                *
156000  $DOSEXT* EQU    DOSEXT          DOS EXTENSION AREA
```

```
100,                            *
101,                            . DEFINE THE DISK WORKING STORAGE MEMORY LOCATIONS
102,                            .
103,  156600                    $DATA$    ORG     DOSEXT+0600
104,  156600                              USE     $DATA$
105,
106,  156600                    $DATA$*   EQU     $               DEFINE CURRENT DATAPAGE
107,  156600                              SKIP    3               * NOT USED *
108,  156603                    $$AVXA    SKIP    2               TEMP HOLD FOR (XA)
109,  156605                    $$AVBC    SKIP    2               TEMP HOLD FOR (BC)
110,  156616                    $DSKREGS  EQU     $+7             DISK DRIVER REGISTER SAVE AREA
111,  156607                              SKIP    8
112,  156617                    $LOGDRV*  SKIP    1               CURRENT DOS LOGICAL DRIVE (INIT =1)
113,                            *
114,                            . THE FOLLOWING TWO VARIABLES ARE ORDER DEPENDENT
115,                            .
116,  156620                    $SECTOR*  SKIP    1               CURRENT SECTOR (ORDER DEPENDENT)
117,  156621                    $TRACK*   SKIP    1               CURRENT TRACK (ORDER DEPENDENT)
118,                            .
119,  156622                    $DEVADR*  SKIP    1               FLOPPY PHYSICAL DEVICE ADDRESS
120,  156623                    $DRVNUM*  SKIP    1               FLOPPY LOGICAL DRIVE (FODR0/FODR1)
121,                            *
122,  156624                    $DRVTAB*
123,                            .
124,                            . CURRENT SECTOR/TRACK FOR EACH DOS LOGICAL DRIVE $SECTOR/$TRACK
125,                            .
126,  000007                    $DRVTBLN  EQU     7               .AND. MASK FOR MAX # DOS DRIVES
127,                                      RPT     $DRVTBLN+1      LOGICAL DRIVES 0-7
128,                                      SKIP    2
129,  000044                    $DATAL    EQU     $-$DATA$        LENGTH OF VARIABLES TO BE INITIALIZED.
130,                            *
131,                            . DISPLAY DRIVER SCRIBBLE AREAS
132,                            .
133,  156644                    ROWS      SKIP    7               "CHARLOD" WORKING AREA
134,  156662                    DSPREGSA  EQU     $+7             DISPLAY INTERFACE REG SAVE AREA
135,  156653                              SKIP    8
136,                            .
137,                            . --- THE REST OF THE PAGE IS NOT USED ---
138,                            .
139,                                      LIST    I
140,                            .
141.                                      INC     1800REST
```

```
                                *
                                * 1800 MACRO ROM RESTART MODULE              JUN 20, 1978   13:20    HSP/HJS
                                *
                                * PHYSICALLY LOCATE THE CODE FOR THE ROM MAKING PROGRAM
                                *
                                          IFEQ      RAMROM,0
                                          SET       010000
                                          LOC       SYSROM
                                          IFEQ      ROMTYPE,2
                                          ERR       RIM-DLL MODULE WILL CAUSE ASSEMBLY ERRORS
                                          XIF
                                *
                                          IFEQ      RAMROM,1
                                          SET       0120000
                                          LOC       SYSROM
                                          IFEQ      ROMTYPE,2
                                          ERR       RIM-DLL MODULE WILL CAUSE ASSEMBLY ERRORS
                                          XIF
                                *
                                          IFEQ      RAMROM,2
                                          SET       0170000
                                          XIF
                                *
170000                          MACROM
                                .SRPOWER
170000  104 266 363                       JUMP      POWERUP         POWER-UP ENTRYPOINT BY JUMP!
                                .SRRSTRT
170003  104 002 364                       JUMP      RESTART         RESTART ENTRYPOINT BY JUMP!
                                .SRSYSMF
                                *                                   SECOND FAULT DURING FAULT DETECTION!
                                *                                   ASSUME ALL SYSTEM MEMORY & CONTROL BAD
170006  036 357                           LD        SESTACK>8        (SAVE BYTE & POINT INTO STACK AREA)
170010  343                               LED                        (ANYWHERE WILL DO IN THE STACK)
                                *                                   NOTE: D-VALUE HAPPENS TO MAKE THIS WORK
170011  174 065                 SYSMOV    DE                       USE STANDARD STACK AS A DEFAULT STACK
170013  046 274 036 361                   DE        SYSPAT         ! RETURN ADDRESS FROM SUBROUTINE !
170017  104 005 373                       JMP       MTSETX         TO INIT STL, BASING, & INTERRUPT VECTOR
                                .SRMEMPE
170022  111 153                           CLICKR                   MEMORY PARITY ERROR CAUSES CLICK
170024  104 000 357                       JMP       SVMEMP         THEN JUMP TO THE RAM VECTOR
                                .SRCLICKR
170027  253                             ACLATT    3                SET LOUDNESS TO MAXIMUM AND
170030  210 254 360                     ACJUMP    CLICKROM         GO TO CLICK ROM CODE
                                .SRSTPE
170033  111 153                           CLICKR                   SECTOR TABLE PARITY ERROR CAUSES CLICK
170035  104 074 357                       JMP       SVSTPAR        THEN JUMP TO THE RAM VECTOR
                                *
170040  111 050 000 000         ONEMSA    EJMP      0              RAM SERVICE INTERRUPT DOES EI AND JUMP
                                *                                   (USE WASTED SPACE)
                                *
                                .SRTMOUT  JMP       TIMOUT         I-DECODE PROCESSOR TIMEOUT ERROR
                                          IFNE      $+3,SRNEXTAL
```

```
55,F                                        ERR                         ROM VECTOR ADDRESSING MIS-MATCH ERROR
56,F                                        XIF
57,F
58,F 170044  106 242 372          TIMOUT    CALL      SSTATE            (PUT HERE TO FILL THE SPACE)
59,F 170047  066 014                        LL        TIMOUTM           GET ADDRESS OF MESSAGE
60,F 170051  104 055 362                    JMP       ERROR                (USING WASTED SPACE)
```

- JUMP VECTORS TO EXTERNALLY USABLE ROUTINES AND ADDRESSES OF USEABLE TABLES -

```
                        .
                        DC      0               * NOT USED *
                        .
                        IFNE    $,0170055       * MAGIC NUMBER *
                        ERR     JUMP VECTOR ADDRESSING ERROR !
                        XIF
                        .
                        *
                        . JUMP VECTOR TO EXTERNALLY USABLE DISPLAY / KEYBOARD ROUTINES
                        .
                        *
0170055 104 143 373     $KEYCHAR* JUMP  DOSF62N         DOSFUNC 6 SUB 2 WITHOUT BLINK 'DE'
0170060 104 117 362     $KBDSINI* JUMP  KBDSPINI        INIT DISPLAY AND KEYBOARD CONTROL INFO
                        .
0170063 104 207 362     $CHARLOD* JUMP  CHARLOD         LOAD CHARACTER SET SUBROUTINE
0170066 104 004 374     $DSPINIT* JUMP  DSPINIT         INITIALIZE THE DISPLAY
0170071 104 341 374     $DISPLAY* JUMP  DISPLAY         1800/1500 DISPLAY ROUTINE
0170074 104 337 374     $DISPDOS* JUMP  DISPDOS         DOS COMPATIBLE DISPLAY ROUTINE
                        *
0170077 104 312 374     D$CBL*    JUMP  DSPCBL          COMPUTE A DISPLAY BUFFER LOCATION
0170102 104 237 374     D$CURSES* JUMP  CURSES          SUSPEND THE CURSOR
0170105 104 221 373     D$BLNKDE* JUMP  BLINKDE         BLINK THE CURSOR AT (DE)
0170110 104 244 373     D$BLINK*  JUMP  BLINK           BLINK THE CURSOR AT "SECPOS"
                        *
0170113 104 077 373     DO$FNC60* JUMP  DOSF60          DOS FUNCTION 6 SUBFUNCTION 0
0170116 104 125 373     DO$FNC61* JUMP  DOSF61          DOS FUNCTION 6 SUBFUNCTION 1
0170121 104 140 373     DO$FNC62* JUMP  DOSF62          DOS FUNCTION 6 SUBFUNCTION 2
0170124 104 171 373     DO$FNC63* JUMP  DOSF63          DOS FUNCTION 6 SUBFUNCTION 3
0170127 104 177 373     DO$FNC64* JUMP  DOSF64          DOS FUNCTION 6 SUBFUNCTION 4
0170132 104 204 373     DO$FNC65* JUMP  DOSF65          DOS FUNCTION 6 SUBFUNCTION 5
0170135 104 211 373     DO$FNC66* JUMP  DOSF66          DOS FUNCTION 6 SUBFUNCTION 6
0170140 104 345 373     DO$FNC67* JUMP  DOSF67          DOS FUNCTION 6 SUBFUNCTION 7
0170143 104 046 374     DO$FNC68* JUMP  DOSF68          DOS FUNCTION 6 SUBFUNCTION 8
0170146 104 105 374     DO$FNC69* JUMP  DOSF69          DOS FUNCTION 6 SUBFUNCTION 9
0170151 104 144 374     DO$FNC6A* JUMP  DOSF610         DOS FUNCTION 6 SUBFUNCTION 10
0170154 104 204 374     DO$FNC6B* JUMP  DOSF611         DOS FUNCTION 6 SUBFUNCTION 11
0170157 104 216 374     DO$FNC6C* JUMP  DOSF612         DOS FUNCTION 6 SUBFUNCTION 12 * NEW *
```

- JUMP VECTORS TO EXTERNALLY USABLE ROUTINES AND ADDRESSES OF USEABLE TABLES -

```
                                        +
                                                IFLE    ROMTYPE,1
                                        .
                                        . JUMP VECTOR TO EXTERNALLY USABLE DISKETTE ROUTINES
                                        .
170162  104 201 365     $SEEK*      JUMP    $$SEEK          SEEK TO THE TRACK IN (D)
170165  104 252 365     $STEP*      JUMP    $$STEP          STEP IN OR OUT 1 TRACK
170170  104 071 366     $READ*      JUMP    $$READ          READ A SECTOR
170173  104 025 366     $WRITE*     JUMP    $$WRITE         WRITE A SECTOR
170176  104 373 365     $ONLINE*    JUMP    $$ONLINE        SEE IF DRIVE IS ON LINE
170201  104 230 366     $RESTORE*   JUMP    $$RESTORE       SEEK TRACK 0
170204  104 310 366     $DOSDRIV*   JUMP    $$DOSDRIV       CONVERT DOS DRIVE TO MICRO BUS DRIVE
                                        .
170207  104 135 366     $DOIO*      JUMP    $$DOIO          - NOT USED BY DOS -
170212  104 117 366     $BUFIO*     JUMP    $$BUFIO         - NOT USED BY DOS -
170215  104 327 365     $STATUS*    JUMP    $$STATUS        - NOT USED BY DOS -
170220  143             $FDSTAT*    FDSTAT                  - NOT USED BY DOS - (INLINE CAUSE SMALL)
170221  007                         RET
170222  300                         NOP                     (FILL TO 3 BYTE ENTRY)
                                            XIF
                                        *
                                                IFGE    ROMTYPE,2
                                        .
                                        . JUMP VECTOR TO ERROR ROUTINE, NO USEABLE DISKETTE ROUTINES
                                        .
                            $SEEK*      JUMP    ROMVERR         SEEK TO THE TRACK IN (D)
                            $STEP*      JUMP    ROMVERR         STEP IN OR OUT 1 TRACK
                            $READ*      JUMP    ROMVERR         READ A SECTOR
                            $WRITE*     JUMP    ROMVERR         WRITE A SECTOR
                            $ONLINE*    JUMP    ROMVERR         SEE IF DRIVE IS ON LINE
                            $RESTORE*   JUMP    ROMVERR         SEEK TRACK 0
                            $DOSDRIV*   JUMP    ROMVERR         CONVERT DOS DRIVE TO MICRO BUS DRIVE
                                        .
                            $DOIO*      JUMP    ROMVERR         - NOT USED BY DOS -
                            $BUFIO*     JUMP    ROMVERR         - NOT USED BY DOS -
                            $STATUS*    JUMP    ROMVERR         - NOT USED BY DOS -
                            $FDSTAT*    JUMP    ROMVERR         - NOT USED BY DOS -
                                            XIF
```

- AUDIO CHANNEL CONTROL COMMAND SEQUENCE FOR STANDARD SOUNDS -

```
140,F                                   *
141,F 170223                            BEEPCODE
142,F                                   .
143,F                                   . AUDIO CHANNEL CODE FOR A BEEP
144,F                                   .
145,F 170223  260 002                           ACSI    2                       SET AUDIO CHANNEL STATUS
146,F 170225  240 377                           ACLLC   0,255                   BEEP FOR 208 MS
147,F 170227  000 000 000 040 077       BEEPD01 DC      00,00,00,32,63,63,63,32 1200 HZ TONE
148,F 170237  230 012                           ACDSBNZ 0,BEEPD01
149,F                                   .
150,F 170241  240 017                           ACLLC   0,15                    PAUSE FOR 150 MS
151,F 170243  241 144                   BEEPD02 ACLLC   1,100
152,F 170245  231 002                   BEEPD03 ACDSBNZ 1,BEEPD03
153,F 170247  230 006                           ACDSBNZ 0,BEEPD02
154,F 170251  200                               ACHALT                          ZERO STATUS AND HALT
155,F                                   *
156,F                                   .
157,F                                   . AUDIO CHANNEL CODE FOR A CLICK
158,F                                   .
159,F 170252  260 000                   CLIKCODE ACSI   0                       SET AUDIO CHANNEL STATUS
160,F 170254  077 077 077 077 077       CLICKROM DC     63,63,63,63,63          GO HIGH FOR 500 US
161,F 170261  000 000 000 000 000               DC      00,00,00,00,00          GO LOW FOR 500 US
162,F 170266  040                               DC      32                      BACK TO THE MIDDLE
163,F 170267  270                               ACKIL                           HALT WITHOUT CHANGING STATUS (FOR CLICKR)
```

```
166,F
167,F 170270                                *
168,F                                       VECTI
169,F                                       *
170,F                                       * VERSION 2, 2200 COMPATIBLE INTERRUPT VECTOR INITIALIZATION VALUES
171,F 170270  104 331 361                   *
172,F 170273  104 331 361                           JUMP    MEMPAT          MEMORY PARITY ERROR
173,F 170276  104 370 361                           JUMP    MEMPAT
174,F 170301  104 370 361                           JUMP    INPART          INPUT PARITY ERROR
175,F 170304  104 000 362                           JUMP    INPART
176,F 170307  104 000 362                           JUMP    OUTPAT          OUTPUT PARITY ERROR
177,F 170312  104 010 362                           JUMP    OUTPAT
178,F 170315  104 010 362                           JUMP    WVIOLT          WRITE VIOLATION
179,F 170320  104 020 362                           JUMP    WVIOLT
180,F 170323  104 020 362                           JUMP    AVIOLT          ACCESS VIOLATION
181,F 170326  104 030 362                           JUMP    AVIOLT
182,F 170331  104 030 362                           JUMP    IVIOLT          INSTRUCTION VIOLATION
183,F 170334  104 040 360                           JUMP    IVIOLT
184,F 170337  104 040 360                           JUMP    ONEMSA          ONE MILLISECOND
185,F 170342  104 050 362                           JUMP    ONEMSA
186,F 170345  104 050 362                           JUMP    SYSCAT          SYSTEM CALL
187,F 170350  104 045 367                           JUMP    SYSCAT
188,F 170353  104 045 367                           JUMP    BRKPNR          BREAK POINT
189,F 170356  104 040 362                           JUMP    BRKPNR
190,F 170361  104 040 362                           JUMP    UAINST          UNASSIGNED INSTRUCTION
191,F 170364  104 321 361                           JUMP    UAINST
192,F 170367  104 321 361                           JUMP    STPART          SECTOR TABLE PARITY ERROR
193,F                                               JUMP    STPART
194,F                                       *
195,F                                       * ADDITIONAL VECTOR INITIALIZATION VALUES
196,F 170372  007                           *
197,F 170373  000 000                               RET                     SVDISKW$ - DISK WAITS
198,F 170375  007                                   DA      0
199,F 170376  000 000                               RET                     SVDISPW$ - DISPLAY WAITS
200,F                                               DA      0
201,F 170400  252                           *
202,F 170401  210 223 360                           ACLATT  2               SVBEEP - BEEP (0151)
203,F 170404  000 000                               ACJUMP  BEEPCODE
204,F                                               DA      0                  PAD OUT TO 6 BYTES
205,F 170406  252                           *
206,F 170407  210 252 360                           ACLATT  2               SVCLIK - CLICK (0153)
207,F 170412  000 000                               ACJUMP  CLIKCODE
208,F                                               DA      0                  PAD OUT TO 6 BYTES
209,F 000124                                *
                                            VECTIL  EQU     $-VECTI
```

```
212,F                                   *
213,F 170414                            ERRORM
214,F                                   .
215,F                                   . ERROR MESSAGES
216,F                                   .
217,F                                           IFGE    ROMTYPE,2
218,F                                   ROMVERM DC      '*EV ROM VECTOR',$NS,*MERROR
219,F                                           XIF
220,F 170414  052 105 124 040 124       TIMOUTM DC      '*ET TIMEOUT',$NS,*MERROR
221,F 170432  052 105 060 040 123       SYSPRM  DC      '*E0 SYSTEM RAM',$NS,*MERROR
222,F                                   .
223,F 170453  052 105 071 040 123       STPARM  DC      '*E9 SECTOR',$NS,*MPARIT
224,F 170470  052 105 061 040 115       MEMPAM  DC      '*E1 MEMORY',$NS,*MPARIT
225,F 170505  052 105 062 040 111       INPARM  DC      '*E2 INPUT',$NS,*MPARIT
226,F 170521  052 105 063 040 117       OUTPAM  DC      '*E3 OUTPUT'
227,F 170533  040 120 101 122 111       MPARIT  DC      ' PARITY',$NS,*MERROR
228,F                                   .
229,F 170545  052 105 064 040 127       WVIOLM  DC      '*E4 WRITE',$NS,*MPROTE
230,F 170561  052 105 065 040 101       AVIOLM  DC      '*E5 ACCESS'
231,F 170573  040 120 122 117 124       MPROTE  DC      ' PROTECT',$NS,*MERROR
232,F                                   .
233,F 170606  052 105 066 203 225       IVIOLM  DC      '*E6',$NS,*MINSTE
234,F 170614  052 105 067 203 225       SYSCAM  DC      '*E7',$NS,*MINSTE
235,F 170622  052 105 070               UAINTM  DC      '*E8'
236,F 170625  040 111 116 123 124       MINSTE  DC      ' INSTRUCTION'
237,F 170641  040 105 122 122 117       MERROR  DC      ' ERROR* ',$ES    (AT LEAST 1 BLANK AFTER THE *, NO EEOL)
238,F                                   .
239,F 170652  052 105 115 040 115       MERXMSG DC      '*EM MEMORY TEST',$NS,*MERROR   USED BY 1800MOVI IF FATAL
240,F                                   .                                       ERROR HAS OCCURED WHEN SHOULDN'T
241,F                                           IFNE    ERRORM>8,$>8
242,F                                           ERR     ERROR MESSAGE ADDRESSING PROBLEMS
243,F                                           XIF
244,F                                   *
245,F                                   ,ONEMSA
246,F                                   .
247,F                                   . ONE MILLISECOND - RE-ENABLE INTERRUPTS AND JUMP TO LOGICAL LOCATION ZERO
248,F                                   .
249,F                                   .       EJMP    0               (AT 0170037 TO SAVE SPACE)
250,F                                   *
251,F 170674                            SYSPAT
252,F                                   .
253,F                                   . SYSTEM RAM FAILURE - MUST FINISH CLEARING MEMORY AND RE-INITING TABLES
254,F                                   .                      BEFORE CAN DISPLAY ERROR MESSAGE
255,F                                   .
256,F 170674  111 153                           CLICKR                  MAKE NOISE THAT ERROR HAPPENED
257,F 170676  353                               LHD
258,F 170677  364                               LLE                     DE = SVMEMP + VECTIL
259,F 170700  026 254                           LC      256-VECTIL
260,F 170702  250                               XRA
261,F 170703  310                               LBA                     ZERO OUT THE REST OF SYSTEM RAM (167400)
262,F 170704  035                               DECP    HL
263,F 170705  021                               BT                      (ASSUME CAN READ WHAT WRITTEN)
```

```
264,F
265,F 170706  106 117 362                .
                                         CALL    KBDSPINI        INIT KEY TRANS, DISPLAY MEMORY & FONT
266,F
267,F 170711  106 242 372                .
                                         CALL    SSTATE          NOW SWITCH TO DEBUG STACK
268,F 170714  066 032                    LL      SYSPRM          POINT TO THE MESSAGE
269,F 170716  104 055 362                JMP     ERROR           GO DISPLAY IT AND ENTER DEBUG
270,F                                    *
271,F                                    .
272,F                                    . THE REST OF THE TRAPS SIMPLY SAVE THE STATE OF THE MACHINE AND GIVE A MESSAGE
273,F                                    .
274,F 170721  106 242 372        STPART  CALL    SSTATE          SECTOR TABLE PARITY ERROR
275,F 170724  006 053                    LA      STPARM
276,F 170726  104 336 361                JUMP    MSTPAT          THE REST IS LIKE MEMORY PARITY ERROR
277,F                                    *
278,F 170731  106 242 372        MEMPAT  CALL    SSTATE          MEMORY PARITY ERROR
279,F 170734  006 070                    LA      MEMPAM          POINT L-REG TO THE MESSAGE
280,F                                    .
281,F 170736  115 164 207        MSTPAT  DPL     HL,OLDTOS       SWAP THE TOP TWO STACK ENTRIES
282,F 170741  047                        DL      DE,HL           SO THE DISPLAY WILL BE OF THE
283,F 170742  176 044 277                NDL     -1-64           BAD MEMORY LOCATION WHEN DEBUG
284,F 170745  117 015                    INCP    HL,2            IS FINALLY ENTERED
285,F 170747  176 064 100                ORL     64
286,F 170752  111 047                    DL      BC,HL           PICK UP OLD TOS-1
287,F 170754  027                        DS      DE,HL           STORE OLD TOS AT TOS-1
288,F 170755  117 035                    DECP    HL,2
289,F 170757  176 064 100                ORL     64
290,F 170762  111 027                    DS      BC,HL           STORE OLD TOS-1 AT TOS
291,F 170764  360                        LLA                     (NOTE: CODE TO INC & DEC STACK POINTER)
292,F 170765  104 055 362                JUMP    ERROR
293,F                                    *
294,F                                    IFGE    ROMTYPE,2
295,F                            ROMVERR CALL    SSTATE          ATTEMPT MADE TO USE DELETED ROM VECTOR
296,F                                    LL      ROMVERM
297,F                                    JUMP    ERROR
298,F                                    XIF
299,F                                    *
300,F 170770  106 242 372        INPART  CALL    SSTATE          5500 BUS INPUT PARITY ERROR
301,F 170773  066 105                    LL      INPARM
302,F 170775  104 055 362                JUMP    ERROR
303,F                                    *
304,F 171000  106 242 372        OUTPAT  CALL    SSTATE          5500 BUS OUTPUT PARITY ERROR
305,F 171003  066 121                    LL      OUTPAM
306,F 171005  104 055 362                JUMP    ERROR
```

- INTERRUPT VECTOR HANDLERS SET UP AT POR AND RESTART -

```
307,F
308,F 171010  106 242 372  *
309,F 171013  066 145      WVIOLT   CALL     SSTATE          (SECTOR TABLE) WRITE VIOLATION
310,F 171015  104 055 362             LL       WVIOLM
311,F                                 JMP      ERROR
312,F 171020  106 242 372  *
313,F 171023  066 161      AVIOLT   CALL     SSTATE          (SECTOR TABLE) ACCESS VIOLATION
314,F 171025  104 055 362             LL       AVIOLM
315,F                                 JUMP     ERROR
316,F 171030  106 242 372  *
317,F 171033  066 206      IVIOLT   CALL     SSTATE          PRIV'D INSTRUCTION VIOLATION
318,F 171035  104 055 362             LL       IVIOLM
319,F                                 JUMP     ERROR
320,F 171040  106 242 372  *
321,F 171043  066 222      UAINST   CALL     SSTATE          UN-IMPLEMENTED INSTRUCTION VIOLATION
322,F 171045  104 055 362             LL       UAINTM
323,F                                 JUMP     ERROR
324,F 171050  106 242 372  *
325,F 171053  066 214      SYSCAT   CALL     SSTATE          SYSTEM CALL VIOLATION
326,F                                 LL       SYSCAM
327,F                      .          JUMP     ERROR
328,F                      *
329,F                      .TIMOUT  CALL     SSTATE          I-DECODE PROCESSOR TIMEOUT ERROR
330,F                      .        LL       TIMOUTM         (MOVED TO VECTOR POINT 0170044)
331,F                      .        JUMP     ERROR
332,F 171055               *
333,F                      ERROR
334,F                      .
335,F                      . ERROR DISPLAY
336,F 171055  056 361      .
337,F 171057  046 013              LH       MEMPAM>8        DISPLAY ERROR MESSAGE
338,F 171061  036 050              LE       BL                 ON THE BOTTOM LINE
339,F                              LD       40                 STARTING ON COLUMN 40
340,F 171063  313          .       LB       $OW             WRITE BLANKS & DO NOT WAIT ON DISPLAY KEY
341,F 171064  106 341 374          LBD                         (SAVE A BYTE)
342,F 171067  076 327              CALL     DISPLAY         DISPLAY THE MESSAGE
343,F 171071  113 144 207          LX       CURADR>8
344,F 171074  104 120 367          DPL      DE,OLDTOS       CAUSE TOP STACK ENTRY
                                   JMP      BRKPNN          TO BE DISPLAYED
```

```
                                        *
                                        . UNIVERSAL KEYBOARD TRANSLATE TABLE (WILL WORK ON 3600 KEYBOARD AS WELL)
                                        .
                                        .                                       000 TO 017      020 TO 037 IN TABLE
                                        .                                       UNSHIFTED KEYS  SHIFTED KEYS
171077  060 061 062 063 064   UTRANT    DC      '01234567'                      0-7 NUMBER PAD  0-7 NUMBER PAD
171107  010                           DC      BSP                             <BSP>           * UNDEFINED *
171110  070 071                       DC      '89'                            8,9 NUMBER PAD  8,9 NUMBER PAD
171112  030                           DC      CAN                             <CAN> - SHIFTED <CAN> - UNSHIFTED
171113  010                           DC      BSP                             * UNDEFINED *   <BSP>
171114  015                           DC      ENTER                           <ENT>           <ENT>
171115  056                           DC      '.'                             "." NUMBER PAD  "." NUMBER PAD
171116  040                           DC      ' '                             * UNDEFINED *   <0> (SP)
000020                        UTRANTL   EQU     $-UTRANT
                                        .
                                                IFNE    $+1>8,UTRANT>8
                                                ERR     TABLE IS NOT TO CROSS PAGE BOUNDARIES
                                                XIF
                                        .
                                        *
171117                          KBDSPINI
                                        .
                                        . KEYBOARD DISPLAY INITIALIZATION ROUTINE - INITS THE WHOLE KBD/DSP SYSTEM
                                        .
171117  066 150 056 357                 HL      SEKBS2
171123  250                             XRA
171124  340                             LEA                             ZERO OUT THE KEYBOARD CONTROL BYTES
171125  330                             LDA                                (SEKBS2 & SEKBS1)
171126  027                             DS      DE,HL
                                        .                               INIT THE TRANSLATE TABLE
171127  310                             LBA                             NO AUTO INCREMENT OR STOP
171130  026 020                         LC      UTRANTL                 SIZE OF UTRANT
171132  066 077 056 362                 HL      UTRANT                  USE LOWER 040 TRANSLATE TABLE
171136  036 334                         LD      SEKTRAN>8               TO INIT THE KEYBOARD TRANSLATE TABLE
171140  340                             LEA
171141  021                             BT
171142  066 077                         LL      UTRANT                  AND AGAIN (FOR SECOND HALF)
171144  026 020                         LC      UTRANTL
171146  021                             BT
171147  353                             LHD
171150  364                             LLE
171151  376                             LML                             * CODE FOR SPACE (040) IS ONE-TO ONE *
171152  174 015                         INCP    DE                      REST OF TABLE IS ONE TO ONE
171154  006 001                         LA      1                       DO IT THE SIMPLEST WAY
171156  026 137                         LC      128-UTRANTL-UTRANTL-1   FOR THE REMAINDER
171160  021                             BT
                                        .
171161  106 004 374                     CALL    DSPINIT                 INIT THE DISPLAY POINTERS AND CONTROLS
                                        .
171164  250                             XRA                             INIT THE DISPLAY FONT CODES
171165                          KBDSPILP
171165  066 216 056 363                 HL      TRIANGLE                FILL THE DISPLAY FONT MEMORY
```

```
399 F 171171  155                         LODCF                       WITH TRIANGLES FIRST
400 F 171172  004 001                     AD        1
401 F 171174  074 200                     CP        0200
402 F 171176  110 165 362                 JFZ       KBDSPILP
403 F                                 .
404 F 171201  066 272 056 362             HL        SPLCS           THEN LOAD THE CHARACTERS IN THE TABLE
405 F 171205  016 000                     LB        CURSOR             STARTING WITH THE CURSOR CHARACTER
406 F                                 .   JUMP      CHARLOD            AND RETURN FROM THERE (SAVE 3 BYTES)
407 F                                 .
```

```
                                *
                                CHARLOD
                                .
                                . ENTER:  HL => LIST OF CHARACTER FONTS TO BE LOADED
                                .
                                . THE BYTES MATCH SPECIFICATIONS FOR BYTES ON THE 5500 AND THE ROUTINE
                                . AUTOMATICALLY CONVERTS THEM INTO THE 1800 LODCF INSTRUCTION FORMAT.
                                . THIS IS A LIST OF GROUPS OF FIVE BYTES EACH SPECIFYING A COLUMN OF
                                . THE FONT MATRIX.
                                . THE FONT ENTRIES MUST HAVE THE MSBIT CLEAR ON THE FIRST OF THE FIVE
                                . COLUMN FONT BYTES.
                                . THE ASCII CODE THE FIVE BYTES REPRESENT MUST PRECEED THEM AND THE HIGH
                                . ORDER BIT MUST BE SET TO MARK WHICH ASCII CODE IS BEING SPECIFIED.
                                . IF THE FIRST ENTRY IN THE LIST DOES NOT HAVE AN ASCII CODE, THE VALUE
                                . IN THE B-REG IS USED.
                                . IF THE ASCII CHARACTER IS THE NEXT ONE IN SEQUENCE, IT MAY BE LEFT OUT.
                                . THE LIST IS TERMINATED BY A 0200.
                                . THEREFORE, THE ONLY WAY TO LOAD CODE 000 IS IF IT THE FIRST ENTRY IN THE LIST
                                . AND THE B-REG IS ZERO.
                                .
                                .   EXIT: ALL REGISTERS USED EXCEPT X-REG
                                .         ONE STACK POSITION IN ADDITION TO THE CALL USED
                                .         ZERO CONDITION TRUE
                                .
171207  307                     CHRLOP  LAM                     GET THE LIST ENTRY
171210  054 200                         XR      0200            CLEAR HIGH BIT IF SET (OR SET IT)
171212  053                             RTZ                     STOP IF TERMINATOR (0200) REACHED
                                .
171213  046 005                         LE      5               SET THE COLUMN COUNT
171215  160 223 362                     JTS     CHRGO           START CONVERSION IF FONT BIT NOW SET
                                .
171220  310                             LBA                     ELSE SAVE ASCII CHAR IN THE B-REG
171221  015                             INCP    HL              POINT HL TO THE NEXT COLUMN BYTE
                                .
171222  307                     CHRCOL  LAM                     GET THE COLUMN FONT BYTE
171223  070                     CHRGO   PUSH    HL              SAVE THE TABLE POINTER
171224  066 244 056 335                 HL      ROWS            POINT TO TEMP GENERATION AREA
                                .
171230  012                     CHRROW  SRC                     SET CARRY FROM LSB
171231  327                             LCM
171232  062 212                         ACCC                    SHIFT CARRY BIT IN
171234  372                             LMC
171235  015                             INCP    HL              POINT TO THE NEXT ROW
171236  176 074 253                     CPL     ROWS+7
171241  110 230 362                     JFZ     CHRROW          CONTINUE FOR 7 ROWS
                                .
171244  060                             POP     HL              HL => NEXT COLUMN FONT BYTE
171245  015                             INCP    HL
171246  174 024 001                     SUE     1
171251  110 222 362                     JFZ     CHRCOL          LOOP UNTIL 5 COLUMNS DONE
                                .
171254  070                             PUSH    HL              SAVE THE TABLE POINTER
```

```
462,F 171255  066 244 056 335                 HL        ROWS              LOAD THE 1800 CHARACTER
463,F 171261  301                             LAB
464,F 171262  155                             LODCF
465,F 171263  060                             POP       HL                RESTORE THE TABLE POINTER
466,F 171264  111 004 001                     ADB       1                 BUMP B-REG TO NEXT ASCII CHAR
467,F 171267  104 207 362                     JMP       CHRLOP            GO THROUGH THE LOOP AGAIN
468,F                                         *
469,F                                         *
470,F 171272                                  SPLCS
471,F                                         *
472,F                                         * SPECIAL CHARACTER SET FOR POWER UP INITIALIZATION OF THE RAM DISPLAY
473,F                                         *
474,F 171272  177 177 177 177 177             DC        0177,0177,0177,0177,0177   000 (CURSOR CHARACTER)
475,F 171277  240                             DC        ' '+0200
476,F 171300  000 000 000 000 000             DC        0000,0000,0000,0000,0000   040 (SPACE - BLANK)
477,F 171305  252                             DC        '*'+0200
478,F 171306  024 010 076 010 024             DC        0024,0010,0076,0010,0024   *
479,F 171313  255                             DC        '-'+0200
480,F 171314  010 010 010 010 010             DC        0010,0010,0010,0010,0010   -
481,F 171321  272                             DC        ':'+0200
482,F 171322  000 021 000 000 000             DC        0000,0021,0000,0000,0000   :
483,F 171327  260                             DC        '0'+0200
484,F 171330  076 105 111 121 076             DC        0076,0105,0111,0121,0076   0
485,F 171335  021 041 177 001 001             DC        0021,0041,0177,0001,0001   1
486,F 171342  043 105 105 111 061             DC        0043,0105,0105,0111,0061   2
487,F 171347  102 101 111 125 042             DC        0102,0101,0111,0125,0042   3
488,F 171354  014 024 044 177 004             DC        0014,0024,0044,0177,0004   4
489,F 171361  162 121 121 121 116             DC        0162,0121,0121,0121,0116   5
490,F 171366  036 051 111 111 106             DC        0036,0051,0111,0111,0106   6
491,F 171373  100 107 110 120 140             DC        0100,0107,0110,0120,0140   7
492,F 171400  066 111 111 111 066             DC        0066,0111,0111,0111,0066   8
493,F 171405  061 111 111 112 074             DC        0061,0111,0111,0112,0074   9
494,F 171412  301                             DC        'A'+0200
495,F 171413  077 110 110 110 077             DC        0077,0110,0110,0110,0077   A
496,F 171420  177 111 111 111 066             DC        0177,0111,0111,0111,0066   B
497,F 171425  076 101 101 101 042             DC        0076,0101,0101,0101,0042   C
498,F 171432  177 101 101 042 034             DC        0177,0101,0101,0042,0034   D
499,F 171437  177 111 111 111 101             DC        0177,0111,0111,0111,0101   E
500,F 171444  177 110 110 110 100             DC        0177,0110,0110,0110,0100   F
501,F 171451  076 101 101 111 056             DC        0076,0101,0101,0111,0056   G
502,F 171456  177 010 010 010 177             DC        0177,0010,0010,0010,0177   H
503,F 171463  000 101 177 101 000             DC        0000,0101,0177,0101,0000   I
504,F 171470  006 001 101 176 100             DC        0006,0001,0101,0176,0100   J
505,F 171475  177 010 024 042 101             DC        0177,0010,0024,0042,0101   K
506,F 171502  177 001 001 001 001             DC        0177,0001,0001,0001,0001   L
507,F 171507  177 040 030 040 177             DC        0177,0040,0030,0040,0177   M
508,F 171514  177 060 010 006 177             DC        0177,0060,0010,0006,0177   N
509,F 171521  076 101 101 101 076             DC        0076,0101,0101,0101,0076   O
510,F 171526  177 110 110 110 060             DC        0177,0110,0110,0110,0060   P
511,F 171533  076 101 105 102 075             DC        0076,0101,0105,0102,0075   Q
512,F 171540  177 110 114 112 061             DC        0177,0110,0114,0112,0061   R
513,F 171545  061 111 111 111 106             DC        0061,0111,0111,0111,0106   S
```

```
514,F 171552  100 100 177 100 100            DC      0100,0100,0177,0100,0100   T
515,F 171557  176 001 001 001 176            DC      0176,0001,0001,0001,0176   U
516,F 171564  160 014 003 014 160            DC      0160,0014,0003,0014,0160   V
517,F 171571  177 002 014 002 177            DC      0177,0002,0014,0002,0177   W
518,F 171576  143 024 010 024 143            DC      0143,0024,0010,0024,0143   X
519,F 171603  160 010 017 010 160            DC      0160,0010,0017,0010,0160   Y
520,F 171610  103 105 111 121 141            DC      0103,0105,0111,0121,0141   Z
521,F 171615  200                            DC      0200                       -END-OF-LIST-
522,F
                                             .
523,F 171616  037 017 007 003 001   TRIANGLE  DC      037,017,007,003,001,000,000  TRIANGLE IN 1800 FORMAT
524,F                                        .
```

- SECTOR TABLE INITIALIZATION VALUES -

```
527,F                                   *
528,F                                   * SECTOR TABLE INITIALIZATION VALUES
529,F                                   *
530,F                                   *                       ENCODED STL VALUE     ADDR    CARD#    CARD#
531,F 171625  014                       SMST    DC      $-SMST<4+STAE+STWE    150K - 12K 1 - 32K 1
532,F 171626  034                               DC      $-SMST<4+STAE+STWE    160K - 12K 1 - 32K 1
533,F 000002                            SMSTL   EQU     $-SMST
534,F                                   *
535,F                                   *                       ENCODED STL VALUE     ADDR    CARD#    CARD#
536,F 171627  054                       UMST    DC      $-SMST<4+STAE+STWE    000K - 12K 1 - 32K 1
537,F 171630  074                               DC      $-SMST<4+STAE+STWE    010K - 12K 2 - 32K 1
538,F 171631  114                               DC      $-SMST<4+STAE+STWE    020K - 12K 2 - 32K 1
539,F 171632  134                               DC      $-SMST<4+STAE+STWE    030K - 12K 2 - 32K 1
540,F 171633  154                               DC      $-SMST<4+STAE+STWE    040K - 12K 3 - 32K 1
541,F 171634  174                               DC      $-SMST<4+STAE+STWE    050K - 12K 3 - 32K 1
542,F 171635  214                               DC      $-SMST<4+STAE+STWE    060K - 12K 3 - 32K 2
543,F 171636  234                               DC      $-SMST<4+STAE+STWE    070K - 12K 4 - 32K 2
544,F 171637  254                               DC      $-SMST<4+STAE+STWE    100K - 12K 4 - 32K 2
545,F 171640  274                               DC      $-SMST<4+STAE+STWE    110K - 12K 4 - 32K 2
546,F 171641  314                               DC      $-SMST<4+STAE+STWE    120K - 12K 5 - 32K 2
547,F 171642  334                               DC      $-SMST<4+STAE+STWE    130K - 12K 5 - 32K 2
548,F 171643  354                               DC      $-SMST<4+STAE+STWE    140K - 12K 5 - 32K 2
549,F 000015                            UMSTL   EQU     $-UMST
550,F                                   *
551,F                                   *                       ENCODED STL VALUE                     CARD#
552,F 171644  015                       HMST    DC      $-HMST<4+STA16+STAE+STWE            - 32K 3
553,F 171645  035                               DC      $-HMST<4+STA16+STAE+STWE            - 32K 3
554,F 171646  055                               DC      $-HMST<4+STA16+STAE+STWE            - 32K 3
555,F 171647  075                               DC      $-HMST<4+STA16+STAE+STWE            - 32K 3
556,F 171650  115                               DC      $-HMST<4+STA16+STAE+STWE            - 32K 3
557,F 171651  135                               DC      $-HMST<4+STA16+STAE+STWE            - 32K 3
558,F 171652  155                               DC      $-HMST<4+STA16+STAE+STWE            - 32K 3
559,F 171653  175                               DC      $-HMST<4+STA16+STAE+STWE            - 32K 3
560,F 171654  215                               DC      $-HMST<4+STA16+STAE+STWE            - 32K 4
561,F 171655  235                               DC      $-HMST<4+STA16+STAE+STWE            - 32K 4
562,F 171656  255                               DC      $-HMST<4+STA16+STAE+STWE            - 32K 4
563,F 171657  275                               DC      $-HMST<4+STA16+STAE+STWE            - 32K 4
564,F 171660  315                               DC      $-HMST<4+STA16+STAE+STWE            - 32K 4
565,F 171661  335                               DC      $-HMST<4+STA16+STAE+STWE            - 32K 4
566,F 171662  355                               DC      $-HMST<4+STA16+STAE+STWE            - 32K 4
567,F 171663  375                               DC      $-HMST<4+STA16+STAE+STWE            - 32K 4
568,F                                   *
569,F 000037                            PISTL   EQU     $-SMST          POWER INITIALIZE SECTOR TABLE LENGTH
570,F                                   *
571,F                                           IFNE    $>8,SMST>8
572,F                                           ERR     SECTOR TABLE ADDRESSING PROBLEMS - PAGE BOUNDARY
573,F                                           XIF
574,F                                   *
575,F 171664  054                       WRAPST  DC      2<4+STAE+STWE   SPECIAL MEMORY SIZE TEST SECTOR TABLE
576,F 171665  055                               DC      2<4+STA16+STAE+STWE
```

```
                                - POWER UP AND RESTART -

579,F
580,F 171666                    *
581,F                           POWERUP
582,F                           *
583,F                           * ZERO ALL OF MEMORY, INITIALIZE THE SECTOR TABLE, ZERO THE BASE REGISTER
584,F                           *       AND LOAD THE RAM DISPLAY WITH NEEDED CHARACTERS.
585,F 171666  040               *
586,F 171667  066 263 056 363           DI                      KILL INTERRUPTS
587,F 171673                            HL      SMST+PISTL-1    START FROM THE LAST (124K) ENTRY
588,F 171673  026 001           POWERL1
589,F 171675  077                       LC      1               LOAD INTO THE ZEROTH SECTOR TABLE
590,F                                   STL                        ENTRY, THE VALUE POINTED TO BY HL
591,F 171676  036 000           *
592,F 171700  343                       LD      0               ZERO THAT SECTOR
593,F 171701  026 376 016 017           LED
594,F 171705                            BC      4096-2          4K IS SECTOR LENGTH
595,F 171705  174 027           POWERL2
596,F 171707  113 035                   DS      DE,BC           NOTE:  ALL MEMORY MUST BE ZEROED
597,F 171711  100 305 363               DECP    BC,2            BEFORE A STLOAD MOVES THE SYSTEM
598,F                                   JFC     POWERL2         EMULATION SUPPORT AREA TO MEMORY
599,F                           *                               WITH BAD PARITY ELSE SYSTEM DIES
600,F 171714  035               *
601,F 171715  176 074 225               DECP    HL              BACK TO THE NEXT SECTOR TABLE ENTRY
602,F 171720  100 273 363               CPL     SMST            CATCH OFF END OF TABLE
603,F                                   JFC     POWERL1
604,F 171723  066 225           *
605,F 171725  026 322                   LL      SMST            LOAD THE SYSTEM RAM SECTOR TABLE
606,F 171727  062 077                   LC      (SEDSPBF>8,AND,0360)+SMSTL      ENTRIES
607,F                                   STLOC
608,F 171731  066 300 056 357   *
609,F 171735  176 065                   HL      SESTACK         SET UP THE STACK POSITION
610,F 171737  106 045 373               SYSMOV  HL
611,F                                   CALL    SETUP           NOW SET UP EVERYTHING ELSE
612,F 171742  106 117 362       *
613,F                                   CALL    KBDSPINI        INIT KEYBOARD & DISPLAY INFORMATION
614,F 171745  250               *
615,F 171746  111 010                   XRA                     GO TO DEBUG IF NOT 1800 PROCESSOR
616,F 171750  074 002                   INFO
617,F 171752  110 127 367               CP      PN1800          NO MESSAGE? CALL SSTATE & CALL IDENT MAYBE
618,F                                   JFZ     DEBUG
619,F                           *
620,F 171755  046 000                   IFLE    ROMTYPE,1
621,F                                   LE      0               TURN OFF ALL THE DRIVE LIGHTS
622,F 171757  036 001           *
623,F 171761  106 264 366       POWERDCL LD     FODR0
624,F 171764  036 002                   CALL    TOFFTLDE        TURN OFF THE LIGHT IN DRIVE 0
625,F 171766  106 264 366               LD      FODR1
626,F 171771  174 004 001               CALL    TOFFTLDE        TURN OFF THE LIGHT IN DRIVE 1
627,F 171774  174 044 003               ADE     1               INCREMENT THE DRIVE NUMBER
628,F 171777  110 357 363               NDE     $DRVTBLN>1      CATCH ALL DRIVES DONE
629,F                                   JFZ     POWERDCL
630,F                                   XIF
                                *                               FALLS THROUGH INTO THE RESTART CODE
```

- POWER UP AND RESTART -

```
631,F                                   *
632,F 172002                            RESTART
633,F                                   .
634,F                                   . RESTART - INVOKED BY THE TRANSITION FROM BOTH 'RESTART' AND 'RUN' KEYS
635,F                                   .           DOWN TO NOT BOTH DOWN.  THE DIAGNOSTIC ROUTINE IS ENTERED
636,F                                   .           IF THE 'DISPLAY' KEY IS DOWN DURING THE TRANSITION.
637,F                                   .
638,F
639,F 172002  040                              DI                              DISABLE INTERRUPTS
640,F 172003  106 242 372                      CALL    SSTATE                  SAVE THE MACHINE STATE
641,F 172006  111 153                          CLICKR                          FORCE AUDIO TO CLEAR THE CHANNEL
642,F                                   .
643,F 172010  076 357                          LX      SEKBS2>8                SET UP THE PAGE REGISTER
644,F 172012  105 151                          PL      A,SEKBS1                JUMP TO DEBUG IF THE "DISPLAY" KEY
645,F 172014  044 003                          ND      SEDSPKY+SEKBDKY             IS DOWN AND NOT "KEYBOARD" KEY
646,F 172016  074 001                          CP      SEDSPKY
647,F 172020  150 133 367                      JTZ     DEBUGI
648,F                                   .
649,F 172023  066 225 056 363                  HL      SMST                    INSURE THAT THE STACK WILL NOT MOVE
650,F 172027  026 322                          LC      (SEDSPBF>8.AND.0360)+SMSTL
651,F 172031  062 077                          STLOC                               WHEN RE-SECTORING  HAPPENS IN MTSETUP
652,F 172033  015                       RESWAIT INCP    HL                      * DELAY HERE TO ALLOW KBD STATUS TO SETTLE
653,F 172034  100 033 364                      JFC     RESWAIT                     (USES HL FOR APPROX. 40 MILLISEC DELAY)
654,F                                   .
655,F 172037  106 045 373                      CALL    SETUP                   INIT SECTORING AND BASING
656,F                                          IFLE    ROMTYPE,1
657,F 172042  250                              XRA
658,F 172043  165                              SISTART                         STOP ANY COMMING IN PROGRESS
659,F 172044  062 161                          SIMODOUT                        STOP ANY TALKING IN PROGRESS
660,F 172046  113 161                          SIACUDUT                        STOP ANY DIALING IN PROGRESS
661,F                                          XIF
662,F                                   .
663,F 172050  105 150                          PL      A,SEKBS2                CLEAR THE SCREEN IF THE "F1" KEY
664,F 172052  044 001                          ND      SEFUNC1                     IS UP, AND
665,F 172054  152 117 362                      CTZ     KBDSPINI                    INIT FONTS AND KEYBOARD TRANSLATE TABLE
666,F                                          IFLE    ROMTYPE,1
667,F 172057  106 020 367                      CALL    RESTDSK                 INIT DISK VARS TO ALL ONES
668,F                                          XIF
669,F 172062  111 030                          ALPHAL                          INSURE THAT DOS(WHATEVER) ENTERED IN ALPHA
670,F                                   .      JMP     RESTIPL                 THE REST IS IN "1800BOOT"
671,F                                   .                                      OR ITS VARIATIONS (SAVE 3 BYTES)
142,                                    .
143,                                    . .. WARNING:  THE FIRST BYTE OF THE NEXT MODULE MUST BE LABELED "RESTIPL"
144,                                    .              AS THERE IS A FALL-THROUGH INTO IT TO DO THE RESTART BOOT
145,                                    .              FROM THE PREVIOUS MODULE.
146,                                    .
147,                                           IFNE    ROMTYPE,1
148.                                           INC     1800BOOT
```

```
                                    .
                                    . 1800 DISKETTE IPL RESTART ROUTINE      MAR 20, 1978      16:00   HSP/HJS
                                    *
172064                              RESTIPL
                                    .
                                    . RESTART IPL BLOCK LOAD ROUTINE
                                    .
                                            IFEQ    ROMTYPE,3
                                    . IF THE KEYBOARD KEY IS DOWN AND THERE IS A RIM ON THE I/O BUS
                                    .       THEN LOAD THE RIM-DLL CODE INTO LOW MEMORY AND GO TO IT, ELSE
                                            XIF
                                    . SCAN THROUGH THE DRIVES WAITING FOR ONE ON WITH A DISK ON LINE,
                                    .       CONTROLLER READY, AND THE DISK NOT BUSY.  IF NOTHING
                                    .       WILL LOAD FROM THE CURRENT DRIVE, CHECK THE KEYBOARD STATUS
                                    .       FOR THE DISPLAY KEY BEING DEPRESSED.
                                    .       IF NOT DOWN, RESTART FROM THE TOP ALL OVER AGAIN.
                                    .       IF DOWN, SCAN TO THE NEXT DRIVE IN SEQUENCE UNTIL ALL DRIVES SCANNED.
                                    .       IF ALL DRIVES SCANNED, RESTART FROM THE TOP ALL OVER AGAIN.
                                    .
                                            IFEQ    ROMTYPE,3
                                            CALL    DOSF60N
                                            JFS     RESTNRIM        KEYBOARD KEY DOWN?
                                            JFP     RESTNRIM        DISPLAY KEY DOWN ALSO?
                                    .
                                            MLA     *RIMDLLS+RIMADRL CHECK THE RIM'S TO SEE IF ANY THERE
                                    RIMLOOP EX      ADR
                                            IN
                                            ORA
                                            JFZ     RIMBOOT         THERE IS AT LEAST ONE AT ITS ADDRESS
                                    .
                                            INCP    HL
                                            LAM
                                            ORA
                                            JFZ     RIMLOOP         CONTINUE TILL ALL RIMS TRIED
                                    .
                                    RIMWAIT CALL    DOSF60N         WAIT FOR BOTH KEYS COMING UP
                                            JTS     RIMWAIT
                                            JTP     RIMWAIT
                                            XIF
                                    .
                                            IFNE    ROMTYPE,2
172064  250                         RESTNRIM XRA
172065  076 335                             LX      $DATA$>8        SET UP THE PAGE REGISTER
                                    .
172067                              RESTLOOP
172067  106 115 364                         CALL    LOADIPL         TRY TO LOAD THE IPL BLOCK ON DRIVE (A)
                                            IFEQ    ROMTYPE,3
                                            CALL    BOOTCYN         TRY TO BOOT THE CYNTHIA DISK
                                            XIF
                                            IFNE    ROMTYPE,2
172072  106 102 373                         CALL    DOSF60N           (X & B REGS NOT DESTROYED)
172075  130 064 364                         JFP     RESTIPL         'DSP KEY' NOT DOWN, TRY DRIVE ZERO
```

```
55,G
56,G 172100  105 217                .
57,G 172102  004 001                        PL      A,$LOGDRV
58,G 172104  074 010                        AD      1               TRY THE NEXT LOGICAL DRIVE
59,G 172106  140 067 364                    CP      $DRVTBLN+1      IF NOT ALL TRIED
60,G                                        JTC     RESTLOOP
61,G 172111  153                    .
62,G 172112  104 064 364                    EX      CLICK
                                            JMP     RESTIPL         IF ALL TRIED, THEN START ALL OVER AGAIN
```

```
 63,G
 64,G 172115                                +
 65,G                                       LOADIPL
 66,G                                       .
 67,G                                       . LOAD THE DISK IPL BLOCK (TRACK 0 PHYSICAL SECTOR 1) FROM DRIVE (A-REG)
 68,G                                       .       AND EXECUTE IF GOOD FORMAT (LSB,MSB,-LSB,-MSB,0)
 69,G 172115  106 310 366                   .
 70,G 172120  111 143                       .       CALL    $$DOSDRIV       SELECT THE LOGICAL DRIVE GIVEN IN A-REG
 71,G 172122  106 373 365                           FDSCLR                  MASTER CLEAR ANY DRIVE IN USE ***
 72,G 172125  043                           .       CALL    $$ONLINE        EXIT IF DRIVE NOT ON LINE
 73,G                                               RTC                        AND READY TO GO
 74,G 172126  106 230 366                   .       CALL    $$RESTORE       TRY TO RESTORE THAT DRIVE
 75,G 172131  043                           .       RTC                     EXIT IF NOT ON LINE
 76,G                                       .
 77,G 172132  036 000                       .       LD      0               IPL BLOCK ON TRACK 0
 78,G 172134  046 001                               LE      1                                  SECTOR 1
 79,G 172136  363                           .       LLD                     HL => PAGE WHERE IPL IS TO BE READ
 80,G 172137  056 330                               LH      $DISKBUF>8
 81,G 172141  106 071 366                   .       CALL    $$READ          READ THE IPL BLOCK
 82,G 172144  153                           .       EX      CLICK           CLICK EACH ATTEMPTED READ
 83,G 172145  043                           .       RTC                     EXIT IF BAD READ
 84,G                                       .
 85,G 172146  047                           .       DL      DE,HL           ELSE GET THE STARTING ADDRESS
 86,G 172147  117 015                               INCP    HL,2
 87,G 172151  111 047                       .       DL      BC,HL           AND THE 1'S COMPLEMENT
 88,G 172153  117 015                               INCP    HL,2
 89,G 172155  062 254                       .       XREC                    SEE IF BC IS THE 1'S
 90,G 172157  111 253                               XRDB                       COMPLEMENT OF DE
 91,G 172161  062 015                       .       INCP    BC              IF BC IS -1 (SHOULD BE) WILL SET CARRY
 92,G 172163  003                           .       RFC                     EXIT IF BAD IPL FORMAT
 93,G                                       .
 94,G 172164  307                           .       LAM                     THE 5TH BYTE MUST BE ZERO
 95,G 172165  260                                   ORA
 96,G 172166  013                           .       RFZ                     EXIT IF BAD IPL FORMAT
 97,G                                       .
 98,G 172167  105 217                       .       PL      A,$LOGDRV       ELSE STORE DRIVE NUMBER THERE
 99,G 172171  370                                   LMA
100,G 172172  174 070                       .       PUSH    DE              SAVE THE DESTINATION ADDRESS
101,G 172174  026 373                       .       LC      251             MOVE THE NEXT 251 BYTES
102,G 172176  250                                   XRA                        FROM THE DISK BUFFER TO
103,G 172177  310                                   LBA                        THE STARTING ADDRESS
104,G 172200  021                                   BT
105,G 172201  060                           .       POP     HL              START EXECUTING THE IPL
106,G 172202  015                                   INCP    HL                 AFTER THE DRIVE NUMBER
107,G 172203  070                           .       PUSH    HL
108,G 172204  007                           .       RET
109,G                                      .
110,G                                      .       IFEQ    ROMTYPE,3
111,G                                      .       INC     1800BCYN
112,G                                      .       XIF
```

```
113,G                   +
114,G                           IFGE    ROMTYPE,2
115,G                   RIMBOOT DI
116,G                           XRA
117,G                           LBA
118,G                           LC      RIMDLLNL        THE DLL CODE IS MOVED TO BOTTOM MEMORY
119,G                           LEA
120,G                           LDA
121,G                           HL      RIMDLLS         FROM ITS COPY IN ROM
122,G                           BT
123,G                           JMP     000000          AND THEN JUMPED TO, NEVER TO RETURN
124,G                   .
125,G                   RIMDLLS EQU     $               RIM DOWN LINE LOAD ROUTINE
126,G                           LOC     0
127,G                           INC     RIMDLL
128,G                           LOC     *
129,G                           XIF
149,                            XIF
150,                    .
151,                            IFEQ    ROMTYPE,1
152,                            INC     1800BAPF        (ABOVE WARNING HOLDS FOR THIS MODULE TOO!)
153,                            XIF
154,
155,                    .
156,                            IFEQ    ROMTYPE,0
                                INC     1800DIAG
```

```
                                .
                                . 1800 FLEXIBLE DISKETTE DIAGNOSTIC          JUNE 8, 1978    10:30    HSP/HJS
                                .
172205                          FDIAG
                                .
                                . ENTERED FROM DEBUG WITH THE "12345H" COMMAND
                                .
                                . PERFORMS THE FOLLOWING TESTS:
                                .
                                .   0. VERIFY - MEDIA VERIFY (SEQUENTIAL VERIFICATION OF ALL SECTORS)
                                .              UNTIL A SECTOR CAN NOT BE READ.
                                .
                                . 1&2. SEEK12 - ALTERNATE CYLINDER SEEK FROM 1 TO 0 OR 2 TO 1 WITH READ-VERIFY.
                                .
                                .   3. FIND   - ATTEMPTS TO FIND TRACK 38 ON A DOUBLE DENSITY DISKETTE.
                                .              USED TO CO-RELATE ACCURACY OF TRACK 0 FLAG.
                                .              CONTINUOUSLY READS ANY SECTOR.
                                .
                                .   4. ALIGN  - POSITION TO TRACK 38 AND HOLD THE HEAD DOWN
                                .              UNTIL THE KEYBOARD KEY IS DEPRESSED. SINGLE (ANY) DENSITY
                                .              READS WILL BE ATTEMPTED BUT THE DATA WILL NOT BE DISPLAYED.
                                .
                                .      NOTE: - COMMANDS 0, 1, 2, AND 3
                                .              WILL WAIT AFTER EACH OPERATION
                                .              IF THE DISPLAY KEY IS DEPRESSED.
                                .              THEY WILL EXECUTE CONTINUOUSLY
                                .              UNTIL THE KEYBOARD KEY IS DEPRESSED OR
                                .              UNTIL THE SELECTED DRIVE GOES OFF LINE.
                                .
172205  106 112 370                     CALL    C12345          ENTRY MUST BE '12345H'
                                .
172210  106 004 374                     CALL    DSPINIT         INIT DISPLAY POINTERS
                                .
172213  106 020 367             FDIAGDR CALL    RESTDSK         INIT ALL DRIVE CONTROLS TO ALL ONES
172216  106 157 365                     CALL    FGETRSP         GET THE DRIVE NUMBER
172221  224 213 377 207 022             DC      $HD,$VA,-1,$CK,$EEOL,'DRIVE: ',$ES
                                .
172236  043                             RTC                     MANUAL EXIT FROM DISK DIAGNOSTIC
172237  074 010                         CP      8               MUST BE 0-7
172241  100 213 364                     JFC     FDIAGDR
                                .
172244  106 310 366                     CALL    $$DOSDRIV       SELECT THE DRIVE
172247  106 230 366                     CALL    $$RESTORE          AND RESTORE IT.
172252  140 213 364                     JTC     FDIAGDR         CATCH DRIVE NOT ON LINE
                                .
172255  106 157 365                     CALL    FGETRSP         GET THE TEST NUMBER
172260  224 124 105 123 124             DC      $HD,'TEST 0:VFY, 1:T1-0, 2:T2-1, 3:FND, 4:ALN ',$ES
                                .
172333  043                             RTC                     <0 MANUAL EXIT FROM DISK DIAGNOSTIC
172334  150 031 365                     JTZ     FAUTO           0 => MEDIA VERIFY
                                .
172337  074 003                         CP      3               1 => SEEK BETWEEN TRACKS 1 AND 0
```

```
55,H 172341  140 013 365                JTC     FSEEK           2 => SEEK BETWEEN TRACKS 2 AND 1
56,H 172344  150 001 365                JTZ     FFIND           3 => FIND TRACK 38 & DD READ IT
57,H                                    .
58,H 172347  074 004                    CP      4
59,H 172351  110 213 364                JFZ     FDIAGDR         >4 TRY ANOTHER NUMBER
60,H                                    .       JTZ     FALIGN          4 => SEEK TO TRACK 38
61,H                                    *
62,H 172354                             FALIGN
63,H                                    .
64,H                                    . SEEK TO TRACK 38 AND CONTINUOUSLY READ IT
65,H                                    .
66,H 172354  036 046                    LD      38              STEP TO CORRECT ALIGNMENT TRACK
67,H 172356  046 000                    LE      0               (ASSUMING TRACK ZERO FLAG CORRECT)
68,H 172360  106 201 365                CALL    $$SEEK
69,H                                    .
70,H 172363  016 000            FALOOP  LB      FFREAD          READ (ACCEPT ANY DATA)
71,H 172365  076 335                    LX      $DATA$>8            (FROM THE CORRECT PAGE IN MEMORY)
72,H 172367  106 135 366                CALL    $$DOIO
73,H 172372  106 102 373                CALL    DOSF60N         TEST THE KEYBOARD KEY
74,H 172375  120 363 364                JFS     FALOOP          TO KNOW WHEN TO END
75,H                                    .
76,H 172400  007                        RET
77,H                                    *
78,H 172401                             FFIND
79,H                                    .
80,H                                    . FIND TRACK 38 ON A DOUBLE DENSITY DISKETTE
81,H                                    .
82,H 172401  036 046                    LD      38
83,H 172403  046 000                    LE      0               ANY SECTOR WILL DO
84,H 172405  106 125 365                CALL    FCHECKR         CHECK AND READ THE SECTOR
85,H 172410  104 001 365                JMP     FFIND           CONTINUE TILL STOPPED
86,H                                    *
87,H 172413                             FSEEK
88,H                                    .
89,H                                    . ALTERNATE CYLINDER SEEK FROM 2 TO 1 OR FROM 1 TO 0 WITH READ VERIFY
90,H                                    .
91,H 172413  340                        LEA                     SAVE 2 OR 1 AS MARKER OF SEEK TYPE
92,H 172414  334                FSEEK1  LDE                     TRACK 2 OR 1 SECTOR 2 OR 1
93,H 172415  106 125 365                CALL    FCHECKR         CHECK AND READ A SECTOR
94,H 172420  113 024 001                SUD     1               TRACK 1 OR 0 SECTOR 2 OR 1
95,H 172423  106 125 365                CALL    FCHECKR         READ THE OTHER TRACK/SECTOR
96,H 172426  104 014 365                JUMP    FSEEK1
```

```
 97,H
 98,H 172431                        *
 99,H                               FAUTO
100,H                               .
101,H                               . MEDIA VERIFY - READ ALL SECTORS ON ALL TRACKS UNTIL FAULT
102,H 172431  036 000               .         LD      0               START AT TRACK 0
103,H                               .
104,H 172433  006 002               FAUT00    LA      2               START AT SECTOR 2
105,H                               .                                 BY STARTING AT 2 (AND ENDING AT SECTOR 25)
106,H                               .                                 26-1 TRANSITION IN MIDDLE OF TRACK READ
107,H                               .                                 25-2 TRANSITION ACROSS TRACKS READ
108,H                               .                                 THIS GIVES MAX TIME FOR STEP TO COMPLETE
109,H                               .
110,H 172435  340                   FAUT01    LEA
111,H 172436  106 125 365                     CALL    FCHECKR         CHECK AND READ A SECTOR
112,H 172441  140 076 365                     JTC     FAUTOER
113,H                               .
114,H 172444  304                             LAE                     INCREMENT THE SECTOR NUMBER
115,H 172445  004 002                         AD      2
116,H 172447  074 033                         CP      MAXSECT+1       SEE IF .LT. 27
117,H 172451  140 035 365                     JTC     FAUT01          DO NEXT SECTOR IF SO
118,H                               .                                 HERE A = 28 IF HALF WAY & 27 WHEN DONE
119,H 172454  024 033                         SU      MAXSECT+1       BACK TO BEGINNING OF TRACK (IF BACK TO 1)
120,H 172456  110 035 365                     JFZ     FAUT01
121,H                               .
122,H 172461  153                             EX      CLICK           CLICK ONCE EACH TRACK (WENT BACK TO ZERO)
123,H 172462  113 004 001                     ADD     1               INCREMENT THE TRACK NUMBER
124,H 172465  113 074 115                     CPD     MAXTRAK+1       DO NEXT TRACK IF NOT UP TO MAXIMUM
125,H 172470  140 033 365                     JTC     FAUT00
126,H
127,H 172473  104 031 365           .         JMP     FAUTO           ELSE DO THE DISK ALL OVER AGAIN
128,H                               *
129,H 172476  066 172 056 365       FAUTOER   HL      FAUTOMSG
130,H 172502  046 013 036 062               DE      LC+50<8+8L
131,H 172506  106 337 374                   CALL    DISPDOS
132,H 172511  070                           PUSH                    (CORRECT FOR POP)
133,H                               *
134,H 172512  060                   FDIAGEX   POP                     FORGET NORMAL RETURN (FROM FCHECKR)
135,H 172513  076 327                       LX      CURADR>8
136,H 172515  046 221 036 335               DE      $TRACK
137,H 172521  113 146 200                   DPS     DE,CURADR
138,H 172524  007                           RET                     RETURN TO DSPCAD (RATHER THAN JUMP)
```

```
139,H
140,H 172525
141,H
142,H
143,H
144,H 172525  106 102 373          *
145,H 172530  170 125 365          FCHECKR
146,H 172533  160 112 365          .
147,H                              . READ A SECTOR (IF ABLE TO) ROUTINE
148,H 172536  106 373 365          .
149,H 172541  140 112 365                CALL    DOSF60N         TEST KBD/DSP KEYS (NO CURSOR TO BLINK)
150,H                                    JTP     FCHECKR         DSP KEY MEANS 'HOLD IT'
151,H 172544  106 201 365                JTS     FDIAGEX         KBD KEY MEANS 'THAT'S ALL, FOLKS!'
152,H                              .
153,H                                    CALL    $$ONLINE        SEE IF DISK IS STILL ON-LINE
154,H 172547  056 320                    JTC     FDIAGEX         IT ISN'T, JUST RETURN
155,H 172551  106 071 366          .
156,H 172554  003                        CALL    $$SEEK          SEEK TO THE TRACK IN THE D-REGISTER
157,H 172555  153                  .                             BECAUSE OF CALL RESTDSK WILL DO RESTORE
158,H 172556  007                  .                             INITIALLY (AND ALWAYS)
159,H                                    LH      SEDSPBF>8       READ A SECTOR INTO THE DISPLAY
160,H 172557  060                        CALL    $$READ             BUFFER (MUST START ON A PAGE)
161,H 172560  106 337 374                RFC
162,H 172563  070                        EX      CLICK           NOTE THAT ERROR WAS FOUND (OF SOME SORT)
163,H 172564  106 303 375                RET
164,H 172567  024 060              *
165,H 172571  007                  FGETRSP POP     HL              HL => MESSAGE
166,H                                    CALL    DISPDOS         DISPLAY IT
167,H 172572  007 052 105 122 122        PUSH    HL              PUT RETURN ADDRESS BACK ON STACK
168,H                                    CALL    KEYCHAR         AND GET RESPONSE
157,                               .     SU      '0'             UNBIAS ASCII DIGIT
158,                                     RET
159,                               *
160,                               FAUTOMSG DC     $BP,!*ERR*!,$ES
161,                               .
162,                                     XIF
163,                               .
164,                                     IFEQ    ROMTYPE,2
                                         INC     1800RIMT
                                         XIF
                                   .
                                         IFLE    ROMTYPE,1
                                         INC     1800DSKR
```

```
3,I
4,I                     .
5,I                     . 1800 DISKETTE INTERFACE ROUTINES          MAY 10, 1978    12:13    HSP/HJS
6,I 172601              .
7,I                     $$SEEK
8,I                     .
9,I                     . SEEK (MOVE HEAD) TO A TRACK
10,I                    .
11,I                    .       ENTRY: D      = DESIRED TRACK NUMBER (0-76)
12,I                    .              $TRACK = CURRENT TRACK NUMBER (ASSUMED TO BE ACCURATE)
13,I                    .
14,I                    .       EXITS: ALL REGISTERS PRESERVED
15,I                    .              $TRACK = NEW CURRENT TRACK # (ASSUMING ENTRY VALUE ACCURATE)
16,I                    .              (TC)   = DRIVE OFF LINE OR RESTORE UNSUCCESSFUL
17,I                    .              (FC)   = SEEK COMPLETED
18,I                    .              5 EXTRA STACK LEVELS USED (MAX)
19,I 172601  106 371 366                .
20,I 172604  105 221                    CALL    $DSKSAVR        SAVE ALL REGISTERS AND LOAD X
21,I 172606  074 377                    PL      A,$TRACK        PICK UP LAST-KNOWN TRACK
22,I 172610  110 220 365                CP      -1              JUMP IF TRACK IS KNOWN
23,I                                    JFZ     $SEEKOK
24,I 172613  106 233 366                .
25,I                                    CALL    $RESTORX        ELSE RESTORE (DON'T SAVE REGS AGAIN)
26,I                                    .                       NOTE: THIS CALL HAS CALL BACK TO SEEK
27,I 172616  043                        .                             AT LABEL SEEKOUT
28,I                                    RTC                     CATCH DRIVE OFF LINE OR BAD RESTORE
29,I 172617  250                        .
30,I 172620  223            $SEEKOK     XRA                     ELSE SET THE TRACK TO ZERO
31,I 172621  053                        SUD                     (ACTUAL)-(DESIRED)
32,I                                    RTZ                     DISK IS ALREADY THERE
33,I 172622  320                        .
34,I 172623  136 221                    LCA                     SAVE THE DIFFERENCE
35,I 172625  006 040                    PS      D,$TRACK        SAVE THE DESIRED TRACK NUMBER
36,I 172627  120 237 365                LA      FOMVOT          MOVE OUT TOWARD THE RIM
37,I                                    JFS     $SEEKLOP           IF DIFFERENCE POSITIVE
38,I 172632  250                        .
39,I                                    XRA                     ELSE GET 2'S COMP OF NEG DIFFERENCE
40,I 172633  222            $SEEKOUT    .
41,I                                    SUC                     - ENTER HERE FROM $$RESTORE -
42,I 172634  320                        .           5 EXTRA STACK LEVELS USED (MAX) & WAIT OVERHEAD
43,I 172635  006 020                    LCA
44,I                                    LA      FOMVIN          AND MOVE IN TOWARD THE CENTER
45,I 172637  106 252 365    $SEEKLOP    .
46,I 172642  043                        CALL    $$STEP          STEP IN OR OUT BASED ON A-REG
47,I                                    RTC                     EXIT IF DRIVE OFF LINE
48,I 172643  062 024 001                .
49,I 172646  110 237 365                SUC     1               ELSE DECREMENT THE DISTANCE TO GO
50,I                                    JFZ     $SEEKLOP        KEEP STEPPING IF NOT ZERO
51,I 172651  007                        .
                                        RET                     ELSE RESTORE REGISTERS AND RETURN
```

```
52,I
53,I 172652                          +
54,I                                 $$STEP
55,I                                 .
56,I                                 . STEP IN OR OUT ONE TRACK
57,I                                 .
58,I                                 .        ENTRY: A       = OPERATION (FOMVIN/FOMVOUT)
59,I                                 .               X       = $DATA$>8
60,I                                 .               $DRVNUM = DRIVE NUMBER (FODR0/FODR1)
61,I                                 .               $DEVADR = MICRO-BUS ADDRESS
62,I                                 .
63,I                                 .        EXITS: ALL REGISTERS PRESERVED
64,I                                 .               (TC)(FZ) = DRIVE OFF LINE
65,I                                 .               (FC)(TZ) = TRACK 0 DETECTED
66,I                                 .               (FC)(FZ) = TRACK 0 NOT DETECTED
67,I                                 .               1 EXTRA STACK LEVELS USED (MAX) & WAIT OVERHEAD
68,I 172652  107 203                 .
69,I 172654  116 206                          PS      A,$$AVXA        SAVE THE A-REGISTER
70,I 172656  114 223                          PS      B,$$AVBC+1      SAVE THE B-REGISTER
71,I                                          PL      B,$DRVNUM       B = DRIVE SELECT (1 OR 2)
72,I 172660  111 260                 ...      NDB     -1-FODR0-FODR1  ONLY THE BOTTOM 2 BITS ARE OK
73,I 172662  105 222                          ORAB                    'OR' IN THE OUTPUT COMMAND
74,I 172664  064 220                          PL      A,$DEVADR       GET THE MICRO-BUS ADDRESS
75,I 172666  145                              OR      FCOMOD          'OR' IN THE MODE
76,I                                          UBOUT
77,I 172667  106 102 357             *
78,I                                 $STEPWTL CALL    SVDISKWS        CALL THE "WAIT" ROUTINE
79,I 172672  105 222                 .
80,I                                          PL      A,$DEVADR       GET THE BUS ADDRESS
81,I 172674  111 145                 ...      OR      FCINST          INPUT STATUS (= 0)
82,I                                          UBIN                    WAIT FOR STEP TO COMPLETE
83,I 172676  301                     .
84,I 172677  044 001                          LAB                     CATCH DRIVE OFF LINE
85,I 172701  152 275 366                      ND      FSONLN
86,I 172704  140 322 365                      CTZ     TOFFTL          TURN OFF THE LIGHT AND SET
87,I                                          JTC     $STEPXIT           CARRY TRUE IF OFF LINE
88,I 172707  301                     .
89,I 172710  044 040                          LAB                     WAIT IF STEP IN PROGRESS
90,I 172712  110 267 365                      ND      FSSTIP
91,I                                          JFZ     $STEPWTL
92,I 172715  301                     .
93,I 172716  044 100                          LAB                     ELSE SEE IF TRACK 0 DETECTED
94,I 172720  054 100                          ND      FSTRK0
95,I                                          XR      FSTRK0          EXIT FALSE CARRY TRUE ZERO IF SO
96,I 172722  114 206                 .
97,I 172724  105 203                 $STEPXIT PL      B,$$AVBC+1      RESTORE (B)
98,I 172726  007                              PL      A,$$AVXA        RESTORE (A)
                                              RET
```

```
 99,I
100,I 172727                          *
101,I                                 $$STATUS
102,I                                 *
103,I                                 * GET THE DISKETTE STATUS
104,I                                 *
105,I                                 *       NOTE:  THIS ROUTINE SHOULD NOT BE CALLED IF TRANSFERS ARE STILL
106,I                                 *              IN PROGRESS, (NORMALLY DOESN'T HAPPEN, ALL I/O INSIDE $$DOIO)
107,I                                 *
108,I                                 *       ENTRY: $DEVADR = MICRO-BUS ADDRESS
109,I                                 *              $DRVNUM = DRIVE NUMBER (FODR0/FODR1)
110,I                                 *
111,I                                 *       EXITS: B = DEVICE STATUS
112,I                                 *              1 EXTRA STACK LEVELS USED (MAX)
113,I 172727  022 070                 *
114,I 172731  076 335                         PUSH    XA              SAVE THE XA REGISTER PAIR
115,I 172733  105 222                         LX      $DATA$>8        SET UP THE PAGE REGISTER
116,I                                 STATWAIT PL     A,$DEVADR       SEE IF THE CONTROLLER IS IN ACTION
117,I 172735  111 145                 ...     OR      FCINST             BY CHECKING ITS STATUS
118,I 172737  301                             UBIN                    THIS CAUSES A WAIT ON LAST ACTION IF
119,I 172740  044 003                         LAB                        ACCIDENTLY ENTER HERE TO CHANGE DRIVES
120,I 172742  074 003                         ND      FSONLN+FSTRIP   IF BOTH ONLINE AND TRANSFER IN PROGRESS
121,I 172744  152 102 357                     CP      FSONLN+FSTRIP      ARE SET (SOMETHING GOING ON)
122,I 172747  150 333 365                     CTZ     SVDISKW$           WAIT (CC NOT MODIFIED) AND
123,I                                         JTZ     STATWAIT           LOOP TILL ONE GOES AWAY
124,I 172752  106 275 366                 *   CALL    TOFFTL          THIS WILL CLEAR FSTRIP IF STILL SET
125,I                                 *
126,I 172755  105 222                         PL      A,$DEVADR       GET THE MICRO-BUS ADDRESS
127,I 172757  064 220                         OR      FCOMOD             SET MODE TO SELECT DRIVE
128,I 172761  114 223                         PL      B,$DRVNUM       B = DRIVE NUMBER (FODR0/FODR1)
129,I 172763  145                             UBOUT                      SELECT A DRIVE
130,I 172764  105 222                         PL      A,$DEVADR       GET THE MICRO-BUS ADDRESS
131,I                                 ...     OR      FCINST             INPUT STATUS
132,I 172766  111 145                         UBIN                    GET STATUS INTO (B)
133,I 172770  022 060                         POP     XA                 RESTORE THE XA REGISTER PAIR
134,I 172772  007                             RET
```

```
135,I
136,I 172773                          +
137,I                                 $$ONLINE
138,I                                 .
139,I                                 . CHECK FOR DRIVE ON-LINE AND READY
140,I                                 .
141,I                                 .         ENTRY: (SAME AS FOR $$STATUS)
142,I                                 .
143,I                                 .         EXITS: ALL REGISTERS PRESERVED
144,I                                 .                (TC)(FZ) - DRIVE OFF LINE OR NOT READY
145,I                                 .                (FC)(TZ) - DRIVE ON LINE AND WRITE ENABLED
146,I                                 .                (FC)(FZ) - DRIVE ON LINE AND WRITE PROTECTED
147,I                                 .                3 EXTRA STACK LEVELS USED (MAX)
148,I 172773  022 070                 .
149,I 172775  076 335                           PUSH    XA                  SAVE (XA)
150,I 172777  116 206                           LX      $DATA$>8
151,I 173001  106 327 365                       PS      B,$$AVBC+1          SAVE (B)
152,I 173004  301                               CALL    $$STATUS            GET DISKETTE STATUS
153,I 173005  044 001                           LAB                            IN THE A-REGISTER
154,I 173007  152 275 366                       ND      FSONLN                 (DRIVE MAY BE CHANGED BY $STATUS CALL)
155,I 173012  140 020 366                       CTZ     TOFFTL              TURN OFF THE LIGHT IF NOT READY
156,I                                           JTC     ONLINXIT            JUST EXIT IF NOT ON LINE
157,I 173015  111 044 010             .
158,I 173020  114 206                           NDB     FSFPRO              SET FALSE ZERO IF FILE PROTECTED
159,I 173022  022 060                 ONLINXIT  PL      B,$$AVBC+1          RESTORE (B)
160,I 173024  007                               POP     XA                  RESTORE (XA)
                                                RET
```

```
                                        +
                                        $$WRITE
                                        .
                                        . WRITE A SECTOR (DOUBLE DENSITY)
                                        .
                                        .       ENTRY: H = MSB OF MEMORY BUFFER
                                        .              D = TRACK TO BE WRITTEN
                                        .              E = SECTOR TO BE WRITTEN
                                        .              A = WRITE/ WRITE-VERIFY CONTROL
                                        .
                                        .       EXITS: ALL REGISTERS PRESERVED
                                        .              $TRACK  = TRACK WRITTEN
                                        .              $SECTOR = SECTOR WRITTEN
                                        .              (TC)    - MISSING D.C. GAP OR CRC ERROR DURING VERIFY
                                        .              (FC)    - DATA WRITTEN FROM THE PAGE POINTED TO BY H-REG
                                        .              3 EXTRA STACK LEVELS USED (MAX)
                                        .
000007                                  DSKWV   EQU     7               DISK WRITE/VERIFY
000006                                  DSKWRT  EQU     6               DISK WRITE - NO VERIFY
                                        .
173025  106 371 366                     CALL    $DSKSAVR        SAVE ALL REGISTERS AND LOAD X
173030  066 000                         LL      0               INITIALIZE (L) TO 0'TH BYTE
173032  105 222                         PL      A,$DEVADR       GET THE DEVICE ADDRESS
                                        .
173034  016 060                         LB      FDWPI           LOAD WRITE PREAMBLE
173036  141                             FDDATA
173037  043                             RTC                     CATCH OPERATION IN PROGRESS
                                        .
173040  016 050                         LB      FDOTD           LOAD BUFFER DOUBLE DENSITY
173042  141                             FDDATA
173043  043                             RTC                     CATCH OPERATION IN PROGRESS
                                        .
173044  174 134 212                     DPLR    DE,$DSKREGS-4   RESTORE THE TRACK/SECTOR
173047  016 050                         LB      FFWRITE+FFDBL   DO DOUBLE DENSITY WRITE
173051  106 135 366                     CALL    $$DOIO
173054  110 113 366                     JFZ     $DSKERR         CATCH NO D.C. GAP OR SECTOR NOT FOUND
                                        .
173057  105 211                         PL      A,$DSKREGS-5    GET THE OPERATION CODE
173061  074 006                         CP      DSKWRT          JUST EXIT TRUE ZERO FALSE CARRY
173063  053                             RTZ                        IF NON-VERIFIED WRITE OPERATION
```

```
                                *
                                . DO "WRITE-VERIFY" - RE-READ & COMPUTE CRC)
                                .
173064  006 030                         LA      FDVRD           DO DOUBLE DENSITY READ/VERIFY
173066  104 100 366                     JMP     READDATA
                                *
173071                          $$READ
                                .
                                . READ A SECTOR (DOUBLE DENSITY)
                                .
                                .       ENTRY: H = MSB OF MEMORY BUFFER
                                .              D = TRACK TO BE READ
                                .              E = SECTOR TO BE READ
                                .
                                .       EXITS: ALL REGISTERS PRESERVED
                                .              $TRACK  = TRACK READ
                                .              $SECTOR = SECTOR READ
                                .              (TC)    = MISSING D,C, GAP OR CRC ERROR DURING READ
                                .              (FC)    = DATA READ INTO THE PAGE POINTED TO BY THE H-REG
                                .              3 EXTRA STACK LEVELS USED (MAX)
                                .
173071  106 371 366                     CALL    $DSKSAVR        SAVE ALL REGISTERS AND LOAD X
173074  066 000                         LL      0               MAKE HL => START OF BUFFER PAGE
173076  006 010                         LA      FDIND           CAUSE INPUT OF DOUBLE DENSITY DATA
                                .
173100  106 133 366             READDATA CALL   $$DOIODR        DO DOUBLE DENSITY READ
173103  110 113 366                     JFZ     $DSKERR         CATCH SOMETHING WRONG
                                .
173106  310                             LBA
173107  105 222                         PL      A,$DEVADR       GET THE DEVICE ADDRESS
173111  141                             FDDATA                  GET THE DATA CHECKING CRC
173112  003                             RFC                     EXIT IF DATA READ OR VERIFY OKAY
                                .
173113  006 200                 $DSKERR LA      0200            SET CARRY TRUE
173115  200                             ADA
173116  007                             RET
                                *
173117                          $$BUFIO
                                .
                                . PERFORM AN FDDATA OPERATION
                                .
                                .       ENTRY: B        = SUBFUNCTION NUMBER
                                .              C        = VALUE USED BY FDDATA (IF B = FDOUT)
                                .              HL       = LOCATION OF DATA AREA IN MEMORY (IF USED)
                                .              $DEVADR  = MICRO-BUS ADDRESS
                                .
                                .       EXITS: X        = MSB OF $DATA$
                                .              A,C      - SCRATCHED
                                .              B        - CHANGED (LEAST SIGNIFICANT TWO BITS)
                                .              2 EXTRA STACK LEVELS USED (MAX)
                                .
                                .       NOTE:  !! TO MAKE ROOM, MAY WANT TO TRIM THIS ROUTINE DOWN TO
```

```
253,I                                   .               SIMPLY FDDATA INSTRUCTION AND RETURN.
254,I                                   .               I.E. ALL REGS MUST BE INITIALIZED AND NONE SAVED
255,I                                   .
256,I 173117  076 335                           LX      $DATA$>8
257,I 173121  105 222                           PL      A,$DEVADR
258,I 173123  174 070                           PUSH    DE
259,I 173125  070                               PUSH    HL
260,I 173126  141                               FDDATA
261,I 173127  060                               POP     HL
262,I 173130  174 060                           POP     DE
263,I 173132  007                               RET
264,I                                   *
265,I                                   . DO A DISK READ/WRITE OPERATION (FDCMD)
266,I                                   .
267,I                                   .       ENTRY: X = $DATA$>8
268,I                                   .              B = SUBFUNCTION TO BE PERFORMED
269,I                                   .              D = TRACK NUMBER
270,I                                   .              E = SECTOR NUMBER
271,I                                   .
272,I                                   .       EXITS: ALL REGISTERS PRESERVED
273,I                                   .              $TRACK  = TRACK USED (IF VALID)
274,I                                   .              $SECTOR = SECTOR USED (IF VALID)
275,I                                   .              (TZ)    = OPERATION SUCCESSFUL
276,I                                   .              (FZ)    = OPERATION NOT SUCCESSFUL
277,I                                   .              1 EXTRA STACK LEVELS USED (MAX) & WAIT OVERHEAD
278,I                                   .
279,I 173133  016 010                   $$DOIODR LB     FFREAD+FFDBL    DO DOUBLE DENSITY READ
280,I                                   .
281,I 173135  107 203                   $$DOIO   PS     A,$SAVXA        SAVE (A)
282,I 173137  126 204                           PS      C,$SAVXA+1      SAVE (C)
283,I 173141  113 146 205                       DPS     DE,$SAVBC       SAVE (DE)
284,I                                   .
285,I 173144  105 222                           PL      A,$DEVADR       GET MICRO-BUS ADDRESS
286,I 173146  124 223                           PL      C,$DRVNUM       GET DRIVE NUMBER (FODR0/FODR1)
287,I                                   .
288,I 173150  111 141                           FDCMD                   DO THE I/O OPERATION
289,I 173152  140 221 366                       JTC     $DOIOERR        CATCH SOMETHING WRONG
290,I                                   .
291,I 173155  106 102 357               $DOIOW   CALL   SVDISKW$        DO A WAIT
292,I 173160  105 222                           PL      A,$DEVADR
293,I 173162  341                               LEB                     SAVE THE B-REGISTER
294,I 173163  111 145                           UBIN
295,I 173165  301                               LAB                     A = STATUS
296,I 173166  314                               LBE                     RESTORE THE B-REGISTER
297,I 173167  044 001                           ND      FSONLN          CATCH DRIVE OFF LINE
298,I 173171  150 221 366                       JTZ     $DOIOERR
299,I                                   .
300,I 173174  143                               FDSTAT                  RETURN DE = SECTOR/TRACK USED
301,I 173175  320                               LCA                             A = STATUS
302,I 173176  044 002                           ND      FRBUSY          WAIT IF BUSY
303,I 173200  110 155 366                       JFZ     $DOIOW
304,I                                   .
```

```
305,I 173203  113 146 220                DPS       DE,$SECTOR        ELSE SAVE THE SECTOR/TRACK USED
306,I 173206  062 044 004                NDC       FRINDX            EXIT TRUE ZERO IF NOT 4 INDEXES
307,I                                    .
308,I 173211  113 144 205    $DOIOXIT    DPL       DE,$SAVBC         RESTORE (DE)
309,I 173214  124 204                    PL        C,$SAVXA+1        RESTORE (C)
310,I 173216  105 203                    PL        A,$SAVXA          RESTORE (A)
311,I 173220  007                        RET
312,I                                    *
313,I                                    . DO A STATUS CLEAR AND CAUSE RETURN FALSE ZERO
314,I                                    .
315,I 173221  111 143        $DOIOERR    FDSCLR
316,I 173223  064 001                    OR        1
317,I 173225  104 211 366                JMP       $DOIOXIT
318,I                                    *
319,I 173230                             $$RESTORE
320,I                                    .
321,I                                    . SEEK TRACK 0 ON SELECTED DRIVE
322,I                                    .
323,I                                    .         ENTRY: $DRVNUM = DRIVE NUMBER (FODR0/FODR1)
324,I                                    .                $DEVADR = PHYSICAL DEVICE ADDRESS
325,I                                    .
326,I                                    .         EXITS: ALL REGISTERS PRESERVED
327,I                                    .                (TC) - DRIVE OFF LINE OR TRACK 0 NOT FOUND
328,I                                    .                (FC) - DISKETTE AT TRACK 0 AND $TRACK = 0
329,I                                    .                4 EXTRA STACK LEVELS USED (MAX) & WAIT OVERHEAD
330,I                                    .
331,I 173230  106 371 366                         CALL      $DSKSAVR          SAVE THE REGISTERS AND SET UP X
332,I                                    .
333,I 173233  250            $RESTORX    XRA                                 - ENTERED HERE FROM $$SEEK -
334,I 173234  107 221                    PS        A,$TRACK          SET THE TRACK NUMBER TO 0
335,I 173236  026 374                    LC        -4                CAUSE "SEEK" TO STEP IN 4 TRACKS
336,I 173240  106 233 365                CALL      $SEEKOUT
337,I 173243  043                        RTC                         CATCH DRIVE OFF LINE
338,I                                    .
339,I 173244  026 117                    LC        MAXTRAK+3         THEN STEP OUT UNTIL TRACK 0
340,I                                    .                             OR THE MAXIMUM NUMBER OF TRACKS
341,I                                    .                             (+3 AS A SAFETY OF PAST TRACK 77)
342,I 173246  006 040        $RESTORL    LA        FOMVOT            STEP OUT
343,I 173250  106 252 365                CALL      $$STEP
344,I 173253  043                        RTC                         CATCH DRIVE OFF LINE
345,I 173254  053                        RTZ                         EXIT RESTORING REGS WHEN TRACK 0
346,I                                    .
347,I 173255  062 024 001                SUC       1                 ELSE DECREMENT TRACK COUNT
348,I 173260  100 246 366                JFC       $RESTORL          KEEP GOING IF NOT OUT OF RANGE
349,I                                    .                             ELSE EXIT TRUE CARRY FALSE ZERO
350,I 173263  007                        RET                         RESTORE REGISTERS AND EXIT
```

```
351,I
352,I 173264                    +
353,I                           TOFFTLDE
354,I                           .
355,I                           . TURN OFF THE LIGHT IN THE SPECIFIED DRIVE
356,I                           .
357,I                           .       ENTRY: D = DRIVE SELECT CODE (1 OR 2)
358,I                           .              E = DRIVE CONTROLLER NUMBER (0 THRU 3)
359,I                           .
360,I                           .       EXITS: X       = $DATA$ PAGE
361,I                           .              $DEVADR = DRIVE CONTROLLER NUMBER
362,I                           .              A,B     = SCRATCHED
363,I 173264  076 335           .
364,I 173266  146 222                  LX        $DEVADR>8
365,I 173270  304                      PS        E,$DEVADR       SET THE CONTROLLER ADDRESS
366,I 173271  064 220                  LAE                       SELECT THE DRIVE IN (D)
367,I 173273  313                      OR        FCOMOD
368,I 173274  145                      LBD
369,I                                  UBOUT
370,I                           .                                 FALLS THROUGH INTO "TOFFL"
371,I 173275                    *
372,I                           TOFFTL
373,I                           .
374,I                           . TURN OFF THE LIGHT AND RETURN TRUE CARRY FALSE ZERO
375,I                           .    ALSO WILL CLEAR ALL INTERRUPTS PENDING AND
376,I                           .    RESET THE MASTER READ/WRITE FLIP/FLOPS
377,I                           .
378,I                           .       ENTRY: X       = $DATA$ PAGE
379,I                           .              $DEVADR = DEVICE (DRIVE ALREADY SELECTED)
380,I                           .
381,I                           .       EXITS: LIGHT TURNED OFF, CLEARED ALL INTERRUPTS & RESET MASTER FLOPS
382,I                           .              (TC)(FZ)
383,I 173275  105 222           .
384,I 173277  064 060                  PL        A,$DEVADR
385,I 173301  016 177                  OR        FCLEAR
386,I 173303  145                      LB        FKLOFF+FKMAST
387,I 173304  006 201                  UBOUT
388,I 173306  200                      LA        0201
389,I 173307  007                      ADA
                                       RET
```

```
390,I
391,I 173310                          +
392,I                                 $$DOSDRIV
393,I                                 .
394,I                                 . SWAP IN DOS LOGICAL DRIVE INFO.
395,I                                 .
396,I                                 .       ENTRY: A = DOS LOGICAL DRIVE NUMBER (0-7)
397,I                                 .
398,I                                 .       EXITS: ALL REGISTERS PRESERVED
399,I                                 .              $LOGDRV = SELECTED LOGICAL DRIVE (0 THRU 7)
400,I                                 .              $TRACK  = CURRENT TRACK FOR THAT DRIVE
401,I                                 .              $SECTOR = CURRENT SECTOR FOR THAT DRIVE
402,I                                 .              $DRVNUM = DRIVE SELECT CODE (1 OR 2)
403,I                                 .              $DEVADR = DRIVE MICRO-BUS ADDRESS (0, 1, 2, OR 3)
404,I                                 .              2 EXTRA STACK LEVELS USED (WAX)
405,I 173310  106 371 366             .       CALL    $DSKSAVR        SAVE THE REGISTERS AND SET UP X
406,I                                 .
407,I 173313  105 207                 .       PL      A,$DSKREGS-7    GET THE A-REGISTER BACK
408,I 173315  044 007                 .       ND      $DRVTBLN        LIMIT TO THE MAXIMUM DRIVE NUMBER
409,I 173317  310                     .       LBA                     SAVE THE DRIVE NUMBER IN B-REG
410,I                                 .
411,I 173320  105 217                 .       PL      A,$LOGDRV       GET CURRENT DOS LOGICAL DRIVE NUMBER
412,I 173322  271                     .       CPB                     EXIT RESTORING REGISTERS
413,I 173323  053                     .       RTZ                        IF SAME DRIVE AS LAST TIME
414,I                                 .
415,I 173324  056 335                 .       LH      $DRVTAB>8       SET UP THE H-REGISTER
416,I 173326  200                     .       ADA                     GENERATE INDEX INTO $DRVTAB
417,I 173327  140 343 366             .       JTC     $DOSDF0         CATCH NO CURRENT DRIVE
418,I                                 .
419,I 173332  044 016                 .       ND      $DRVTBLN<1      MAKE SURE $LOGDRV VALUE GOOD
420,I 173334  066 224                 .       LL      $DRVTAB         INDEX INTO THE $DRVTAB TABLE
421,I 173336  017                     .       INCP    HL,A
422,I 173337  113 144 220             .       DPL     DE,$SECTOR      GET CURRENT $SECTOR/$TRACK
423,I 173342  027                     .       DS      DE,HL           SAVE IT IN THE $DRVTAB ENTRY
424,I                                 .
425,I 173343  116 217                 $DOSDF0 PS      B,$LOGDRV       SAVE THE NEW LOGICAL DRIVE NUMBER
426,I 173345  066 224                 .       LL      $DRVTAB         INDEX INTO THE $DRVTAB
427,I 173347  301                     .       LAB
428,I 173350  200                     .       ADA
429,I 173351  017                     .       INCP    HL,A            (CARRY WILL BE CLEAR!! AFTER THIS)
430,I 173352  047                     .       DL      DE,HL           GET THE $SECTOR/$TRACK FOR NEW DRIVE
431,I 173353  113 146 220             .       DPS     DE,$SECTOR      STORE IT INTO THE CURRENT $SECTOR/$TRACK
432,I                                 .
433,I 173356  111 032                 .       SREB                    DIVIDE IT BY TWO & LSBIT TO CARRY
434,I 173360  006 001                 .       LA      FODR0           A = RIGHT HAND DRIVE IF CARRY CLEAR
435,I 173362  014 000                 .       AC      FODR1-FODR1     A = LEFT HAND DRIVE IF CARRY SET
436,I 173364  107 223                 .       PS      A,$DRVNUM       SAVE THE DRIVE SELECT
437,I 173366  116 222                 .       PS      B,$DEVADR       AND SAVE THE MICRO-BUS ADDRESS
438,I 173370  007                     .       RET                     RESTORE REGISTERS AND EXIT
```

```
                                *
                                $DSKSAVR
                                .
                                . DISK ROUTINE REGISTER SAVE AND PAGE REGISTER SET UP
                                .                 1 STACK LEVEL ADDITIONALLY CREATED
                                .
                                          PUSH      $DSKREGS
                                          REGS
                                          POP       XA              GET RID OF "$DSKREGS" ADR ON THE STACK
                                          POP       XA              XA = RETURN ADDRESS
                                          PUSH      $DSKRESR        MAKE RETURNS AFTER THIS GO TO "$DSKRESR"
                                          PUSH      XA              PUT RETURN ADR BACK ON THE STACK
                                          LX        $DATA$>8        SET UP THE PAGE REGISTER
                                          RET
                                *
                                $DSKRESR
                                .
                                . RESTORE THE DISK ROUTINE REGISTERS AND EXIT
                                .
                                          HL        $DSKREGS
                                          REGL
                                          RET
                                *
                                RESTDSK
                                .
                                . SET ALL DISK CONTROL VARIABLES SO ALL DRIVES MUST BE INITIALIZED (RESTORED)
                                .
                                          LA        -1              INITIALIZE ALL THE DISK VARIABLES
                                          MSA       *$DATA$           TO ALL 1'S
                                          LDH
                                          LE        $DATA$+1
                                          LC        $DATAL-1
                                          XRA
                                          LBA
                                          BT
                                          RET
                                          XIF
                                .
                                          IFEQ      ROMTYPE,0       (MAKE SPACE HERE SO LAST ROMS THE SAME)
                                          RPT       05
                                          DC        0
                                          XIF
                                          IFEQ      ROMTYPE,2       (MAKE SPACE HERE SO LAST ROMS THE SAME)
                                          RPT       0330
                                          DC        0
                                          XIF
                                .
                                          INC       1800DBUG
```

```
439,I
440,I 173371
441,I
442,I
443,I
444,I
445,I 173371  051 216 335
446,I 173374  055
447,I 173375  022 060
448,I 173377  022 060
449,I 173401  051 011 367
450,I 173404  022 070
451,I 173406  076 335
452,I 173410  007
453,I
454,I 173411
455,I
456,I
457,I
458,I 173411  066 216 056 335
459,I 173415  111 055
460,I 173417  007
461,I
462,I 173420
463,I
464,I
465,I
466,I 173420  006 377
467,I 173422  066 200 056 335 370
468,I 173427  335
469,I 173430  046 201
470,I 173432  026 043
471,I 173434  250
472,I 173435  310
473,I 173436  021
474,I 173437  007
165,
166,
167,
168,
169,  173440  000 000 000 000 000
170,
171,
172,
173,
174,
175,
176,
```

```
                                *
                                * 1800 DEBUG COMMAND INTERPRETER              JUN 14, 1978    12:00    HSP/HJS
                                *
173445                          BRKPNR
                                *
                                * BREAK POINT EXECUTION ENTRY
                                *
173445  106 242 372                     CALL    SSTATE          SAVE THE STATE OF THE MACHINE
                                *       LX      CURADR>8        INIT THE PAGE REGISTER (BY SSTATE)
173450  111 124 207                     DPL     BC,OLDTOS       GET THE BREAK POINT ADDRESS
173453  174 047                         DL      DE,BC
173455  174 035                         DECP    DE
173457  066 211 056 327                 HL      BPTABL-BPTES+1  SEARCH TABLE FOR ADDRESS
173463  117 015                 BRKPNL  INCP    HL,2            BUMP MEMORY POINTER TO NEXT ENTRY
173465  176 074 251                     CPL     BPTABE          STOP WHEN AT END OF TABLE
173470  150 120 367                     JTZ     BRKPNN
                                *
173473  307                             LAM     HL              ELSE SEE IF ADDRESSES MATCH
173474  015                             INCP    HL
173475  274                             CPE
173476  110 063 367                     JFZ     BRKPNL
                                *
173501  307                             LAM     HL
173502  273                             CPD
173503  110 063 367                     JFZ     BRKPNL
                                *
173506  015                             INCP    HL              RESTORE CONTENTS IF MATCH FOUND
173507  307                             LAM     HL
173510  174 370                         LMA     DE
173512  174 027                         DS      DE,BC           UPDATE TOS ENTRY WITH BP LOCATION
173514  035                             DECP    HL              CLEAR THE BREAK POINT
173515  006 377                         LA      -1
173517  370                             LMA
173520  113 146 200             BRKPNN  DPS     DE,CURADR       DISPLAY THE POINT AS CURRENT ADDRESS
173523  151                             EX      BEEP            MAKE SOME NOISE
173524  104 133 367                     JMP     DSPCAD          THEN GO TO DEBUG
                                *
173527                          DEBUG
                                *
                                * RETURN FROM DEBUG "CALL" ENTRY
                                *
173527  040                             DI                      MAKE SURE INTERRUPTS WON'T BOTHER ME
                                *                               INTERRUPTS MAY BE ON AFTER THE 'CALL'
173530  106 242 372                     CALL    SSTATE          SAVE THE MACHINE STATE
```

```
47,J
48,J 173533                     *
                                DEBUGI
49,J
                                *
50,J                            * ENTRY FROM "RESTART" INVOCATION
51,J                            *
52,J                                    IFS     ASCII
53,J                                    XRA                     CLEAR ASCII KEYIN MODE
54,J                            KEYINS  PS      A,KEYINF        SET ASCII KEYIN MODE FLAG TO A-REG
55,J                                    XIF
```

```
                                        *
                                        DSPCAD
                                        *
                                        * PUT UP DEBUG "CURRENT ADDRESS" DISPLAY
                                        *
                                                LX      CURADR>8        SET UP THE PAGE REGISTER (SAFTEY)
                                                IFS     ORIGINI
                                                DPL     DE,CURORG       DE = CURRENT ORIGIN
                                                LAE                     SET ZERO CONDX IF ORIGIN EQUALS ZERO
                                                ORD
                                                LE      BL=4            ERASE 4TH TO BOTTOM LINE
                                                LD      73                 IF ORIGIN IS ZERO
                                                CTZ     DOSF69          (ERASE TO END OF LINE)
                                                DPL     DE,CURORG       DE = CURRENT ORIGIN
                                                DPL     BC,CURADR       BC => CURRENTLY ADDRESSED SLOT
                                                SUEC                    COMPUTE THE PHYSICAL ADDRESS
                                                SBOB
                                                LAE
                                                ORD
                                                LE      BL=4            DISPLAY THE LOGICAL ADR ON 4TH TO BOTTOM
                                                CFZ     DSPOC6             LINE IF CURRENT ORIGIN IS NOT ZERO
                                                XIF
 173535  046 010                               LE      BL=3            DISPLAY THE PHYSICAL ADDRESS ON
 173537  111 124 200                           DPL     BC,CURADR          THE 3RD TO THE BOTTOM LINE
 173542  062 070                               PUSH    BC              SAVE THE PHYSICAL ADDRESS
 173544  106 154 372                           CALL    DSPOC6
 173547  060                                   POP     HL              RESTORE THE ADDRESS
 173550  070                                   PUSH    HL              LEAVE IT ON THE STACK
 173551  016 040                               LB      040
 173553  327                                   LCM                     GET THE BYTE TO BE OUTPUT
 173554  046 011                               LE      BL=2
 173556  036 111                               LD      73
 173560  106 312 374                           CALL    DSPCBL          (WRITE CHARACTERS IN BC=REGS)
 173563  371                                   LMB
 173564  015                                   INCP    HL
 173565  111 027                               DS      BC,HL           PUT OUT BLANK, CHARACTER, BLANK
 173567  066 003                               LL      3               DISPLAY THE CHARACTER VALUE
 173571  106 156 372                           CALL    DSPOCR          (B CLOSE ENOUGH TO 0 FOR DSPOCR OF 3)
 173574  060                                   POP     HL              RESTORE THE ADDRESS OF THE ITEM
 173575  111 047                               DL      BC,HL           DISPLAY THE 6 DIGIT CONTENTS
 173577  046 012                               LE      BL=1               ON THE NEXT TO BOTTOM LINE
 173601  106 154 372                           CALL    DSPOC6
 173604  051 133 367                           PUSH    DSPCAD
```

```
101,J
102,J 173607                            *
103,J                                   GETCMD
104,J                                   *
105,J                                   * GET THE NEXT COMMAND FROM THE KEYBOARD
106,J 173607  046 013 036 106           *
                                                DE      NECP<8+8L       GET THE COMMAND LINE CURSOR POSITION
107,J 173613  026 000                           LC      0               ZERO THE COMMAND VALUE ACCUMULATOR
108,J 173615  312                               LBC
109,J 173616  062 070                   GETCML  PUSH    BC              SAVE THE COMMAND VALUE SO FAR
110,J 173620  106 303 375                       CALL    KEYCHAR         GET CHAR FROM KEYBOARD FLASHING CURSOR
111,J 173623  062 060                           POP     BC              RESTORE THE COMMAND VALUE SO FAR
112,J 173625  076 327                           LX      CURADR>8        SET UP THE PAGE REGISTER (SAFTEY)
113,J                                   *
114,J                                           IFS     ASCII
115,J                                           PL      L,KEYINF        CATCH BEING IN ASCII KEYIN MODE
116,J                                           ORLL
117,J                                           JFZ     KEYINA          ENTER ASCII CHARACTER IF SO
118,J                                           XIF
119,J                                   *
120,J                                   * CHECK FOR SPECIAL COMMAND CHARACTERS
121,J                                   *
122,J 173627  074 015                           CP      ENTER           DECODE THE COMMAND
123,J 173631  150 247 370                       JTZ     NEWADR
124,J 173634  074 030                           CP      CAN             CANCEL COMMAND IF CANCEL KEY
125,J 173636  150 207 367                       JTZ     GETCMD
126,J 173641  074 010                           CP      BSP             BACKSPACE ONE IF BACKSPACE KEY
127,J 173643  150 131 370                       JTZ     BACKSP
128,J 173646  074 077                           CP      '?'             CATCH IDENTIFICATION INQUERY
129,J 173650  150 071 372                       JTZ     IDENT
130,J 173653  074 056                           CP      '.'             MODIFY AND INCREMENT
131,J 173655  150 340 370                       JTZ     MODINC
132,J 173660  074 136                           CP      '^'             MODIFY AND INCREMENT USING LAST VALUE
133,J 173662  150 324 370                       JTZ     MODAGN
134,J 173665  074 043                           CP      '##'            # = CLEAR ALL BREAK POINTS
135,J 173667  150 066 371                       JTZ     BPCLR           RESTORING VALUES
136,J 173672  074 101                           CP      'A'             TRY TO ACCUMULATE OCTALS IF NOT LETTERS
137,J 173674  140 150 370                       JTC     GETDIG
138,J 173677  074 173                           CP      'z'+1           (LOWER CASE 'Z')
139,J 173701  100 150 370                       JFC     GETDIG
140,J 173704  066 224 056 367                   HL      ='A'<1+CMDTS    HL => COMMAND LETTER TABLE
141,J 173710  074 133                           CP      'Z'+1           BASED ON SHIFT CASE
142,J 173712  140 326 367                       JTC     DOCMDL
143,J 173715  066 040 056 367                   HL      =' '<1+CMDTNS   (LOWER CASE 'A')
144,J 173721  074 141                           CP      ' '
145,J 173723  140 150 370                       JTC     GETDIG
```

```
146,J
147,J 173726
148,J
149,J
150,J
151,J
152,J 173726  200                     *
153,J 173727  017                     DOCMDL
154,J 173730  057                     .
155,J 173731  070                     . INDEX INTO THE ADDRESS TABLE POINTED TO BY (HL)
156,J 173732  012                     .       AND JUMP OFF TO THE INDICATED ADDRESS.
157,J 173733  115 164 211               .
158,J 173736  113 074 106               ADA                             INDEX INTO THE GIVEN TABLE
159,J 173741  007                       INCP    HL,A                    BY THE COMMAND LETTER
160,J                                   DL      HL,HL                   SET UP TO JUMP TO THE COMMAND
161,J 173742                            PUSH    HL
162,J                                   SRC                             RESTORE THE COMMAND LETTER
163,J                                   DPL     HL,OLDREGS              HL => REGISTER STORAGE
164,J                                   CPD     NECP                    SET TRUE ZERO IF NO DIGITS ENTERED
165,J                                   RET
166,J 173742  200 371                 *
167,J                                 CMDTNS
168,J                                 .
169,J                                 . UNSHIFTED COMMAND ROUTINE ADDRESS TABLE
170,J                                 .
171,J 173744  004 371                   IFNE    ROMTYPE,3
172,J 173746  360 371                   DA      ADRDEV                  A - ADDRESS THE GIVEN OR LAST I/O DEVICE
173,J 173750  257 370                   XIF
174,J 173752  326 371                   IFEQ    ROMTYPE,3
175,J                                   DA      GETCME                  A - * NOT USED * (5500 I/O BUS)
176,J 173754  200 371                   XIF
177,J 173756  200 371                   DA      BPSET                   B - SET A BREAK POINT AT GIVEN OR CURADR
178,J                                   DA      CALL                    C - CALL THE GIVEN OR CURRENT ADDRESS
179,J                                   DA      DECADR                  D - DECREMENT THE CURRENT ADDRESS
180,J                                   DA      EXECUT                  E - CONTINUE EXECUTION
181,J                                   IFNE    ROMTYPE,3
182,J                                   DA      ADRDEV                  F - FETCH NEXT DATA BYTE FROM CURRENT I/O
183,J                                   DA      ADRDEV                  G - GOTO DATA MODE IN THE CURRENT I/O
184,J 173760  205 364                   XIF
185,J                                   IFEQ    ROMTYPE,3
186,J                                   DA      GETCME                  F - * NOT USED * (5500 I/O BUS)
187,J                                   DA      GETCME                  G - * NOT USED * (5500 I/O BUS)
188,J                                   XIF
189,J                                   IFEQ    ROMTYPE,0
190,J                                   DA      FDIAG                   H - HARDWARE FLOPPY DIAGNOSTIC
191,J                                   XIF
192,J                                   IFEQ    ROMTYPE,1
193,J                                   DA      APFDMP                  H - APF MEMORY DUMP
194,J                                   XIF
195,J 173762  211 370                   IFEQ    ROMTYPE,2
196,J 173764  373 371                   DA      TSTRIM                  H - EXTERNAL RIM BUFFER TEST
197,J                                   XIF
                                        IFEQ    ROMTYPE,3
                                        DA      GETCME                  H - * NOT USED * (HARDWARE DIAGNOSTIC)
                                        XIF
                                        DA      INCADR                  I - INCREMENT THE CURRENT ADDRESS
                                        DA      JUMP                    J - JUMP TO THE GIVEN OR CURRENT ADDRESS
                                        IFS     ASCII
```

```
198,J
199,J
200,J
201,J 173766  125 370
202,J
203,J 173770  233 370
204,J 173772  343 370
205,J
206,J
207,J
208,J
209,J
210,J 173774  125 370
211,J 173776  125 370
212,J
213,J
214,J 174000  133 371
215,J
216,J 174002  160 371
217,J
218,J
219,J
220,J
221,J
222,J 174004  120 371
223,J 174006  357 370
224,J 174010  360 375
225,J 174012  314 371
226,J
227,J 174014  273 371
228,J 174016  247 371
229,J 174020  254 371
230,J 174022  261 371
231,J 174024  266 371
232,J
233,J
234,J
235,J
236,J
237,J
238,J
239,J
```

```
          DA        KEYINM          K - SET ASCII KEYIN MODE
          XIF
          IFC       ASCII
          DA        GETCME          K - * NOT USED * (ASCII KEYIN)
          XIF
          DA        LINK            L - LINK TO ADR POINTED TO BY CURADR
          DA        MODIFY          M - MODIFY THE CURRENT ADDRESS CONTENTS
          IFS       ORIGINI
          DA        NOSADR          N - SET NON-OFFSET ADDRESS
          DA        ORIGIN          O - CLEAR OR SELECT ORIGIN
          XIF
          IFC       ORIGINI
          DA        GETCME          N - * NOT USED * (ORIGIN OFFSET)
          DA        GETCME          O - * NOT USED * (ORIGIN SELECT)
          XIF
          IFNE      ROMTYPE,3
          DA        BASSET          P - DISPLAY BASE REGISTER
                                        OR LOAD WITH BC - 0100000
          DA        STLOAD          Q - LOAD THE SECTOR TABLE
          XIF
          IFEQ      ROMTYPE,3
          DA        GETCME          P - * NOT USED * (BASE REGISTER)
          DA        GETCME          Q - * NOT USED * (SECTOR TABLE)
          XIF
          DA        REGMOD          R - SWITCH ALPHA/BETA REGISTER MODE
          DA        STACKD          S - DISPLAY THE SPECIFIED STACK ITEM
          DA        TSTMEM          T - START MEMORY TEST ('12345T')
          DA        EXEUSR          U - USER MODE EXECUTE
          IFNE      ROMTYPE,3
          DA        OUTXC4          V - EX COM4 TO LAST I/O DEVICE
          DA        OUTXWT          W - EX WRITE TO LAST I/O DEVICE
          DA        OUTXC1          X - EX COM1 TO LAST I/O DEVICE
          DA        OUTXC2          Y - EX COM2 TO LAST I/O DEVICE
          DA        OUTXC3          Z - EX COM3 TO LAST I/O DEVICE
          XIF
          IFEQ      ROMTYPE,3
          DA        GETCME          V - * NOT USED * (5500 I/O BUS)
          DA        GETCME          W - * NOT USED * (5500 I/O BUS)
          DA        GETCME          X - * NOT USED * (5500 I/O BUS)
          DA        GETCME          Y - * NOT USED * (5500 I/O BUS)
          DA        GETCME          Z - * NOT USED * (5500 I/O BUS)
          XIF
```

```
240,J
241,J 174026
242,J
243,J
244,J
245,J 174026  310 370
246,J 174030  303 370
247,J 174032  302 370
248,J 174034  305 370
249,J 174036  304 370
250,J 174040  301 370
251,J 174042  125 370
252,J 174044  307 370
253,J 174046  321 371
254,J 174050  037 372
255,J 174052  125 370
256,J 174054  306 370
257,J 174056  125 370
258,J 174060  125 370
259,J
260,J
261,J
262,J
263,J 174062  125 370
264,J
265,J
266,J 174064  133 371
267,J
268,J
269,J
270,J
271,J 174066  125 370
272,J
273,J
274,J
275,J
276,J
277,J 174070  125 370
278,J 174072  125 370
279,J
280,J 174074  125 370
281,J 174076  125 370
282,J 174100  125 370
283,J 174102  125 370
284,J 174104  311 370
285,J
286,J 174106  235 371
287,J 174110  242 371
288,J
289,J
290,J
291.J
```

```
*
CMDTS
.
. SHIFTED COMMAND ROUTINE ADDRESS TABLE
.
        DA      REGA            A - REGISTER DISPLAY
        DA      REGB            B - REGISTER DISPLAY
        DA      REGC            C - REGISTER DISPLAY
        DA      REGD            D - REGISTER DISPLAY
        DA      REGE            E - REGISTER DISPLAY
        DA      REGF            F - CONDITION FLAGS DISPLAY
        DA      GETCME          G - * NOT USED *
        DA      REGH            H - REGISTER DISPLAY
        DA      EXECUI          I - 'E' COMMAND WITH EI/RET
        DA      TSTDSP          J - DISPLAY TEST
        DA      GETCME          K - * NOT USED * (KEYBOARD TEST)
        DA      REGL            L - REGISTER DISPLAY
        DA      GETCME          M - * NOT USED *
        DA      GETCME          N - * NOT USED *
        IFS     ORIGINI
        DA      ORIGIM          O - MODIFY SELECTED ORIGIN
        XIF
        IFC     ORIGINI
        DA      GETCME          O - * NOT USED * (ORIGIN MODIFY)
        XIF
        IFNE    ROMTYPE,3
        DA      BASSET          P - DISPLAY BASE REG OR LOAD WITH C
        XIF
        IFEQ    ROMTYPE,3
        DA      GETCME          P - * NOT USED * (BASE REGISTER)
        XIF
        DA      GETCME          Q - * NOT USED *
        IFS     STACKP
        DA      STACKR          R - STACK ROLL
        DA      STACKS          S - STACK STUFF
        XIF
        IFC     STACKP
        DA      GETCME          R - * NOT USED * (STACK ROLL)
        DA      GETCME          S - * NOT USED * (STACK STUFF)
        XIF
        DA      GETCME          T - * NOT USED * (PSEUDO RANDOM MEM TEST)
        DA      GETCME          U - * NOT USED *
        DA      GETCME          V - * NOT USED *
        DA      GETCME          W - * NOT USED *
        DA      REGX            X - REGISTER DISPLAY
        IFNE    ROMTYPE,3
        DA      OUTXST          Y - EX STATUS
        DA      OUTXDA          Z - EX DATA
        XIF
        IFEQ    ROMTYPE,3
        DA      GETCME          Y - * NOT USED * (5500 I/O BUS)
        DA      GETCME          Z - * NOT USED * (5500 I/O BUS)
```

```
                                XIF
                *
C12345
                .
                . CHECK FOR '12345' FOR COMMAND ENTRY VALUE
                .
                        CPC     012345          CHECK LSB
                        JFZ     C1234P          ABORT IF NOT RIGHT
                        CPB     012345>8        ELSE CHECK MSB
                        RTZ                     RETURN IF BOTH CORRECT
C1234P          POP                             ELSE ABORT THE RETURN
                *
GETCME
                .
                . COMMAND ERROR
                .
                        EX      BEEP            MAKE NOISE IF BAD COMMAND
                        JMP     GETCML
                *
BACKSP
                .
                . BACKSPACE IF NOT AT BEGINNING OF LINE
                .
                        CPD     NECP            DON'T DO ANYTHING IF AT BEGINNING OF LINE
                        JTZ     GETCMD
                        SUD     1               ELSE DECREMENT THE HORIZONTAL POSITION
                        CALL    SBCRL3          AND SHIFT BC BACK RIGHT THREE PLACES
                        JMP     GETCML          THEN CONTINUE THE COMMAND
                *
GETDIG
                .
                . CONVERT OCTAL DIGITS ENTERED BEFORE COMMAND LETTER TO BINARY
                .
                .       LLA                     SAVE THE CHARACTER ENTERED
                        SU      '0'             SEE IF LEGAL ASCII DIGIT
                        JTS     GETCME          COMPLAIN IF NOT
                        CP      8
                        JFS     GETCME
                        ADCC                    ELSE CONVERT TO BINARY
                        ACBB                    BY SHIFTING BC LEFT 3
                        ADCC
                        ACBB
                        ADCC
                        ACBB
                        ORAC                    AND OR'ING IN THE DIGIT
                        CPD     MAXPOS-1
                        ACD     0
                        JMP     GETCML
```

```
342,J                           IFS     ASCII
343,J                   *
344,J                   KEYINM
345,J                   .
346,J                   . SET ASCII KEYIN MODE
347,J                   .
348,J                           CALL    C12345          ENTRY MUST BE '12345K'
349,J                           HL      KEYMSG          DISPLAY KEYIN MODE MESSAGE
350,J                           DE      61<8+BL         POSITION CURSOR FOR MESSAGE
351,J                           CALL    DISPLAY         DISPLAY OR CLEAR THE MESSAGE
352,J                           LA      0377            SET THE KEYIN MODE FLAG
353,J                           POP                     (KEEP THE STACK CORRECT)
354,J                           JMP     KEYINS
355,J                   .
356,J                   KEYINC  HL      KEYMSC          RESET THE KEYIN MODE
357,J                           DE      61<8+BL
358,J                           CALL    DISPLAY         BLANK THE LINE
359,J                           XRA
360,J                           POP                     (KEEP THE STACK CORRECT)
361,J                           JMP     KEYINS          DONE
362,J                   .
363,J                   KEYMSG  DC      ' * ASCII'      KEYIN MODE MESSAGE
364,J                   KEYMSC  DC      $EEOL,SES
365,J                   *
366,J                   KEYINA
367,J                   .
368,J                   . PROCESS NEXT CHARACTER IN ASCII KEYIN MODE
369,J                   .
370,J                           CP      CAN             EXIT THE MODE IF CANCEL ENTERED
371,J                           JTZ     KEYINC
372,J                           CP      BSP             BACK UP ONE LOCATION IF BACKSPACE ENTERED
373,J                           JTZ     DECADR
374,J                           CP      DEL             GO FORWARD ONE LOCATION IF DEL ENTERED
375,J                           JTZ     INCADR
376,J                           CP      'A'             ELSE DO SHIFT KEY INVERSION
377,J                           JTC     KEYINN
378,J                           CP      'Z'+1
379,J                           JTC     KEYINV
380,J                           CP      ' '
381,J                           JTC     KEYINN
382,J                           CP      ' '+1
383,J                           JFC     KEYINN
384,J                   KEYINV  XR      040             INVERT SHIFT IF ALPHA CHARACTER
385,J                   KEYINN  DPL     HL,CURADR       STORE THE ASCII CHAR IN CURRENT ADDRESS
386,J                           LMA                     AND THEN BUMP THE CURRENT ADDRESS
387,J                           XRA
388,J                           EX      CLICK           MAKE SOME NOISE IF ASCII CHARACTER
389,J                           XIF
```

```
390,J
391,J 174211                        *
392,J                               INCADR
393,J                               .
394,J                               . INCREMENT THE CURRENT ADDRESS BY ONE OR THE VALUE GIVEN
395,J 174211  115 164 200           .
396,J 174214  110 223 370                     DPL     HL,CURADR
397,J 174217  026 001 016 000                 JFZ     INCADV
398,J 174223  176 202                         BC      1
399,J 174225  115 211               INCADV    ADCL
400,J 174227  115 166 200                     ACBH
401,J 174232  007                   MODIFNXT  DPS     HL,CURADR
402,J                                         RET
403,J 174233                        *
404,J                               LINK
405,J                               .
406,J                               . LINK TO THE ADDRESS POINTED TO BY THE CURRENT ADDRESS
407,J 174233  110 241 370           .
408,J 174236  111 124 200                     JFZ     LINKADR         USE ADDRESS GIVEN
409,J 174241  174 047                         DPL     BC,CURADR       NO, USE CURRENT ADDRESS
410,J 174243  113 146 200           LINKADR   DL      DE,BC
411,J 174246  007                             DPS     DE,CURADR
412,J                                         RET
413,J 174247                        *
414,J                               NEWADR
415,J                               .
416,J                               . SET THE CURRENT ADDRESS
417,J 174247  113 074 106           .
418,J 174252  053                             CPD     NECP            IGNORE ENTER ONLY
419,J                                         RTZ
420,J                                         IFS     ORIGINI
421,J                                         DPL     DE,CURORG       ELSE ADD ON THE CURRENT ORIGIN
422,J                                         ADEC                    VALUE
423,J                                         ACDB
424,J 174253  111 126 200           NOSADR    XIF
425,J 174256  007                             DPS     BC,CURADR
426,J                                         RET
427,J 174257                        *
428,J                               DECADR
429,J                               .
430,J                               . DECREMENT THE CURRENT ADDRESS BY ONE OR THE VALUE GIVEN
431,J 174257  115 164 200           .
432,J 174262  110 271 370                     DPL     HL,CURADR
433,J 174265  026 001 016 000                 JFZ     DECADV
434,J 174271  176 222                         BC      1
435,J 174273  115 231               DECADV    SUCL
436,J 174275  115 166 200                     SBBH
437,J 174300  007                             DPS     HL,CURADR
                                              RET
```

```
438,J                                   *
439,J                                   . REGISTER DISPLAYS
440,J                                   .
441,J 174301  035                       REGF    DECP    HL
442,J 174302  035                       REGC    DECP    HL
443,J 174303  035                       REGB    DECP    HL
444,J 174304  035                       REGE    DECP    HL
445,J 174305  035                       REGD    DECP    HL
446,J 174306  035                       REGL    DECP    HL
447,J 174307  035                       REGH    DECP    HL
448,J 174310  035                       REGA    DECP    HL
449,J 174311  035                       REGX    DECP    HL
450,J                                   .
451,J 174312  115 166 200                REGDSP  DPS     HL,CURADR       DISPLAY THAT REGISTER CONTENT
452,J 174315  113 074 106                       CPD     NECP            JUST DISPLAY IF NO DIGITS ENTERED
453,J 174320  053                               RTZ
454,J 174321  104 343 370                       JMP     MODIFY          OTHERWISE MODIFY
455,J                                   *
456,J 174324                            MODAGN
457,J                                   .
458,J                                   . 'A' MODIFY AND INCREMENT THE ADDRESS USING NEW OR PREVIOUSLY SET VALUE
459,J                                   .
460,J 174324  113 074 106                       CPD     NECP            CHECK IF DIGITS ARE GIVEN
461,J 174327  150 335 370                       JTZ     MODIFX
462,J 174332  111 126 204                       DPS     BC,MODVAL       UPDATE TO NEW VALUE GIVEN
463,J 174335  111 124 204               MODIFX  DPL     BC,MODVAL       OR USE OLD VALUES
464,J                                   *
465,J 174340                            MODINC
466,J                                   .
467,J                                   . '.' MODIFY AND INCREMENT THE ADDRESS
468,J                                   .
469,J 174340  051 227 370                       PUSH    MODIFNXT        RETURN TO SAVE ADDRESS ROUTINE
470,J                                   *
471,J 174343                            MODIFY
472,J                                   .
473,J                                   . MODIFY THE CURRENTLY ADDRESSED MEMORY LOCATION
474,J                                   .
475,J 174343  115 164 200                       DPL     HL,CURADR       STORE LSB WHERE CURRENT ADDRESS POINTS
476,J 174346  372                               LMC     HL
477,J 174347  015                               INCP    HL              (INCREMENT ADDRESS BY ONE - IF NEEDED)
478,J 174350  111 261                           ORBR
479,J 174352  053                               RTZ                     THAT'S IT IF MSB IS ZERO
480,J 174353  371                               LMB                     ELSE STORE MSB IN FOLLOWING LOCATION
481,J 174354  015                               INCP    HL              (INCREMENT ADDRESS BY TWO - IF NEEDED)
482,J 174355  153                               EX      CLICK           MAKE NOISE IF TWO BYTES STORED
483,J 174356  007                               RET
484,J                                   .
485,J                                           IFS     ORIGINI
486,J                                   +
487,J                                   ORIGIN
488,J                                   .
489,J                                   . SET THE CURRENT ORIGIN TO THE TABLE ENTRY SELECTED
```

```
490,J
491,J                   *
492,J                           JTZ     ORIGIC          JUST CLEAR ORIGIN IF NO SELECT GIVEN
493,J                           LAC                     ELSE MAKE SURE SELECT IS WITHIN RANGE
494,J                           CP      OTABLN
495,J                           JFC     GETCME
496,J                           PS      A,CUROSN        SAVE THE ORIGIN NUMBER
497,J                   ORIGI0  ADA                     INDEX INTO THE ORIGIN TABLE
498,J                           HL      OTABL
499,J                           INCP    HL,A
500,J                           DL      BC,HL           GET THE NEW ORIGIN VALUE
501,J                   ORIGIC  DPS     BC,CURORG       SET THE CURRENT ORIGIN TO THAT VALUE
502,J                           BC      CURORG          DISPLAY THE SELECTED ORIGIN
503,J                           DPS     BC,CURADR
504,J                           RET
505,J                   *
506,J                   ORIGIM
507,J                   *
508,J                   * MODIFY THE CURRENTLY SELECTED ORIGIN TABLE ENTRY
509,J                   *
510,J                           PL      A,CUROSN        GET THE CURRENT ORIGIN SELECT NUMBER
511,J                           JTZ     ORIGI0          SELECT PREVIOUS ENTRY IF NO DIGITS
512,J                           CP      OTABLN          MAKE SURE THE SELECT NUMBER IS WITHIN
513,J                           JFC     GETCME          RANGE IF WE'RE GOING TO UPDATE
514,J                           ADA                     ELSE INDEX INTO ORIGIN TABLE BY
515,J                           HL      OTABL           THE CURRENT SELECT NUMBER
516,J                           INCP    HL,A
517,J                           DS      BC,HL           STORE THE NEW VALUE THERE
518,J                           JMP     ORIGIC          AND SAVE AND DISPLAY IT
519,J                           XIF
520,J                   *
521,J                           IFS     STACKP
522,J                   *
523,J                   STACKR
524,J                   *
525,J                   * ROLL (POP) THE STACK THE NUMBER OF TIMES GIVEN IN THE C-REGISTER
526,J                   *
527,J                           LAC
528,J                           JFZ     STACKOV
529,J                           LA      1
530,J                   STACKOV CP      32-1
531,J                           JFC     GETCME          CAN'T POP MORE THAN STACKL/2-1 ENTRIES
532,J                           DE      DEBUG
533,J                           DPL     HL,OLDTOS
534,J                   *
535,J                   STACKOL DS      DE,HL
536,J                           NDL     -1-64
537,J                           INCP    HL,2            STACK POPS "UP"
538,J                           ORL     64
539,J                           SU      1
540,J                           JFZ     STACKOL
541,J                           JMP     STACKD0
                        *
```

```
542,J                                   STACKS
543,J                                   .
544,J                                   . STUFF (PUSH) A VALUE ONTO THE STACK
545,J                                   .
546,J                                           DPL     HL,OLDTOS
547,J                                           DECP    HL,2            MAKE B.O.S. NEW T.O.S.
548,J                                           ORL     64              STACK GROWS "DOWN"
549,J                                           DS      BC,HL
550,J                                   .
551,J                                   STACKD0 LC      0               CAUSE TOP STACK ENTRY TO BE DISPLAYED
552,J                                           DPS     HL,OLDTOS
553,J                                           XIF
554,J                                   *
555,J 174357                            STACKD
556,J                                   .
557,J                                   . DISPLAY A STACK LOCATION
558,J                                   .
559,J 174357  115 164 207                       DPL     HL,OLDTOS       (HL)=OLD TOP OF STACK VALUE
560,J 174362  302                               LAC                     (C) IS STACK NUMBER
561,J 174363  074 037                           CP      32-1
562,J 174365  100 125 370                       JFC     GETCME
563,J 174370  200                               ADA                     DOUBLE FOR INDEX
564,J 174371  176 044 277                       NDL     -1-64
565,J 174374  017                               INCP    HL,A
566,J 174375  176 064 100                       ORL     64
567,J 174400  115 166 200                       DPS     HL,CURADR
568,J 174403  007                               RETURN
569,J                                   *
570,J 174404                            BPSET
571,J                                   .
572,J                                   . SET A BREAK POINT AT THE CURRENT OR GIVEN ADDRESS
573,J                                   .
574,J 174404  110 012 371                       JFZ     BPSETG          USE CURRENT ADDRESS IF NO DIGITS GIVEN
575,J 174407  111 124 200                       DPL     BC,CURADR
576,J 174412  062 307                   BPSETG  LAM     BC              DON'T ALLOW BP IF BP ALREADY THERE
577,J 174414  074 052                           CP      OPCODEBP
578,J 174416  150 062 371                       JTZ     BPTSER
579,J 174421  066 211 056 327                   HL      BPTABL-BPTES+1  SEARCH THE TABLE FOR AN OPENING
580,J 174425  006 003                   BPSRCH  LA      BPTES           BUMP TO THE NEXT TABLE OPENING
581,J 174427  017                               INCP    HL,A
582,J 174430  176 074 251                       CPL     BPTABE          STOP IF PAST END OF TABLE
583,J 174433  100 062 371                       JFC     BPTSER          ERROR IF NO SLOTS AVAILABLE
584,J 174436  307                               LAM                     GET THE MSB OF THE ADDRESS
585,J 174437  074 377                           CP      -1              KEEP GOING IF NOT A FREE SLOT
586,J 174441  110 025 371                       JFZ     BPSRCH
587,J 174444  035                               DECP    HL              ELSE STORE THE ADDRESS IN THE TABLE
588,J 174445  111 027                           DS      BC,HL
589,J 174447  117 015                           INCP    HL,2            THEN SAVE WHAT WAS THERE IN MEMORY
590,J 174451  062 307                           LAM     BC
591,J 174453  370                               LMA
592,J 174454  006 052                           LA      OPCODEBP        THEN STORE THE BREAK POINT INSTRUCTION
593,J 174456  062 370                           LMA     BC
```

```
594,J 174460  153                     EX      CLICK           MAKE A LITTLE NOISE
595,J 174461  007                     RET                     DON'T CHANGE THE CURRENT ADDRESS
596,J                                 *
597,J 174462  151                     BPTSER  EX      BEEP            TELL THAT CAN NOT BREAKPOINT THERE
598,J 174463  104 253 370                     JMP     NOSADR          SHOW BP ALREADY THERE (OR NO TABLE ROOM)
599,J                                 *
600,J 174466                          BPCLR
601,J                                 .
602,J                                 . CLEAR ALL THE BREAK POINTS
603,J                                 .
604,J 174466  066 213 056 327                 HL      BPTABL          INIT THE MEMORY POINTER TO THE BP TABLE
605,J 174472  176 074 251             BPCLRL  CPL     BPTABE          STOP IF AT END OF TABLE
606,J 174475  150 060 373                     JTZ     BPTCLR
607,J 174500  047                             DL      DE,HL           ELSE DE => BREAK POINT LOCATION
608,J 174501  117 015                         INCP    HL,2            BUMP THE MEMORY POINTER PAST THE ADDRESS
609,J 174503  307                             LAM                     GET THE CONTENTS
610,J 174504  015                             INCP    HL              BUMP THE MEMORY POINTER TO NEXT ENTRY
611,J 174505  113 074 377                     CPD     -1              SKIP ENTRY IF NOT IN USE
612,J 174510  150 072 371                     JTZ     BPCLRL
613,J 174513  174 370                         LMA     DE              ELSE RESTORE THE CONTENTS
614,J 174515  104 072 371                     JMP     BPCLRL          AND MOVE ON TO THE NEXT ENTRY
615,J                                 *
616,J 174520                          REGMOD
617,J                                 .
618,J                                 . SWITCH REGISTER MODES (ALPHA/BETA)
619,J                                 .
620,J 174520  176 064 077                     ORL     SEPSWSV.AND.0177 (VALUE 077 OR AN ERROR)
621,J 174523  115 166 200                     DPS     HL,CURADR       DISPLAY THE MODE
622,J 174526  006 200                         LA      SWALBT
623,J 174530  104 311 372                     JMP     SREGMOD         GO TO SET THE REGISTER MODE
624,J                                 .
625,J                                         IFNE    ROMTYPE,3
626,J                                 *
627,J 174533                          BASSET
628,J                                 .
629,J                                 . SET OR DISPLAY THE BASE REGISTER
630,J                                 .
631,J 174533  066 253 056 357                 HL      SEBRLS          HL => CURRENT BASE STORAGE (BY FIRMWARE)
632,J 174537  150 154 371                     JTZ     BASED           JUST DISPLAY IF NOTHING ENTERED
633,J 174542  074 120                         CP      'P'             SUBTRACT 0100000 IF NOT SHIFTED P
634,J 174544  302                             LAC                     ELSE JUST USE VALUE IN C-REG
635,J 174545  150 153 371                     JTZ     BASSES
636,J                                 .
637,J 174550  301                             LAB                     ELSE BIAS THE VALUE GIVEN SO IT WILL
638,J 174551  024 200                         SU      0100000>8       CORRESPOND TO VALUES BASED TO
639,J                                 .
640,J 174553  072                     BASSES  BRL                     THEN STORE AND UPDATE THE BASE REGISTER
641,J 174554  115 166 200             BASED   DPS     HL,CURADR       CAUSE THE BASE REGISTER STORAGE
642,J 174557  007                             RET                     TO BE DISPLAYED
```

```
643,J
644,J 174560
645,J
646,J
647,J
648,J
649,J 174560  106 112 370
650,J 174563  115 164 200
651,J 174566  307
652,J 174567  074 020
653,J 174571  100 125 370
654,J 174574  320
655,J 174575  015
656,J 174576  077
657,J 174577  007
```

```
*
STLOAD
*
* LOAD THE SECTOR TABLE - LOAD (N) ENTRIES WHERE (N) IS THE BYTE POINTED TO
*       BY THE CURRENT ADDRESS AND THE ENTRIES FOLLOW THAT LOCATION.
*
          CALL      C12345          ENTRY MUST BE '123450'
          DPL       HL,CURADR       IF '123450' THEN LOAD FROM CURADR
          LAM                       A = NUMBER OF ENTRIES
          CP        16              ERROR IF NUMBER OF ENTRIES >15
          JFC       GETCME
                                    SET C TO NUMBER OF ENTRIES
          LCA
          INCP      HL              BUMP HL TO START OF LIST
          STL                       I SURE HOPE EVERYTHING WAS SET UP RIGHT!
          RET
```

- DEBUG COMMAND EXECUTION ROUTINES -

```
                                        +
                                ADRDEV
                                        .
                                . ADDRESS AN I/O DEVICE AND DISPLAY ITS CURRENT STATUS
                                        .
                                . NOTE: MINOR DIFFERENCE FROM 5500 VERSION FOR 'F' AND 'G' MODES
                                .       THE FIRST INPUT BY THIS CODE GIVES ACTUAL DATA NOT STATUS.
                                .       DIFFERENCE FOUND BY GADWA'S GROUP
                                .
 174600  110 210 371                    JFZ     ADRDE1          ADDRESS IS GIVEN
 174603  066 252 056 357 327           MLC     *SEXADR         USE LAST ADDRESSED DEV IF NOT GIVEN
 174610                         ADRDE1                          ADDRESS THE DEVICE
 174610  062 121                        EXC     ADR             (AUTOMATICALLY SETS LAST ADRESS GIVEN)
 174612  074 141                        CP      ' '             DO INPUT IF 'A' COMMAND
 174614  150 223 371                    JTZ     GETSTA
                                .
 174617  125                            EX      DATA            ELSE PUT DEVICE IN DATA MODE
 174620  074 147                        CP      ' '             JUST EXIT IF 'G' COMMAND
 174622  053                            RTZ
                                .
 174623  101                    GETSTA  IN                      GET THE STATUS OF THAT DEVICE
 174624  066 203 056 327 370            MSA     *CURSTA
 174631  115 166 200                    DPS     HL,CURADR       CAUSE THE STATUS TO BE DISPLAYED
 174634  007                            RET
                                *
                                . ADDRESS THE CURRENTLY ADDRESSED I/O DEVICE AND OUTPUT THE VALUE GIVEN
                                .       OR THE LAST VALUE GIVEN FOR OUTPUT DOING AN I/O STROBE BASED
                                .       UPON THE COMMAND LETTER GIVEN.
                                .
 174635  106 300 371            OUTXST  CALL    OUTSETUP        ' ' - EX STATUS
 174640  123                            EX      STATUS
 174641  007                            RET                     OFF TO "GETSTA"
                                *
 174642  106 300 371            OUTXDA  CALL    OUTSETUP        ' ' - EX DATA
 174645  125                            EX      DATA
 174646  007                            RET                     OFF TO "GETSTA"
                                *
 174647  106 300 371            OUTXWT  CALL    OUTSETUP        'W' - EX WRITE
 174652  127                            EX      WRITE
 174653  007                            RET                     OFF TO "GETSTA"
```

- DEBUG COMMAND EXECUTION ROUTINES -

```
698,J
699,J 174654  106 300 371          *
700,J 174657  131                  OUTXC1    CALL      OUTSETUP          'X' - EX COM1
701,J 174660  007                              EX        COM1
702,J                                          RET                         OFF TO "GETSTA"
703,J 174661  106 300 371          *
704,J 174664  133                  OUTXC2    CALL      OUTSETUP          'Y' - EX COM2
705,J 174665  007                              EX        COM2
706,J                                          RET                         OFF TO "GETSTA"
707,J 174666  106 300 371          *
708,J 174671  135                  OUTXC3    CALL      OUTSETUP          'Z' - EX COM3
709,J 174672  007                              EX        COM3
710,J                                          RET                         OFF TO "GETSTA"
711,J 174673  106 300 371          *
712,J 174676  137                  OUTXC4    CALL      OUTSETUP          'V' - EX COM4
713,J 174677  007                              EX        COM4
714,J                                          RET                         OFF TO "GETSTA"
715,J 174700                       *
716,J                              OUTSETUP
717,J                              *
718,J                              * PUSH "GETSTA" ON THE STACK BELOW THE RETURN ADDRESS AND PUT THE RIGHT
719,J                              *       VALUE IN THE A-REGISTER FOR THE OUTPUT INSTRUCTION.
720,J 174700  060                  *
721,J 174701  051 223 371                      POP       HL
722,J 174704  070                              PUSH      GETSTA
723,J 174705  105 202                          PUSH      HL
724,J 174707  053                              PL        A,CUROUT
725,J                                          RTZ
726,J 174710  302                              LAC
727,J 174711  126 202                          PS        C,CUROUT
728,J 174713  007                              RET
729.J                                          XIF
```

```
730,J
731,J 174714                    +
732,J                           EXEUSR
733,J                           .
734,J                           . CONTINUE EXECUTION FORCING USER MODE AND INTERRUPTS ENABLED
735,J 174714  006 005           .
736,J 174716  104 330 371               LA      SWUSER+SWINTE   USER MODE,INT. ENABLED
737,J                                   JUMP    EXECEM
738,J 174721                    *
739,J                           EXECUI
740,J                           .
741,J                           . CONTINUE EXECUTION FORCING INTERRUPTS ENABLED
742,J 174721  006 001           .
743,J 174723  104 330 371               LA      SWINTE          INTERRUPTS ENABLED
744,J                                   JUMP    EXECEM
745,J 174726                    *
746,J                           EXECUT
747,J                           .
748,J                           . CONTINUE EXECUTION WITH NO CHANGE IN MODE OR INTERRUPT FLAGS
749,J 174726  006 000           .
750,J                                   LA      0               NO CHANGE (CAN'T BE XRA OR JTZ FAILS)
751,J 174730  115 164 207       .
752,J 174733  150 340 371       EXECEM  DPL     HL,OLDTOS
753,J                                   JTZ     EXECDM          CATCH NO COMMAND VALUE GIVEN
754,J 174736  111 027           .
755,J 174740  176 044 300               DS      BC,HL           ELSE SET NEW RETURN ADDRESS
756,J 174743  035               EXECDM  NDL     0300
757,J 174744  267                       DECP    HL              POINT TO PSW
758,J 174745  370                       ORM                     SET MODE AS DESIRED
759,J 174746  106 344 372               LMA     HL
760,J                                   CALL    RESTRLOC        CORRECT REGISTER LOCATIONS
761,J 174751  115 164 207       .
762,J 174754  176 065                   DPL     HL,OLDTOS       RESTORE USER SYSTEM SAVE AREA
763,J 174756  062 030                   SYSMOV  HL
764,J                                   SYSRET                  LOAD STATE, REGS, AND RETURN
765,J 174760                    *
766,J                           CALL
767,J                           .
768,J                           . WARNING: 'CALL' MUST NOT BE IN USER MODE OR THE RETURN WILL GENERATE
769,J                           .           AN ERROR BECAUSE MACRO-ROM IS ACCESS PROTECTED
770,J 174760  176 064 077       .
771,J 174763  347                       ORL     SEPSWSV.AND.0177 POINT TO THE SAVED PSW
772,J 174764  174 044 373               LEM                        (MAY NOT USE A-REG)
773,J 174767  374                       NDE     -1-SWUSER       AND TURN OFF THE USER MODE BIT
774,J 174770  113 074 106               LME
775,J                                   CPD     NECP            RESTORE THE CC FOR FOLLOWING CODE
776,J 174773                    .
777,J                           JUMP
778,J                           .
779,J                           . PERFORM A CALL ("DEBUG" IS PUT ON THE STACK AS THE RETURN ADDRESS) OR A
780,J                           . JUMP TO A GIVEN MEMORY LOCATION OR THE CURRENTLY ADDRESSED MEMORY LOCATION
781,J 174773  150 001 372       .
                                        JTZ     CALLKA
```

```
782 J
783 J 174776  111 126 200          *
                          DPS     BC,CURADR       SET ADDRESS TO CALL (JUMP TO)
784 J 175001  022 070     CALLKA  PUSH    XA      (SAVE A = INPUT DEBUG OPERATION)
785 J 175003  106 344 372         CALL    RESTRLOC CORRECT REGISTER LOCATIONS
786 J 175006  022 060             POP     XA
787 J 175010  115 164 207         DPL     HL,OLDTOS
788 J 175013  111 124 200         DPL     BC,CURADR       GET CALL (JUMP) ADDRESS
789 J 175016  046 127 036 367     DE      DEBUG           GET RETURN ADDRESS (IF CALL)
790 J 175022  176 065             SYSMOV  HL              RESTORE USER SYSTEM SAVE AREA
791 J 175024  074 152             CP      ' '             JUMP OPERATION?
792 J 175026  150 033 372         JTZ     JUMPIT
793 J 175031  174 070             PUSH    DE              NO, 'CALL' SO PUSH RETURN ADDRESS
794 J 175033  062 070     JUMPIT  PUSH    BC
795 J 175035  062 030             SYSRET                  LOAD STATE, REGS, AND RETURN
```

- DEBUG COMMAND EXECUTION ROUTINES -

```
796,J
797,J 175037
798,J
799,J
800,J
801,J 175037  066 117
802,J 175041  150 045 372
803,J 175044  362
804,J 175045  070
805,J 175046  106 004 374
806,J 175051  060
807,J 175052  316
808,J 175053  076 357
809,J 175055  106 053 374
810,J
811,J 175060  106 125 373
812,J 175063  150 060 372
813,J 175066  104 004 374
```

```
+
TSTDSP
.
. TEST THE DISPLAY SCREEN BY FILLING IT FULL OF CHARACTERS
.
          LL      'O'             STANDARD CHARACTER
          JTZ     TDSTD
          LLC                     WILL NOT BE USED, VALUE ENTERED WILL BE
TDSTD     PUSH    HL              SAVE IT
          CALL    DSPINIT         INIT THE DISPLAY AND ITS POINTERS
          POP     HL
          LBL                     PUT IN B THE CODE TO BE DISPLAYED
          LX      SEDOPTS>8
          CALL    DOSF68LP        CHANGE THE DISPLAY TO ALL 'B'
.
TDWAIT    CALL    DOSF61          WAIT FOR ANY KEYIN
          JTZ     TDWAIT
          JMP     DSPINIT         RE-INIT THE DISPLAY AND RETURN TO GET KEY
```

```
814,J
815,J 175071                        +
816,J                               IDENT
817,J                               .
818,J                               . DISPLAY THE MICRO AND MACRO ROM VERSION INFORMATION
819,J 175071  066 130 056 372       .
820,J 175075  106 341 374                   HL      IDENTM          DISPLAY THE MACRO ROM VERSION INFORMATION
821,J                                       CALL    DISPLAY         HOPING B=REG IS ZERO
822,J 175100  111 010               .
823,J 175102  321                           INFO                    C = MICRO-ROM VERSION
824,J 175103  036 017                       LCB
825,J 175105  106 115 372                   LD      IDMIC           DISPLAY IT IN OCTAL
826,J                                       CALL    IDENT0          (SAVING A LITTLE SPACE)
827,J 175110  111 010               .
828,J 175112  320                           INFO                    C = PROCESSOR TYPE
829,J 175113  036 013                       LCA
830,J 175115                        IDENT0
831,J 175115  016 000                       LB      0               DISPLAY IT IN OCTAL
832,J 175117  066 003                       LL      3
833,J 175121  106 160 372                   CALL    DSPOCT
834,J 175124  006 072                       LA      ':'
835,J 175126  370                           LMA
836,J 175127  007                           RET
837,J                               *
838,J 175130                        IDENTM
839,J                               .
840,J                               . ROM VERSION IDENTIFICATION MESSAGE
841,J                               .
842,J 175130  224                           DC      $HD
843,J 175131  040 115 072           IDENTS  DC      ' M:'
844,J 175134  060                           DC      $MACROM/64.AND.7+'0'
845,J 175135  061                           DC      $MACROM/8.AND.7+'0'
846,J 175136  061                           DC      $MACROM.AND.7+'0'
847,J 175137  040                           DC      $MACVER         PUT REVISION LETTERS IN MESSAGE
848,J 175140  120                           DC      'P'
849,J 175141  040 040 040 040       DSPBLNK DC      '    '          (INVISIBLE COLON APPEARS WHEN NEEDED)
850,J 000013                        IDPROC  EQU     $-IDENTS-1
851,J 175145  040 040 040 040               DC      '    '
852,J 000017                        IDMIC   EQU     $-IDENTS-1
853,J 175151  040 040 003                   DC      '  ',$ES
854,J 000012                        DSPBLEN EQU     $-DSPBLNK-1
```

- DEBUG SUBROUTINES -

```
857,J                                   *
858,J                                   . DISPLAY OCTAL VALUE RIGHT TO LEFT
859,J                                   .
860,J                                   .       ENTRY: BC = VALUE TO BE DISPLAYED
861,J                                   .              E  = LINE FOR THE DISPLAY (=12 THRU 11)
862,J                                   .              L  = NUMBER OF CHARACTERS (IF ENTERED BELOW DSPOC6)
863,J                                   .              D  = POSITION OF RIGHT-HAND CHARACTER
864,J                                   .                   (IF ENTERED BELOW DSPOCR)
865,J                                   .
866,J                                   .       EXITS: BC - SCRATCHED
867,J                                   .              D  = POSITION TO LEFT OF MOST SIG. DIGIT (BLANKED)
868,J                                   .              E  = ENTRY VALUE
869,J                                   .              HL - SCRATCHED
870,J                                   .
871,J 175154  066 006                   DSPOC6  LL      6               DISPLAY 6 DIGITS
872,J 175156  036 117                   DSPOCR  LD      79              DISPLAY AT RIGHT OF SCREEN
873,J                                   .
874,J 175160  306                       DSPOCT  LAL                     GET THE COUNT IN THE A-REGISTER
875,J 175161  022 070                           PUSH    XA
876,J 175163  106 312 374                       CALL    DSPCBL          POSITION HL INTO DISPLAY BUFFER
877,J 175166  022 060                           POP     XA
878,J                                   .
879,J 175170  022 070                   DSPOCL  PUSH    XA              SAVE THE COUNT ON THE STACK
880,J 175172  302                               LAC                     GET THE LEAST SIGNIFICANT DIGIT
881,J 175173  044 007                           ND      7               ISOLATE THE OCTAL BITS
882,J 175175  004 060                           AD      '0'             CONVERT TO ASCII DIGIT
883,J 175177  370                               LMA                     DISPLAY THE ASCII DIGIT
884,J 175200  035                               DECP    HL
885,J 175201  106 222 372                       CALL    SBCRL3          SHIFT THE 16-BIT VALUE RIGHT 3 BITS
886,J 175204  113 024 001                       SUD     1               DECREMENT THE HORIZONTAL POSITION
887,J 175207  022 060                           POP     XA              DECREMENT THE COUNT
888,J 175211  024 001                           SU      1
889,J 175213  110 170 372                       JFZ     DSPOCL          LOOP IF MORE DIGITS TO GO
890,J                                   .
891,J 175216  016 040                           LB      ' '             THEN BLANK THE POSITION TO THE LEFT
892,J 175220  371                               LMB
893,J 175221  007                               RET
894,J                                   *
895,J 175222                            SBCRL3
896,J                                   .
897,J                                   . SHIFT BC RIGHT LOGICALLY 3 BITS
898,J                                   .
899,J 175222  111 032                           SREB
900,J 175224  062 032                           SREC
901,J 175226  111 032                           SREB
902,J 175230  062 032                           SREC
903,J 175232  111 032                           SREB
904,J 175234  062 032                           SREC
905,J 175236  111 044 037                       NDB     037
906,J 175241  007                               RET
```

```
907 J
908 J 175242                        +
909 J                               SSTATE
910 J                               *
911 J                               * SAVE THE STATE OF THE MACHINE AND SWAP AROUND THE REGISTERS
912 J                               *       SO THE DEBUG DISPLAY WILL SHOW PAIR VALUES AS WELL.
913 J 175242  062 020               *
914 J 175244  066 200 056 327               SYSSAV                  SAVE CURRENT MODE OF REGISTERS
915 J 175250  174 060               SDBGWS  HL      SDBGWS          POINT TO NEW STACK AREA
916 J 175252  176 065                       POP     DE                 (RETURN ADDRESS)
917 J 175254  174 070                       SYSMOV  HL              MOVE SYSTEM SAVE AREA
918 J 175256  076 327                       PUSH    DE                 (TO NEW STACK)
919 J 175260  117 275                       LX      SDBGWS>8
920 J 175262  053                           CPHX                    IS STACK THE DEBUG STACK ?
921 J                                       RTZ                     YES, DONE.
922 J 175263  115 166 207           *
923 J 175266  176 044 300                   DPS     HL,OLDTOS
924 J 175271  117 035                       NDL     0300
925 J 175273  106 363 372                   DECP    HL,2            POINT TO ALPHA (F)
926 J 175276  035                           CALL    SWITZAF
927 J 175277  106 325 372                   DECP    HL              POINT TO ALPHA (AX)
928 J 175302  035                           CALL    RSWAP
929 J 175303  106 325 372                   DECP    HL              POINT TO BETA AX
930 J                                       CALL    RSWAP              (RETURNS A=0)
931 J 175306  176 064 077           *
932 J 175311  257                   SREGMOD XRM     HL              GET PSW (SETTING NEW MODE IF REGMOD)
933 J 175312  370                           LMA     HL
934 J 175313  120 321 372                   JFS     SSTAT1
935 J                               *
936 J 175316  006 011                       LA      9               WANT BETA REGS
937 J 175320  037                           DECP    HL,A
938 J 175321  115 166 211           SSTAT1  DPS     HL,OLDREGS
939 J 175324  007                           RET
940 J                               *
941 J 175325                        RSWAP
942 J                               *
943 J                               * SWAP THE LOCATIONS OF LSB AND MSB REGISTERS IN PAIRS IN MEMORY
944 J                               *
945 J 175325  006 004                       LA      4
946 J                               *
947 J 175327  047                   RSWL    DL      DE,HL
948 J 175330  324                           LCE
949 J 175331  343                           LED
950 J 175332  332                           LDC
951 J 175333  027                           DS      DE,HL
952 J 175334  117 035                       DECP    HL,2
953 J 175336  024 001                       SU      1
954 J 175340  110 327 372                   JFZ     RSWL
955 J                               *
956 J 175343  007                           RETURN
```

```
957 J
958 J 175344                        *
959 J                               RESTRLOC
960 J                               .
961 J                               .      HL POINTS INTO REG-SAVE AREA (OR PSW) ON ENTRY
962 J                               .
963 J                               . RESTORE THE LOCATIONS OF THE REGISTERS TO THEIR NORMAL
964 J                               .       PLACES IN THE STATE STORAGE AREA IN MEMORY.
965 J 175344  176 064 077           .
966 J 175347  117 035                       ORL     SEPSWSV.AND.0177 (VALUE SHOULD BE 077 OR AN ERROR)
967 J 175351  106 325 372                   DECP    HL,2             POINT TO ALPHA XA PAIR SWAPPED
968 J 175354  035                           CALL    RSWAP
969 J 175355  106 325 372                   DECP    HL
970 J 175360  006 022                       CALL    RSWAP
971 J 175362  017                           LA      18
972 J                                       INCP    HL,A
973 J                               .                                FALLS THROUGH INTO SWITZAF
974 J 175363                        *
975 J                               SWITZAF
976 J                               .
977 J                               . SWITCH THE A REGISTER AND FLAGS IN MEMORY FOR BOTH MODES
978 J                               . ON ENTRY, MUST POINT TO ALPHA FLAGS (TO BECOME A-REG AS IT WAS)
979 J 175363  006 011               .
980 J 175365  337                           LA      9
981 J 175366  037                           LDM                      D = ALPHA FLAGS
982 J 175367  111 047                       DECP    HL,A
983 J 175371  037                           DL      BC,HL            BC = ALPHA A-REG AND BETA FLAGS
984 J 175372  015                           DECP    HL,A
985 J 175373  347                           INCP    HL
986 J 175374  372                           LEM                      E = BETA A-REG
987 J 175375  035                           LMC                      STORE BETA FLAGS THERE
988 J 175376  017                           DECP    HL
989 J 175377  027                           INCP    HL,A
990 J 175400  017                           DS      DE,HL            STORE BETA A-REG AND ALPHA FLAGS
991 J 175401  371                           INCP    HL,A
992 J 175402  007                           LMB                      STORE ALPHA A-REG
                                          RET
```

```
 993 J
 994 J 175403                          +
                                       MTSETUP
 995 J                                 .
 996 J                                 . INITIALIZE: THE SECTOR TABLE, THE BASE REGISTER TO ZERO, AND
 997 J                                 .             THE INTERRUPT VECTOR TO THE INTERNAL TRAP MESSAGES.
 998 J                                 .
 999 J 175403  174 060                          POP     DE              ** INSURE CALLER GETS RETURNED TO
1000 J 175405  066 227 056 363         MTSETX   HL      UMST            INIT THE SECTOR TABLE FOR USER
1001 J 175411  026 015                          LC      UMSTL              MEMORY SECTORS
1002 J 175413  077                              STL
1003 J 175414  066 225 056 363                  HL      SMST            INIT THE SECTOR TABLE FOR SYSTEM
1004 J 175420  026 322                          LC      (SEDSPBF>8.AND.0360)+SMSTL    MEMORY SECTORS
1005 J 175422  062 077                          STLOC
1006 J 175424  250                              XRA                     ZERO THE BASE REGISTER
1007 J 175425  072                              BRL
1008 J 175426  174 070                          PUSH    DE              ** IF THE STACK GETS MOVED (1800MOVI)
1009 J                                 .
1010 J 175430  066 270 056 360                  HL      VECTI           INIT THE INTERRUPT VECTOR
1011 J 175434  046 000 036 357                  DE      SVMEMP
1012 J 175440  026 124                          LC      VECTIL          MOVE THE WHOLE TABLE
1013 J                                 .        XRA                     NO BIAS ON THE BLOCK TRANSFER
1014 J 175442  310                              LBA                     DON'T CHECK FOR ANY CHARACTERS
1015 J 175443  021                              BT                      TRANSFER THE ENTRIES TO MEMORY
1016 J 175444  007                              RET
1017 J                                 *
1018 J 175445                          SETUP
1019 J                                 .
1020 J                                 . INITIALIZE: THE SECTOR TABLE, THE BASE REGISTER TO ZERO, THE INTERRUPT
1021 J                                 .             VECTOR TO THE INTERNAL TRAP MESSAGES, THE STACK AREA TO
1022 J                                 .             THE "NORMAL" SYSTEM RAM AREA, THE ORIGIN TABLE TO
1023 J                                 .             ALL ZEROS, AND THE BREAK POINT TABLE TO ALL FREE ENTRIES.
1024 J                                 . WARNING:    IF SECTOR TABLE MOVES, ROUTINE MTSETUP STOPS RETURN TO CALLER.
1025 J                                 .
1026 J                                 .
1027 J 175445  106 003 373                      CALL    MTSETUP
1028 J 175450  060                              POP     HL              SAVE THE RETURN ADDRESS IN HL
1029 J 175451  046 300 036 357                  DE      SESTACK         MOVE THE STACK AREA TO THE
1030 J 175455  174 065                          SYSMOV  DE                 "NORMAL" SYSTEM RAM AREA
1031 J 175457  070                              PUSH    HL              RESTORE THE RETURN ADDRESS
1032 J                                 *
1033 J                                          IFS     ORIGINI
1034 J                                 ORGCLR
1035 J                                 .
1036 J                                 . CLEAR ALL ORIGIN TABLE ENTRIES
1037 J                                 .
1038 J                                          HL      CURORG
1039 J                                          XRA
1040 J                                 OTCLOP   LMA     HL
1041 J                                          INCP    HL
1042 J                                          CPL     OPTABE
1043 J                                          JFZ     OTCLOP
1044 J                                          XIF
```

```
                                        - DEBUG SUBROUTINES -

 1045 J
 1046 J 175460                          *
 1047 J                                 BPTCLR
 1048 J                                 *
 1049 J                                 * FREE ALL BREAK POINT ENTRIES
 1050 J 175460  066 213 056 327         *         HL      BPTABL
 1051 J 175464  006 377                 *         LA      -1
 1052 J 175466  370                     BPTCLL    LMA
 1053 J 175467  015                     *         INCP    HL
 1054 J 175470  176 074 251             *         CPL     BPTABE
 1055 J 175473  110 066 373             *         JFZ     BPTCLL
 1056 J 175476  007                     *         RET
  177 J                                 *
  178 J                                 *         INC     1800DSPR
```

```
3,K
4,K
5,K
6,K
7,K
8,K
9,K
10,K
11,K
12,K
13,K
14,K
15,K
16,K
17,K
18,K
19,K
20,K
21,K
22,K
23,K
24,K
25,K
26,K
27,K
28,K
29,K
30,K
31,K
32,K
33,K
34,K
35,K
36,K
37,K
38,K
39,K
40,K
41,K
42,K
43,K
44,K
45,K
46,K
47,K
48,K
49,K
50,K
51,K
52,K
53,K
54,K
```

```
.
. 1800 DISPLAY INTERFACE ROUTINES          JUNE 8, 1978    11:53    HSP/HJS
.
. GENERAL NOTES ON THE FOLLOWING ROUTINES:
.
.       THE FLASHING OF THE CURSOR IS CONTROLLED BY THE "CURONOFF" AND
.       "CURINBUF" BITS IN THE "SECFLAGS" BYTE.  THE POSITION OF THE
.       CURSOR IS DETERMINED BY THE TWO BYTES IN "SECPOS" (VERTICAL)
.       AND "SECPOS"+1 (HORIZONTAL).
.
.       WHEN THE "CURONOFF" BIT IS A ZERO, THE CURSOR IS NOT FLASHED.
.       OTHERWISE, WHEN THE "SELFREQ" BIT IN THE "CURINBUF" BIT POSITION
.       IS DIFFERENT FROM THE "CURINBUF" BIT, THE CURSOR IS EITHER:
.       1) TURNED ON IF THE "SELFREQ" BIT IS NOW A ONE OR 2) TURNED OFF
.       IF THE "SELFREQ" BIT IS NOW A ZERO.  WHEN THE CURSOR IS TURNED ON,
.       THE CHARACTER AT THE CURSOR POSITION IS FIRST SAVED INTO
.       "SECHIDE" AND THE CHARACTER AT "SECCHAR" IS STORED WHERE THE
.       CURSOR POSITION DIRECTS, PUTTING IT "IN THE BUFFER".
.
.       WHEN THE CURSOR IS TURNED OFF, IF THE CHARACTER AT "SECCHAR"
.       IS NOT THE SAME AS THAT WHERE THE CURSOR POSITION DIRECTS
.       IT IS ASSUMED THAT THE CURSOR WAS OVERWRITTEN BY SOME ROUTINE
.       AND THE CHARACTER AT "SECHIDE" IS NOT RESTORED AS IS NORMALLY
.       DONE.
.
.       WHEN THE "CURINBUF" BIT IS A ONE, THE CHARACTER IN THE CURSOR
.       POSITION IS THE CURSOR CHARACTER.  OTHERWISE, IT IS THE
.       CHARACTER THAT IS TO NORMALLY BE DISPLAYED IN THAT POSITION.
.       THE EXCEPTION TO THIS RULE IS WHEN THE CURSOR POSITION IS NOT
.       WITHIN THE CONFINES OF THE SCREEN (-12 TO +11 VERTICALLY AND
.       0 TO 79 HORIZONTALLY), THE CURSOR IS NOT FLASHED.
.
.       BEFORE MANY OF THE FOLLOWING ROUTINES WRITE TO THE SCREEN, THE
.       CURSOR IS SUSPENDED.  TO SUSPEND THE CURSOR, NO ACTION IS
.       REQURIED IF THE "CURINBUF" BIT IS A ZERO BECAUSE THE CURSOR CHARACTER
.       IS NOT ON THE SCREEN.  OTHERWISE, THE "CUROFF" BIT IS RESET TO
.       ZERO AND THE CHARACTER AT "SECHIDE" IS STORED WHERE THE CURSOR
.       POSITION DIRECTS IF THE CURSOR CHARACTER "SECCHAR" WAS THERE.
.
. WARNING: THE FOLLOWING ROUTINES THAT CALL 'BLINK' OR 'CURSES' ARE NOT
.       RE-ENTRANT, I.E. THEY CAN NOT BE USED IN A MULTI-TASKING ENVIRONMENT
.       UNLESS ALL OF THEM ARE IN THE SAME TASK (ALL FOREGROUND OR ALL
.       BACKGROUND). IF THIS IS NOT DONE, THE CHARACTER AT THE CURSOR
.       POSITION CAN GET PERMANENTLY CHANGED TO THE CURSOR CHARACTER.
.       THIS IS NOT DESIRABLE. (DISPLAY IS INCLUDED IN THIS RESTRICTION)
.
. ALSO:   THE ROUTINES THAT CALL 'CURSESRS' STORE REGISTERS IN A FIXED STORAGE
.       AREA. IF THESE ROUTINES ARE RE-ENTERED, THE REGISTERS AS STORED FROM
.       THE FIRST CALL WILL BE LOST.  THE BASIC DEBUG ROUTINES (IN THE ROM)
.       HAVE BEEN MODIFIED SO THAT THIS WILL NOT NORMALLY HAPPEN DUE TO A
.       DEBUG ENTRY.  BUT, NOT ALL ROUTINES COULD BE MODIFIED SO,
.       KNOWLEDGE & CARE MUST BE TAKEN IN USING THE DEBUGGER.
```

- DOS FUNCTION 6 ROUTINES -

```
                                *
                                . DOS FUNCTION 6
                                .
                                .    SUBFUNCTION:  0 - RETURN KBD/DSP KEY STATUS
                                .                  1 - CHECK FOR CHARACTER READY
                                .                  2 - GET A BYTE FROM THE KEYBOARD
                                .                  3 - WRITE THE BYTE IN (B) TO THE SCREEN
                                .                  4 - RETURN HOME-UP POSITION IN (DE)
                                .                  5 - RETURN HOME-DOWN POSITION IN (DE)
                                .                  6 - TURN CURSOR ON
                                .                  7 - ROLL UP SCREEN
                                .                  8 - ERASE FROM CURSOR TO END OF FRAME
                                .                  9 - ERASE FROM CURSOR TO END OF LINE
                                .                 10 - ROLL DOWN SCREEN
                                .                 11 - TURN CURSOR OFF
                                .                 12 - RETURN FUNCTION KEY & KEYBOARD STATUS
                                .              DSPCBL - CONVERT CURSOR TO BUFFER POSITION
                                .              CURSES - SUSPEND THE CURSOR (REMOVE IT FROM BUFFER)
                                .               BLINK - BLINK THE CURSOR
                                .             BLINKDE - MOVE CURSOR TO DE AND BLINK IT THERE
                                .                 62N - DOS FUNC 6 SUB 2 WITHOUT "DE" AND CURSOR BLINK
                                *
175477                          DOSF60
                                .
                                . DOS FUNCTION:  6   SUBFUNCTION:  0
                                .
                                . - RETURN THE STATES OF THE "KEYBOARD" AND "DISPLAY" KEYS -
                                .
                                .         ENTRY: N/A
                                .
                                .         EXITS: SIGN TRUE IF "KEYBOARD" KEY IS DOWN
                                .                PARITY TRUE IF "DISPLAY" KEY IS DOWN
                                .                B & X ONLY REGISTERS SAVED
                                .                2 EXTRA STACK LEVELS USED (MAX)
                                .
175477  106 244 373                     CALL    BLINK           BLINK THE CURSOR WHERE IT WAS
175502                          DOSF60N                             (SPECIAL - MACRO-ROM NON BLINK ENTRY)
175502  066 151 056 357 307             MLA     *SEKBS1         GET KEYBOARD STATUS 1 INTO (A)
175507  044 003                         ND      SEDSPKY.OR.SEKBDKY
175511  004 121                         AD      CCTAB60         INDEX INTO THE TABLE OF BYTES
175513  360                             LLA                         WHICH WILL GIVE THE CORRECT
175514  056 373                         LH      CCTAB60>8           CONDITION CODE WHEN ADDED
175516  307                             LAM                         TO THEMSELVES.
175517  200                             ADA                     SET THE CONDITION CODE
175520  007                             RET
```

- DOS FUNCTION 6 ROUTINES -

```
102,K                                   *
103,K                                   . THE CONDITION CODE GENERATION TABLE IS PAGE SENSITIVE
104,K                                   .
105,K                                           IFGT    ($,AND,255),252
106,K                                           RPT     -$,AND,255
107,K                                           DC      000             PAD TO END OF PAGE
108,K                                           XIF
109,K                                   .
110,K 175521  003                       CCTAB60 DC      0003            FALSE SIGN, PARITY  & NO KEYS DOWN
111,K 175522  002                               DC      0002            TRUE PARITY     & DISPLAY KEY DOWN
112,K 175523  101                               DC      0101            TRUE SIGN       & KEYBOARD KEY DOWN
113,K 175524  100                               DC      0100            TRUE SIGN, PARITY & BOTH KEYS DOWN
114,K                                   *
115,K 175525                            DOSF61
116,K                                   .
117,K                                   . DOS FUNCTION:  6   SUBFUNCTION:  1
118,K                                   .
119,K                                   . - CHECK THE KEYBOARD FOR CHARACTER READY -
120,K                                   .
121,K                                   .       ENTRY: N/A
122,K                                   .
123,K                                   .       EXITS: FALSE ZERO IF KEYBOARD HAS A CHARACTER
124,K                                   .              B & X ONLY REGISTERS SAVED
125,K                                   .              2 EXTRA STACK LEVELS USED (MAX)
126,K                                   .
127,K 175525  106 244 373                       CALL    BLINK           BLINK THE CURSOR WHERE IT WAS
128,K 175530  066 151 056 357 307               MLA     *SEKBS1         GET KEYBOARD STATUS 1 INTO (A)
129,K 175535  044 004                           ND      SEKBRDY         SET THE ZERO CONDITION FALSE IF
130,K 175537  007                               RET                        THE KEYBOARD HAS A CHARACTER
131,K                                   *
132,K 175540                            DOSF62
133,K                                   .
134,K                                   . DOS FUNCTION:  6   SUBFUNCTION:  2
135,K                                   .
136,K                                   . - GET A TRANSLATED CHARACTER FROM THE KEYBOARD -
137,K                                   .
138,K                                   .       ENTRY: D = HORIZONTAL CURSOR POSITION
139,K                                   .              E = VERTICAL CURSOR POSITION
140,K                                   .              D,E NOT USED IN DOSF62N ($KEYCHAR) ENTRY
141,K                                   .
142,K                                   .       EXITS: ZERO TRUE IF NO CHARACTER READY
143,K                                   .              A  = CHARACTER, AFTER TRANSLATION
144,K                                   .              BC = SCRATCHED (NOT ON DOSF62N = $KEYCHAR = ENTRY)
145,K                                   .              HL = SCRATCHED
146,K                                   .       NOTE:  L  = LOW 7 BITS OF UN-TRANSLATED KEYCODE (IF NEEDED)
147,K                                   .              4 EXTRA STACK LEVELS USED (MAX)
148,K                                   .             (0 EXTRA STACK LEVELS USED - IN DOSF62N ($KEYCHAR) ENTRY)
149,K                                   .
150,K 175540  106 221 373                       CALL    BLINKDE         BLINK THE CURSOR AT (DE)
151,K 175543                            DOSF62N
152,K 175543  066 151 056 357 307               MLA     *SEKBS1         GET KEYBOARD STATUS 1 INTO (A)
153,K 175550  044 004                           ND      SEKBRDY         EXIT IF KEYBOARD NOT READY
```

- DOS FUNCTION 6 ROUTINES -

```
154,K 175552  053                     RTZ
155,K
156,K 175553  257                     XRM                     ELSE RESET READY BIT
157,K 175554  370                     LMA
158,K 175555  066 152 367             MLL     SEKBCH          GET THE CHARACTER
159,K 175560  176 044 177             NDL     0177            STRIP OFF THE SIGN BIT
160,K 175563  056 334                 LH      SEKTRAN>8       GENERATING INDEX INTO TRANSLATE TABLE
161,K 175565  307                     LAM                     GET TRANSLATED CODE
162,K 175566  115 265                 ORHH                    INSURE ZERO FLAG IS FALSE
163,K 175570  007                     RET
164,K
165,K 175571                          *
166,K                                 DOSF63
167,K                                 .
168,K                                 . DOS FUNCTION:  6   SUBFUNCTION:  3
169,K                                 .
170,K                                 . - WRITE A BYTE TO THE SCREEN -
171,K                                 .
172,K                                 .       ENTRY: B = BYTE TO BE WRITTEN
173,K                                 .              D = HORIZONTAL CURSOR POSITION
174,K                                 .              E = VERTICAL CURSOR POSITION
175,K                                 .
176,K                                 .       EXITS: IF D OR E OUT OF RANGE:
177,K                                 .                 CARRY TRUE
178,K                                 .                 A  = SCRATCHED
179,K                                 .              IF D AND E WITHIN RANGE:
180,K                                 .                 A  = SCRATCHED
181,K                                 .                 HL = BUFFER ADDRESS OF BYTE WRITTEN
182,K                                 .              1 EXTRA STACK LEVELS USED (MAX)
183,K                                 .
184,K                                 .       NOTE:  THIS ROUTINE CAN OVER-WRITE A BLINKING CURSOR
185,K                                 .              BUT THE BLINK ROUTINE IS WRITTEN TO ALLOW IT BY NOT
186,K                                 .              RESTORING THE CHARACTER BEHIND THE CURSOR IF THE
187,K                                 .              CURSOR IS NOT THERE ANY MORE.
188,K                                 .              NOT CHECKED BECAUSE FASTER BY ELIMINATING CALL TO CURSES.
189,K 175571  106 312 374             CALL    DSPCBL          COMPUTE BUFFER LOCATION
190,K 175574  043                     RTC                     EXIT IF OUT OF RANGE
191,K
192,K 175575  371                     LMB                     ELSE STORE THE BYTE
193,K 175576  007                     RET                        AND EXIT FALSE CARRY
```

```
194,K
195,K 175577                    *
196,K                           DOSF64
197,K                           .
198,K                           . DOS FUNCTION:  6   SUBFUNCTION:  4
199,K                           .
200,K                           . - RETURN HOME UP POSITION IN (DE) -
201,K                           .
202,K                           .         EXITS: D = HOME UP HORIZONTAL CURSOR POSITION
203,K                           .                E = HOME UP VERTICAL CURSOR POSITION
204,K                           .                0 EXTRA STACK LEVELS USED (MAX)
205,K 175577  046 364           .
206,K 175601  036 000                    LE      BL+1-MAXLIN
207,K 175603  007                        LD      LC
208,K                                    RET
209,K 175604                    *
210,K                           DOSF65
211,K                           .
212,K                           . DOS FUNCTION:  6   SUBFUNCTION:  5
213,K                           .
214,K                           . - RETURN HOME DOWN POSITION IN (DE) -
215,K                           .
216,K                           .         EXITS: D = HOME DOWN HORIZONTAL CURSOR POSITION
217,K                           .                E = HOME DOWN VERTICAL CURSOR POSITION
218,K                           .                0 EXTRA STACK LEVELS USED (MAX)
219,K 175604  046 013           .
220,K 175606  036 000                    LE      BL
221,K 175610  007                        LD      LC
222,K                                    RET
223,K 175611                    *
224,K                           DOSF66
225,K                           .
226,K                           . DOS FUNCTION:  6   SUBFUNCTION:  6
227,K                           .
228,K                           . - TURN THE CURSOR ON -
229,K                           .
230,K                           .         ENTRY: D = HORIZONTAL CURSOR POSITION
231,K                           .                E = VERTICAL CURSOR POSITION
232,K                           .
233,K                           .         EXITS: A,B,C,H,L = SCRATCHED
234,K                           .                IF D OR E OUT OF RANGE: CURSOR "ON" BUT NOT FLASHING
235,K                           .                3 EXTRA STACK LEVELS USED (MAX)
236,K 175611  066 242 056 357 307 .
237,K 175616  064 001                    MLA     *SECFLAGS       SET THE CURSOR ON/OFF SWITCH TO ON
238,K 175620  370                        OR      CURONOFF
239,K                                    LMA
                                .                                FALLS THROUGH INTO BLINKDE
```

```
240,K                                   *
241,K 175621                            BLINKDE
242,K                                   .
243,K                                   . IF THE CURSOR POSITION GIVEN IN (DE) IS DIFFERENT FROM THE OLD POSITION,
244,K                                   .       SUSPEND THE CURSOR AND UPDATE THE OLD CURSOR POSITION TO THE NEW.
245,K                                   .              3 EXTRA STACK LEVELS USED (MAX)
246,K                                   .
247,K 175621  066 245 056 357           .       HL      SECPOS          BC = OLD CURSOR POSITION
248,K 175625  111 047                   .       DL      BC,HL
249,K 175627  062 254                   .       XREC                    SEE IF BC = DE
250,K 175631  111 253                   .       XRDB
251,K 175633  062 261                   .       ORBC
252,K 175635  022 070                   .       PUSH    XA              SAVE XA IN CASE CURSES IS CALLED
253,K 175637  112 237 374               .       CFZ     CURSES          SUSPEND CURSOR IF BC .NE. DE
254,K 175642  022 060                   .       POP     XA              RESTORE XA
255,K                                   *
256,K 175644                            BLINK
257,K                                   .
258,K                                   . IF THE CURSOR ON/OFF SWITCH IS ON AND THE LINE FREQUENCY COUNTER BIT
259,K                                   .       USED FOR FLASHING IS DIFFERENT FROM THE CURSOR "IN BUFF" BIT,
260,K                                   .       UPDATE THE CURSOR "IN BUFF" BIT AND THEN IF THE CURSOR POSITION
261,K                                   .       IS WITHIN THE RANGE OF THE SCREEN, EITHER RESTORE THE HIDDEN
262,K                                   .       CHARACTER INTO THE BUFFER IF THE "IN BUFF" BIT IS NOW A ZERO OR SAVE
263,K                                   .       THE BUFFER LOCATION IN THE HIDDEN CHARACTER AND PUT THE CURSOR
264,K                                   .       CHARACTER IN THE BUFFER IF THE "IN BUFF" BIT IS A ONE.
265,K                                   .       IF ON RESTORING THE HIDDEN CHARACTER IT IS NOTED THAT THE CURSOR
266,K                                   .       CHARACTER IS NOT IN THE BUFFER (OVERWRITTEN) THEN THE CHARACTER IS
267,K                                   .       NOT RESTORED (ALLOWING WRITING IN THE BUFFER ON TOP OF THE CURSOR).
268,K                                   .              1 EXTRA STACK LEVELS USED (MAX)
269,K                                   .
270,K 175644  066 242 056 357 307       MLA     *SECFLAGS       EXIT IF THE CURSOR IS OFF
271,K 175651  044 001                   ND      CURONOFF
272,K 175653  053                       RTZ
273,K 175654  307                       LAM                     EXIT IF THE "OFF" BIT EQUALS THE
274,K 175655  066 153                   LL      SELFREQ           LINE FREQUENCY BIT
275,K 175657  257                       XRM
276,K 175660  044 010                   ND      CURINBUF
277,K 175662  053                       RTZ
278,K                                   .
279,K 175663  066 242                   LL      SECFLAGS        UPDATE THE "IN BUF" BIT
280,K 175665  257                       XRM
281,K 175666  370                       LMA
282,K 175667  066 245                   LL      SECPOS          GET THE CURSOR POSITION
283,K 175671  047                       DL      DE,HL
284,K 175672  106 312 374               CALL    DSPCBL          GET THE MEMORY ADDRESS IN THE BUFFER
285,K 175675  043                       RTC                     EXIT IF THE CURSOR IS OFF THE SCREEN
286,K                                   .
287,K 175676  022 070                   PUSH    XA              SAVE THE X-REGISTER
288,K 175700  076 357                   LX      SECFLAGS>8      SET UP THE PAGE REGISTER
289,K 175702  105 242                   PL      A,SECFLAGS
290,K 175704  044 010                   ND      CURINBUF        BRANCH ON CURSOR POSITION (IN/OUT BUFFER)
291,K 175706  105 244                   PL      A,SECCHAR         (GET THE CURSOR CHARACTER)
```

```
292,K 175710  150 322 373          JTZ       BLINKOUT        RESTORE CHARACTER IF THE CURSOR WENT OUT
293,K
294,K 175713  327                  LCM                       ELSE SAVE THE NEW HIDDEN CHARACTER
295,K 175714  126 243              PS        C,SECHIDE          AND PUT THE CURSOR CHARACTER
296,K 175716  370                  LMA                          INTO THE DISPLAY BUFFER
297,K 175717  022 060              POP       XA              RESTORE THE XA REGISTERS
298,K 175721  007                  RET
299,K
300,K 175722  124 243    BLINKOUT  PL        C,SECHIDE          (GET THE HIDDEN CHARACTER)
301,K 175724  277                  CPM                       IS THE CURSOR STILL IN MEMORY
302,K 175725  022 060              POP       XA
303,K 175727  013                  RFZ                       NO, IT WAS OVERWRITTEN
304,K 175730  372                  LMC                       YES, RESTORE THE HIDDEN CHARACTER
305,K 175731  007                  RET
```

```
                        - DOS FUNCTION 6 ROUTINES -

                        +
                        .                                       ** MOVED HERE TO SAVE 3 BYTES ***
                        .                                       ** PART OF THE DISPLAY CODE   ***
306,K                   DISPNEWL  LD      LC                    RESET THE HORIZONTAL POSITION
307,K 175734  174 004 001         ADE     1                     BUMP THE VERTICAL POSITION
308,K 175737  174 074 014         CPE     BL+1                  EXIT IF NOT AT BOTTOM OF SCREEN
309,K 175742  013                 RFZ
310,K                   .
311,K 175743  046 013             LE      BL                    ELSE SET VERT POSITION TO BOTTOM
312,K                   .         JMP     DOSF67                   OF SCREEN, ROLL UP, AND RETURN
313,K                   *
314,K 175745            DOSF67
315,K                   .
316,K                   . DOS FUNCTION:  6   SUBFUNCTION:  7
317,K                   .
318,K                   . - ROLL THE SCREEN UP ONE LINE -
319,K                   .
320,K                   .       ENTRY: D = HORIZONTAL CURSOR POSITION
321,K                   .              E = VERTICAL CURSOR POSITION
322,K                   .
323,K                   .       EXITS: ALL REGISTERS PRESERVED
324,K                   .              IF D OR E OUT OF RANGE:
325,K                   .                 CARRY TRUE
326,K                   .                 SCREEN IN ORIGINAL POSITION
327,K                   .              IF D AND E BOTH IN RANGE:
328,K                   .                 CARRY FALSE
329,K                   .                 SCREEN ROLLED ONE LINE
330,K                   .            3 EXTRA STACK LEVELS USED (MAX)
331,K                   .
332,K 175745  106 225 374         CALL    CURSESRS              SAVE REGISTERS AND SUSPEND CURSOR
333,K 175750  106 312 374         CALL    DSPCBL                SEE IF DE IS WITHIN RANGE
334,K 175753  043                 RTC                           RESTORE REGS AND EXIT IF NOT
335,K                   .
336,K 175754  115 164 156         DPL     HL,SEDLN01            PUT THE ADDRESS OF THE TOP LINE INTO
337,K 175757  115 166 236         DPS     HL,SEDNUB                THE POINTER BELOW THE BOTTOM LINE
338,K                   .
339,K 175762  250                 XRA
340,K 175763  106 114 374         CALL    BLANKIT               BLANK OUT THE WHOLE LINE
341,K                   .
342,K 175766  046 156 036 357     DE      SEDLN01               ROLL UP THE LINE POINTERS
343,K 175772  353                 LHD
344,K 175773  066 160             LL      SEDLN01+2
345,K 175775  026 060             LC      MAXLIN*2              (ALL THE LINES)
346,K 175777  250                 XRA
347,K 176000  310                 LBA
348,K 176001  021                 BT
349,K                   .
350,K 176002  250                 XRA                           SET CARRY FALSE
351,K 176003  007                 RET                           RESTORE THE REGISTERS AND EXIT
352,K
353,K
354,K
```

```
                                - DOS FUNCTION 6 ROUTINES -

                                +
                                DSPINIT
                                .
                                . INITIALIZE ALL DISPLAY POINTERS AND CONTROL INFORMATION
                                .                 3 EXTRA STACK LEVELS USED (MAX)
                                .
176004                          .         DE      SDSPBUF         POINT TO THE SCREEN BUFFER
176010                          .         HL      SEDLN01         POINT TO THE POINTER LIST
176014                          .         LA      MAXPOS          LENGTH OF EACH LINE
                                .
176016                          DSPINITL  DS      DE,HL           SET UP A LINE POINTER
176017                          .         INCP    DE,A            BUMP THE POINTER VALUE
176021                          .         INCP    HL,2            BUMP THE POINTER ADDRESS
176023                          .         CPL     SEDLN01+(2*MAXLIN)   GO UNTIL ALL LINES DONE
176026                          .         JFZ     DSPINITL
                                .
176031                          .         XRA                     ZERO THE DISPLAY ROUTINE FLAGS
176032                          .         MSA     SEDOPTS+1          USED BY THE DOS FUNCTIONS
                                .                                 SET CURSOR FLAGS TO SUSPENDED
176035                          .         MSA     SECFLAGS           AND NOT IN THE BUFFER
                                .         LA      CURSOR          INITIALIZE THE CHARACTER (ASSUMED ZERO)
176040                          .         MSA     SECCHAR            USED FOR THE CURSOR
                                .
176043                          .         CALL    DOSF64          GET HOME UP POSITION IN (DE)
                                .         JUMP    DOSF68          ERASE THE ENTIRE SCREEN AND EXIT
                                *
176046                          DOSF68
                                .
                                . DOS FUNCTION:  6   SUBFUNCTION:  8
                                .
                                . - ERASE FROM THE CURSOR THRU THE END OF FRAME -
                                .
                                .         ENTRY: D = HORIZONTAL CURSOR POSITION
                                .                E = VERTICAL CURSOR POSITION
                                .
                                .         EXITS: ALL REGISTERS PRESERVED
                                .                IF D OR E OUT OF RANGE:
                                .                   CARRY TRUE
                                .                   NO LINES ERASED
                                .                IF D AND E BOTH WITHIN RANGE:
                                .                   CARRY FALSE
                                .                   LINE(S) ERASED
                                .                3 EXTRA STACK LEVELS USED (MAX)
                                .
176046                          .         CALL      CURSESRS          SAVE REGISTERS AND SUSPEND CURSOR
176051                          .         LB        ' '               STANDARD ERASE CHARACTER
                                .
176053                          DOSF68LP  CALL      DSPCBL            COMPUTE THE BUFFER POSITION
176056                          .         RTC                         RESTORE REGS AND EXIT IF DE OUT OF RANGE
                                .
176057                          .         LAD                         BLANK TO THE END OF THAT LINE
176060                          .         LDB                           (SAVE THE B-REGISTER)
```

```
407,K 176061  174 070                     PUSH    DE
408,K 176063  106 116 374                 CALL    BLANKITB
409,K 176066  174 060                     POP     DE
410,K 176070  313                         LBD                     (RESTORE THE B-REGISTER)
411,K 176071  036 000                     LD      LC              RESET THE HORIZONTAL POSITION
412,K 176073  174 004 001                 ADE     1               INCRMENT THE VERTICAL LINE POSITION
413,K 176076  174 074 014                 CPE     BL+1            ERASE UNTIL BOTTOM LINE REACHED
414,K 176101  110 053 374                 JFZ     DOSF68LP
415,K
416,K 176104  007                         RET
417,K                                   *
418,K 176105                            DOSF69
419,K                                   .
420,K                                   . DOS FUNCTION:  6   SUBFUNCTION:  9
421,K                                   .
422,K                                   . - ERASE FROM THE CURSOR THRU THE END OF LINE -
423,K                                   .
424,K                                   .       ENTRY: D = HORIZONTAL CURSOR POSITION
425,K                                   .              E = VERTICAL CURSOR POSITION
426,K                                   .
427,K                                   .       EXITS: (SEE "DOSF68" ABOVE)
428,K                                   .              3 EXTRA STACK LEVELS USED (MAX)
429,K                                   .
430,K 176105  106 225 374                 CALL    CURSESRS        SAVE REGISTERS AND SUSPEND CURSOR
431,K 176110  106 312 374                 CALL    DSPCBL          POINT TO LINE TO BE BLANKED
432,K 176113  303                         LAD                     ERASE TO THE END OF THE LINE
433,K                                     JMP     BLANKIT
434,K                                   *
435,K 176114                            BLANKIT
436,K                                   .
437,K                                   . BLANK A LINE WITH EITHER NORMAL OR INVERTED VIDEO BLANKS
438,K                                   .
439,K                                   .       ENTRY   A       = HORZ CURSOR POSITION
440,K                                   .               B       = BLANKING CHARACTER (IF ENTER AT "BLANKITB")
441,K                                   .               HL      = ADDRESS OF THE LINE
442,K                                   .               X       = SEDOPTS>8
443,K                                   .               SEDOPTS = INVERTED VIDEO OPTION IN B7
444,K                                   .               0 EXTRA STACK LEVELS USED
445,K                                   .
446,K 176114  016 040                     LB      ' '             BLANK LINE WITH STANDARD BLANK CODE
447,K 176116  026 117                   BLANKITB  LC      MAXPOS-1        COMPUTE HOW MANY CHARACTERS TO BLANK
448,K 176120  062 220                     SUAC
449,K 176122  105 241                     PL      A,SEDOPTS+1     GET THE INVERTED VIDEO OPTION FLAG
450,K 176124  044 200                     ND      $OI
451,K 176126  261                         ORB                     'OR' IN A BLANK CHARACTER
452,K 176127  370                         LMA                     BLANK THE FIRST CHARACTER
453,K 176130  062 262                     ORCC                    EXIT OF C = 0
454,K 176132  053                         RTZ
455,K                                   .
456,K 176133  346                         LEL                     DE => NEXT CHARACTER
457,K 176134  335                         LDH
458,K 176135  174 015                     INCP    DE
```

```
                        - DOS FUNCTION 6 ROUTINES -

459,K 176137  250                     XRA                     ELSE DO THE BLOCK TRANSFER
460,K 176140  310                     LBA                        TO CLEAR THE REST OF THE LINE
461,K 176141  021                     BT
462,K 176142  250                     XRA                     EXIT FALSE CARRY
463,K 176143  007                     RET
464,K                         *
465,K 176144                  DOSF610
466,K                         .
467,K                         . DOS FUNCTION:  6   SUBFUNCTION:  10
468,K                         .
469,K                         . - ROLL THE SCREEN DOWN ONE LINE -
470,K                         .
471,K                         .       ENTRY: D = HORIZONTAL CURSOR POSITION
472,K                         .              E = VERTICAL CURSOR POSITION
473,K                         .
474,K                         .       EXITS: (SEE "DOSF67" ABOVE)
475,K                         .              3 EXTRA STACK LEVELS USED (MAX)
476,K                         .
477,K 176144  106 225 374             CALL    CURSESRS        SAVE REGISTERS AND SUSPEND CURSOR
478,K 176147  106 312 374             CALL    DSPCBL          SEE IF DE WITHIN RANGE
479,K 176152  043                     RTC                     RESTORE REGS AND EXIT IF NOT
480,K                         .
481,K 176153  115 164 234             DPL     HL,SEDLBOT      PUT THE ADDRESS OF THE BOTTOM LINE
482,K 176156  115 166 154             DPS     HL,SEDNUL          IN THE POINTER ABOVE THE TOP LINE
483,K                         .
484,K 176161  250                     XRA
485,K 176162  106 114 374             CALL    BLANKIT         BLANK OUT THE WHOLE LINE
486,K                         .
487,K 176165  046 235 036 357         DE      SEDLBOT+1       ROLL DOWN THE LINE POINTERS
488,K 176171  353                     LHD
489,K 176172  066 233                 LL      SEDLBOT-2+1
490,K 176174  026 060                 LC      MAXLIN*2        (ALL THE LINES)
491,K 176176  250                     XRA
492,K 176177  310                     LBA
493,K 176200  111 021                 BTR
494,K                         .
495,K 176202  250                     XRA                     SET CARRY FALSE
496,K 176203  007                     RET                     RESTORE THE REGISTERS AND EXIT
497,K                         *
498,K 176204                  DOSF611
499,K                         .
500,K                         . DOS FUNCTION:  6   SUBFUNCTION:  11
501,K                         .
502,K                         . - TURN THE CURSOR OFF -
503,K                         .
504,K                         .       ENTRY: D = HORIZONTAL CURSOR POSITION
505,K                         .              E = VERTICAL CURSOR POSITION
506,K                         .
507,K                         .       EXITS: ALL REGISTERS PRESERVED
508,K                         .              3 EXTRA STACK LEVELS USED (MAX)
509,K                         .              A-REG CHANGED IF DOSF611N ENTRY USED, CURSOR MUST BE SUSPENDED
510,K                         .
```

```
                                - DOS FUNCTION 6 ROUTINES -

511,K 176204  106 225 374               CALL    CURSESRS        SAVE REGISTERS AND SUSPEND CURSOR
512,K 176207                    DOSF611N
513,K 176207  105 242                   PL      A,SECFLAGS      SET THE CURSOR ON/OFF BIT TO "OFF"
514,K 176211  044 376                   ND      -1-CURONOFF
515,K 176213  107 242                   PS      A,SECFLAGS
516,K 176215  007                       RET                     RESTORE THE REGISTERS AND EXIT
517,K                           *
518,K 176216                    DOSF612
519,K                           .
520,K                           . DOS FUNCTION:  6   SUBFUNCTION:  12
521,K                           .
522,K                           . - RETURN FUNCTION KEY AND OTHER STATUS -
523,K                           .
524,K                           .       EXITS: ALL REGISTERS PRESERVED
525,K                           .              BC EXCEPTED, THEY CONTAIN THE STATUS BITS
526,K                           .              B = (X, X, X, X, SEKBDWN, SEKBRDY, SEKBDKY, SEDSPKY)
527,K                           .              C = (SEINTKY, SEATTKY, SERSTKY, SEFUNC5, 4, 3, 2, 1)
528,K                           .
529,K                           .              0 EXTRA STACK LEVELS USED (MAX)
530,K                           .
531,K                           . WARNING: CHANGING OF LOCATIONS SEKBS2 OR SEKBS1 WITHOUT DUE REGARD FOR EFFECT
532,K                           .          CAN CAUSE LOSS OF CHARACTERS, FUNCTION KEYS AND AUTO RESTARTS!!!
533,K                           .
534,K 176216  026 150 016 357           BC      SEKBS2
535,K 176222  062 047                   DL      BC,BC           GET THE STATUS BYTES
536,K 176224  007                       RET
```

- DISPLAY INTERFACE CONTROL SUBROUTINES -

```
539,K                                   *
540,K 176225                            CURSESRS
541,K                                   .
542,K                                   . SAVE THE REGISTERS AND CAUSE A "RETURN" AFTER THE CALL TO THIS ROUTINE
543,K                                   .       TO TRANSFER CONTROL TO "DSPRESTR".  THEN SUSPEND THE CURSOR.
544,K                                   .             3 EXTRA STACK LEVELS USED (MAX)
545,K                                   .
546,K 176225  051 271 335                       PUSH    DSPREGSA+7      SAVE THE REGISTERS IN THE
547,K 176230  055                               REGS                       AREA FOR THE DISPLAY
548,K 176231  060                               POP     HL              GET RID OF "DSPREGSA" ADDRESS ON STACK
549,K 176232  060                               POP     HL              PUT "DSPRESTR" ON STACK BELOW
550,K 176233  051 303 374                       PUSH    DSPRESTR           THE RETURN ADDRESS
551,K 176236  070                               PUSH    HL
552,K                                   *
553,K 176237                            CURSES
554,K                                   .
555,K                                   . SAVE THE NEW CURSOR POSITION GIVEN IN (DE) IN "SECPOS", THEN, IF
556,K                                   .       THE CURSOR CHARACTER IS IN THE BUFFER, SUSPEND THE CURSOR
557,K                                   .       BY PLACING THE CHARACTER IN "SECHIDE" BACK INTO THE BUFFER
558,K                                   .       AS DIRECTED BY THE OLD CURSOR POSITION BEFORE THIS ROUTINE
559,K                                   .       WAS ENTERED.
560,K                                   .
561,K                                   .       ENTRY: D = HORIZONTAL CURSOR POSITION
562,K                                   .              E = VERTICAL CURSOR POSITION
563,K                                   .
564,K                                   .       EXITS: HL,A = SCRATCHED
565,K                                   .              X => SECFLAGS PAGE
566,K                                   .              1 EXTRA STACK LEVELS USED (MAX)
567,K                                   .
568,K 176237  076 357                           LX      SECFLAGS>8
569,K 176241  115 164 245                       DPL     HL,SECPOS       SAVE THE OLD CURSOR POSITION IN HL
570,K 176244  113 146 245                       DPS     DE,SECPOS       UPDATE THE CURSOR POSITION TO THAT IN DE
571,K 176247  105 242                           PL      A,SECFLAGS      EXIT IF THE CURSOR CHARACTER IS
572,K 176251  044 010                           ND      CURINBUF           NOT CURRENTLY IN THE BUFFER
573,K 176253  053                               RTZ                        (THE "CURIN BUFF" BIT IS A ZERO)
574,K                                   *
575,K                                   . CLEAR THE CURSOR ON BIT AND STORE THE HIDDEN CHARACTER IN THE
576,K                                   .       CURSOR POSITION GIVEN IN (HL)
577,K                                   .
578,K 176254  105 242                           PL      A,SECFLAGS      ELSE CLEAR THE CURSOR IN BUFF BIT
579,K 176256  044 367                           ND      -1-CURINBUF
580,K 176260  107 242                           PS      A,SECFLAGS
581,K 176262  346                               LEL                     HL => WHERE THE CURSOR WAS
582,K 176263  335                               LDH
583,K 176264  106 312 374                       CALL    DSPCBL
584,K 176267  113 144 245                       DPL     DE,SECPOS       RESTORE THE NEW CURSOR POSITION IN DE
585,K 176272  043                               RTC                     EXIT IF CURSOR WAS OFF THE SCREEN
586,K                                   .
587,K 176273  105 244                           PL      A,SECCHAR       CHECK IF THE CURSOR IS STILL IN MEMORY
588,K 176275  277                               CPM                        IF NOT, DON'T RESTORE THE HIDDEN CHAR
589,K 176276  013                               RFZ
590,K 176277  105 243                           PL      A,SECHIDE       ELSE, STORE THE HIDDEN CHARACTER
```

- DISPLAY INTERFACE CONTROL SUBROUTINES -

```
591,K 176301  370                     LMA                             WHERE THE CURSOR WAS
592,K 176302  007                     RET                             AND EXIT
593,K
                                *
594,K 176303  066 271 056 335 DSPRESTR  HL      DSPREGSA+7      RESTORE THE REGISTERS FROM THE
595,K 176307  111 055                   REGL                        SAVE AREA FOR THE DISPLAY
596,K 176311  007                       RET                         ROUTINES.
597,K
                                *
598,K 176312                    DSPCBL
599,K
                                .
600,K                           . COMPUTE DISPLAY BUFFER LOCATION
601,K                           .
602,K                           .       ENTRY: D  = HORIZONTAL CURSOR POSITION
603,K                           .              E  = VERTICAL CURSOR POSITION
604,K                           .
605,K                           .       EXITS: IF D AND E BOTH WITHIN RANGE:
606,K                           .                CARRY FALSE
607,K                           .                HL = MEMORY LOCATION IN THE SCREEN BUFFER
608,K                           .                A  = ZERO
609,K                           .              IF D OR E NOT WITHIN RANGE:
610,K                           .                CARRY TRUE
611,K                           .                A  = SCRATCHED
612,K                           .              0 EXTRA STACK LEVELS USED (MAX)
613,K                           .
614,K 176312  303                       LAD                             EXIT TRUE CARRY IF D OUT OF RANGE
615,K 176313  004 260                   AD      -MAXPOS
616,K 176315  043                       RTC
617,K
                                .
618,K 176316  304                       LAE                             EXIT TRUE CARRY IF E OUT OF RANGE
619,K 176317  024 014                   SU      BL+1                      MAKE A = -MAXLIN THRU -1
620,K 176321  074 350                   CP      -MAXLIN
621,K 176323  043                       RTC
622,K
                                .
623,K 176324  200                       ADA                             DOUBLE UNBIASED E VALUE TO GET INDEX
624,K 176325  066 236 056 356           HL      SEDNUB-256                 INTO THE LIST OF LINE POINTERS
625,K 176331  017                       INCP    HL,A                       (256 CORRECTS FOR SINGLE BYTE INCP)
626,K 176332  057                       DL      HL,HL                   GET POINTER TO BEGINNING OF LINE
627,K 176333  303                       LAD                             INDEX INTO THE SELECTED LINE
628,K 176334  017                       INCP    HL,A
629,K 176335  250                       XRA                             EXIT FALSE CARRY
630,K 176336  007                       RET
```

```
633,K                           *
634,K                           . DISPLAY A STRING ROUTINE
635,K                           .
636,K                           .       CONTROL CHARACTERS:
637,K                           .
638,K 000003                    $ES*    EQU     003             END OF DISPLAY STRING
639,K 000007                    $BP*    EQU     007             BEEP
640,K 000011                    $H*     EQU     011             NEW DISPLAY COLUMN FOLLOWS
641,K 000013                    $V*     EQU     013             NEW DISPLAY ROW FOLLOWS
642,K 000015                    $EL*    EQU     015             ADVANCE TO NEW LINE AND TERMINATE STRING
643,K 000021                    $EEOF*  EQU     021             ERASE TO END OF FRAME
644,K 000022                    $EEOL*  EQU     022             ERASE TO END OF LINE
645,K 000023                    $RU*    EQU     023             ROLL UP
646,K 000024                    $RD*    EQU     024             ROLL DOWN
647,K 000033                    $F*     EQU     033             FORCE DISPLAY OF NEXT CHARACTER
648,K                           .
649,K 000203                    $NS*    EQU     0203            NEW STRING ADDRESS FOLLOWS (LSB,MSB)
650,K 000207                    $CK*    EQU     0207            CLICK
651,K 000211                    $HA*    EQU     0211            DISPLAY COLUMN ADJUSTMENT FOLLOWS
652,K 000213                    $VA*    EQU     0213            DISPLAY ROW ADJUSTMENT FOLLOWS
653,K 000215                    $NL*    EQU     0215            ADVANCE TO NEW LINE
654,K                           .       EQU     0221            * NOT USED *
655,K                           .       EQU     0222            * NOT USED *
656,K 000223                    $HU*    EQU     0223            HOME UP (UPPER LEFT-HAND CORNER)
657,K 000224                    $HD*    EQU     0224            HOME DOWN (LOWER LEFT-HAND CORNER)
658,K 000233                    $O*     EQU     0233            NEW OPTIONS FOLLOW
659,K                           .
660,K 000200                    $OI*    EQU     B7                 INVERTED VIDEO OPTION
661,K 000100                    $OS*    EQU     B6                 SKIP BLANKS OPTION
662,K 000040                    $OW*    EQU     B5                 INHIBIT WAIT ON DISPLAY KEY OPTION
```

```
663,K                                   *
664,K                                   * THE DISPLAY ROUTINE ENTRY POINTS
665,K                                   *
666,K                                   *       "DISPDOS" INITIALIZES THE OPTIONS IN THE B-REGISTER TO
667,K                                   *                 A DOS-COMPATIBLE SET
668,K                                   *       "DISPLAY" STORES THE OPTIONS IN THE B-REGISTER IN "SEDOPTS"
669,K                                   *
670,K                                   *       ENTRY: B  = OPTIONS (UNLESS ENTERING AT DISPDOS)
671,K                                   *              D  = HORIZONTAL CURSOR POSITION (0 THUR 79)
672,K                                   *              E  = VERTICAL CURSOR POSITION (-12 THRU 11)
673,K                                   *              HL = STRING ADDRESS
674,K                                   *
675,K                                   *       EXITS: A  = ENTRY VALUE
676,K                                   *              B  = LAST OPTION VALUE SET
677,K                                   *              C  = ENTRY VALUE
678,K                                   *              DE = CURSOR POSITION AFTER LAST CHARACTER DISPLAYED
679,K                                   *                   OR CURSOR POSITION WHEN LAST CONTROL EXECUTED
680,K                                   *              HL = ADDRESS OF BYTE AFTER STRING TERMINATOR
681,K                                   *              6 EXTRA STACK LEVELS USED (MAX)
682,K                                   *                EXCLUDING DISPLAY WAIT VECTOR OVERHEAD (IF USED)
683,K                                   *
684,K 176337  016 140                   DISPDOS   LB      $DS+$DW         SKIP BLANKS AND INHIBIT DSP KEY WAIT
685,K                                   *
686,K 176341  022 070                   DISPLAY   PUSH    XA              SAVE THE X-REGISTER
687,K 176343  070                               PUSH    HL              SAVE THE STRING POINTER
688,K 176344  105 237 374                       CALL    CURSES          MAKE SURE THE CURSOR IS SUSPENDED
689,K 176347  111 126 240                       DPS     BC,SEDOPTS      SAVE THE BC REGISTER PAIR IN 'SEDOPTS'
690,K                                   *
691,K 176352  060                       DISPCPRP  POP     HL              RESTORE THE STRING POINTER
692,K 176353  006 377                   DISPCPSL  LA      -1              CLEAR THE PHYSICAL SCREEN POINTER
693,K 176355  107 246                           PS      A,SECPOS+1          MSB TO ALL ONES (CAN BE IN ROM!)
694,K                                   *
695,K 176357                            DISPLANC
696,K                                   *
697,K                                   * DISPLAY THE NEXT CHARACTER IN THE STRING POINTED TO BY (HL)
698,K                                   *
699,K 176357  307                               LAM                     GET THE NEXT STRING CHARACTER
700,K 176360  015                               INCP    HL              BUMP THE STRING POINTER
701,K                                   *
702,K 176361  310                               LBA                     SAVE THE CHARACTER IN THE B-REG
703,K 176362  124 241                           PL      C,SEDOPTS+1     C = DISPLAY OPTION BITS
704,K                                   *
705,K 176364  074 033                           CP      $F              FALL THROUGH IF THE CHARACTER
706,K 176366  150 062 375                       JTZ     DISPFORC              (SPECIAL HANDLING)
707,K 176371  070                               PUSH    HL                    (SAVE THE STRING POINTER)
708,K 176372  140 007 375                       JTC     DISPFUNC        CANNOT BE A CONTROL
709,K 176375  074 203                           CP      $NS
710,K 176377  140 065 375                       JTC     DISPOUT
711,K 176402  074 234                           CP      $0+1
712,K 176404  100 065 375                       JFC     DISPOUT
713,K                                   *
714,K 176407  066 152 056 375           DISPFUNC  HL      DISPFTAB-2      SEE IF FUNCTION IN TABLE
```

```
715,K 176413  117 015          DISPFCLP  INCP      HL,2
716,K 176415  307                        LAM
717,K 176416  260                        ORA
718,K 176417  150 065 375                JTZ       DISPOUT          WRITE CHAR IF TERMINATOR REACHED
719,K                          .
720,K 176422  015                        INCP      HL               HL => FUNCTION EXECUTION ADDRESS
721,K 176423  271                        CPB                        SEE IF CONTROL CHARACTER DESIRED
722,K 176424  110 013 375               JFZ       DISPFCLP         KEEP SCANNING IF NOT
723,K                          .
724,K 176427  302                        LAC                        SEE IF THE "INHIBIT WAIT BEFORE CONTROL"
725,K 176430  044 040                    ND        $OW                 OPTION BIT IS SET
726,K 176432  110 047 375               JFZ       DISPDOCC         JUST DO CONTROL FUNCTION IF IT IS
727,K                          .
728,K 176435  106 105 357      DISPDSPW  CALL      SVDISPWS         CALL THE DISPLAY WAIT ROUTINE
729,K                                                               (WILL ALWAYS DO IT ONCE IF ENABLED!!)
730,K 176440  105 151          .                                   ELSE SEE IF THE DISPLAY KEY IS DOWN
731,K 176442  044 001                    PL        A,SEKBS1
732,K 176444  110 035 375                ND        SEDSPKY
733,K                                    JFZ       DISPDSPW         WAIT ON IT IF IT IS
734,K 176447  062 060          .
735,K 176451  051 353 374      DISPDOCC  POP       BC               PUT STRING POINTER INTO BC FOR A WHILE
736,K 176454  057                        PUSH      DISPCPSL         PUT NEXT CHAR RETURN ADR ON THE STACK
737,K 176455  070                        DL        HL,HL            GET THE CONTROL ROUTINE ADDRESS
738,K 176456  362                        PUSH      HL                  AND PUT THAT ON THE STACK
739,K 176457  351                        LLC                        MAKE HL => NEXT STRING CHARACTER
740,K 176460  250                        LHB
741,K 176461  007                        XRA                        MAKE A = 0 FOR $H AND $V
742,K                                    RET                        OFF TO THE CONTROL ROUTINE
743,K 176462  317              *
744,K 176463  015              DISPFORC  LBM                        FORCE THE NEXT CHARACTER
745,K 176464  070                        INCP      HL                  TO BE DISPLAYED REGARDLESS OF ITS
746,K                                    PUSH      HL                  VALUE
747,K                          .         JUMP      DISPOUT             NOTE: SPECIAL HANDLING OF THIS $F CODE
748,K 176465  302              *
749,K 176466  044 200          DISPOUT   LAC                        ISOLATE THE VIDEO INVERSION OPTION BIT
750,K 176470  111 250                    ND        B7                  AND INVERT THE SIGN BIT OF THE
751,K 176472  115 164 245                XRAB                          CHARACTER TO BE WRITTEN IF SET
752,K 176475  305                        DPL       HL,SECPOS        GET THE PHYSICAL SCREEN POSITION
753,K 176476  004 001                    LAH                        COMPUTE THE PHYSICAL LOCATION
754,K 176500  142 312 374                AD        1                   IF IT IS NOT KNOWN (IN ROM!)
755,K 176503  140 146 375                CTC       DSPCBL
756,K                                    JTC       DISPOFFS         CATCH OFF THE SCREEN
757,K 176506  006 177          .
758,K 176510  241                        LA        0177             CATCH WRITING SPACES
759,K 176511  074 040                    NDB
760,K 176513  110 124 375                CP        ' '
761,K                                    JFZ       DISPWRIT         JUST WRITE IF NOT
762,K 176516  302              .
763,K 176517  044 100                    LAC                        ELSE SEE IF SPACES ARE TO BE SKIPPED
764,K 176521  110 125 375                ND        B6
765,K                                    JFZ       DISPSKIP         DON'T WRITE IF SO
766,K 176524  371              .
                               DISPWRIT  LMB                        ELSE STORE THE BYTE ON THE SCREEN
```

```
767,K 176525  015                 DISPSKIP  INCP      HL              INCREMENT THE PHYSICAL SCREEN POSITION
768,K 176526  115 166 245                   DPS       HL,SECPOS       SAVE THE PHYSICAL SCREEN POSITION
769,K 176531  060                           POP       HL              RESTORE THE STRING POINTER
770,K 176532  113 004 001                   ADD       1               INCREMENT THE CURSOR POSITION
771,K 176535  113 074 120                   CPD       MAXPOS          ON TO THE NEXT STRING CHARACTER
772,K 176540  140 357 374                   JTC       DISPLANC           IF CURSOR STILL ON THE SCREEN
773,K                               .
774,K 176543  104 353 374                   JMP       DISPCPSL        ELSE CLEAR THE PHYSICAL SCREEN LOCATION
775,K                               *
776,K 176546  113 004 001           DISPOFFS  ADD       1               INCREMENT THE CURSOR POSITION AND
777,K 176551  104 352 374                   JMP       DISPCPRP        ON TO THE NEXT STRING CHARACTER
778,K                               *
779,K 176554                        DISPFTAB
780,K                               .
781,K                               . DISPLAY FUNCTION TABLE
782,K                               .
783,K 176554  011 254 375                   DC        $H,*DISPHCOL
784,K 176557  013 261 375                   DC        $V,*DISPVCOL
785,K 176562  211 253 375                   DC        $HA,*DISPHADJ
786,K 176565  213 260 375                   DC        $VA,*DISPVADJ
787,K 176570  215 332 373                   DC        $NL,*DISPNEWL
788,K 176573  203 240 375                   DC        $NS,*DISPGNSA
789,K 176576  223 177 373                   DC        $HU,*DOSF64
790,K 176601  224 204 373                   DC        $HD,*DOSF65
791,K                               .       DC        $F,*DISPFORC       (SPECIAL HANDLING)
792,K 176604  233 242 375                   DC        $O,*DISPGOPT
793,K 176607  023 345 373                   DC        $RU,*DOSF67
794,K 176612  024 144 374                   DC        $RD,*DOSF610
795,K 176615  022 105 374                   DC        $EEOL,*DOSF69
796,K 176620  021 046 374                   DC        $EEOF,*DOSF68
797,K 176623  007 247 375                   DC        $BP,*DISPBEEP
798,K 176626  207 251 375                   DC        $CK,*DISPCLIK
799,K 176631  015 265 375                   DC        $EL,*DISPENDL
800,K 176634  003 270 375                   DC        $ES,*DISPEXIT
801,K 176637  000                           DC        0               TERMINATOR
802,K                               *
803,K 176640  057                   DISPGNSA  DL        HL,HL           GET NEW STRING ADDRESS
804,K 176641  007                           RET
805,K                               *
806,K 176642  307                   DISPGOPT  LAM                       GET THE NEW OPTIONS
807,K 176643  015                           INCP      HL              INCREMENT THE STRING POINTER
808,K 176644  107 241                       PS        A,SEDOPTS+1     STORE THE NEW OPTIONS
809,K 176646  007                           RET
```

```
810,K
811,K 176647  151                *
812,K 176650  007  DISPBEEP  EX        BEEP
813,K                          RET
814,K 176651  153                *
815,K 176652  007  DISPCLIK  EX        CLICK
816,K                          RET
817,K 176653  303                *
818,K                DISPHADJ  LAD                     CAUSE HORIZONTAL ADJUSTMENT
819,K 176654  207                *                         WHEN ENTERED HERE
820,K 176655  330  DISPHCOL  ADM                     A = ZERO WHEN ENTERED HERE
821,K 176656  015                LDA                       SO D = STRING VALUE
822,K 176657  007                INCP    HL
823,K                          RET
824,K 176660  304                *
825,K                DISPVADJ  LAE                     CAUSE VERTICAL ADJUSTMENT
826,K 176661  207                *                         WHEN ENTERED HERE
827,K 176662  340  DISPVCOL  ADM                     A = ZERO WHEN ENTERED HERE
828,K 176663  015                LEA                       SO E = STRING VALUE
829,K 176664  007                INCP    HL
830,K                          RET
831,K 176665  106 332 373       *
832,K 176670  062 060  DISPENDL  CALL    DISPNEWL        CAUSE ADVANCE TO A NEW LINE
833,K 176672  111 124 240 DISPEXIT  POP     BC              CAUSE EXIT OF THE DISPLAY ROUTINE
834,K 176675  250                DPL     BC,SEDOPTS      RESTORE C-REG AND LAST OPTIONS IN B-REG
835,K 176676  107 241           XRA                     CLEAR THE OPTIONS
836,K 176700  022 060           PS      A,SEDOPTS+1
837,K 176702  007                POP     XA              RESTORE THE XA REGISTER PAIR
                                 RET                     FINALLY, EXIT THE DISPLAY ROUTINE
```

```
838,K
839,K 176703
840,K
841,K
842,K
843,K
844,K
845,K
846,K
847,K
848,K
849,K 176703  066 141 056 372
850,K 176707  016 040
851,K 176711  106 341 374
852,K 176714  113 024 012
853,K 176717  106 211 373
854,K 176722  106 140 373
855,K 176725  150 322 375
856,K
857,K 176730  310
858,K 176731  106 171 373
859,K 176734  076 357
860,K 176736  106 207 374
861,K 176741  301
862,K 176742  007
179,
180,
```

```
+
KEYCHAR
.
. GET A KEYBOARD CHARACTER
.
.         ENTRY: DE = CURSOR POSITION TO BE FLASHED WHILE WAITING
.
.         EXITS: A AND B = KEYBOARD CHARACTER
.                C,H,L - SCRATCHED
.                5 STACK LEVELS USED (MAX) * LESS? *
.
          HL      DSPBLNK
          LB      $OW             BLANK OUT SOME SPACE FOR THE KEYIN
          CALL    DISPLAY         WITHOUT USING DOSF69 (BECAUSE OF CURSESRS)
          SUD     DSPBLEN         CORRECT FOR UPDATING OF THE D-REGISTER
          CALL    DOSF66          TURN ON THE CURSOR
KEYCHARW  CALL    DOSF62          WAIT FOR A CHARACTER TO BE INPUT
          JTZ     KEYCHARW
.
          LBA                     ECHO THE CHARACTER
          CALL    DOSF63
          LX      SECFLAGS>8
          CALL    DOSF611N        TURN THE CURSOR OFF (OVERWRITTEN BY 63)
          LAB                     EXIT WITH THE CHARACTER IN A AND B
          RET
.
          INC     1800MOVI
```

```
                                        *
                                        * 1800 MOVING INVERSIONS MEMORY TEST                    JAN 30, 1978   14:43   HJS
                                        *
000004                                  MAXSTLS   EQU      4                  16K BLOCKS TESTED PER PASS
040000                                  MAXSZ     EQU      MAXSTLS<12         MEMORY SIZE BEING TESTED (POWER OF 2)
                                                                              MUST ALLOW 8K SPACE WITH MIN. MEMORY SIZE
                                        *
176743  022 040 061 062 070             MTMSG128  DC       $EEOL,' 128K',$ES
176752  040 066 064 003                 MTMSG64   DC       ' 64',$ES
176756  060 064                         MTINC     DC       '04'               STEPS IN 4K CHUNCKS
                                        *
176760                                  TSTMEM
176760  106 112 370                             CALL     C12345             SEE IF ENTERED MEMORY TEST LEGALLY
176763  106 045 373                             CALL     SETUP
176766  106 004 374                             CALL     DSPINIT            INITIALIZE THE WORLD
176771  020                                     BETA
176772  036 007                                 LD       8-1                DISPLAY ERROR ADDRESS POINTER (HORIZ)
176774  056 000                                 LH       0
176776  365                                     LLH                         INITIALIZE THE PASS NUMBER
177777  030                                     ALPHA
177000  046 000 036 100                         DE       MAXSZ              START INITIAL STEPSIZE AS 1
177004  104 114 377                             JMP      MSTART             FINISH INIT AS IF ENDED A PASS
                                        *
177007                                  MTSIZE
                                        *
                                        * SIZE CALCULATION DONE PER PASS
                                        *
177007  174 070                                 PUSH     DE                 SAVE THE INCREMENT SIZE
177011  046 006 036 111                         DE       73<8+BL-5          POINT TO MEMORY LOCATION FOR 'K' COUNTER
177015  066 343 056 375                         HL       MTMSG128
177021  106 337 374                             CALL     DISPDOS            DISPLAY THE INITIAL MESSAGE
177024  036 114                                 LD       76
177026  106 312 374                             CALL     DSPCBL             GET ADDRESS OF THE COUNTER IN DISPLAY RAM
177031  070                                     PUSH
                                        *
177032  066 264 056 363                         HL       WRAPST             SPECIAL WRAP AROUND MEMORY TEST
177036  026 002                                 LC       2
177040  077                                     STL
                                        *
177041  066 001 056 357                         HL       SVMEMP+1
177045  046 215 036 377                         DE       MTSIZMPF           SET INTERRUPT VECTOR FOR SIZE CALCULATION
177051  027                                     DS       DE,HL
                                        *
177052  076 000                                 LX       0>8                POINTER NEEDED LATER
177054  046 264                                 LE       SMST+PISTL         ASSUME MEMORY OVER 64K BOUNDARY
177056  250                                     XRA
                                        *
177057  066 377 056 020                         HL       010377             SET 72K PHYS. TO ALL ONES
177063  376                                     LML                            (OR 8K PHYS. IF WRAPPED)
177064  107 377                                 PS       A,000377           SET 8K PHYS. TO ZEROS
177066  277                                     CPM                         READ 72K PHYS. (OR 8K) TO SEE IF NON-ZERO
177067  110 110 376                             JFZ      MTSIZ32S           NO WRAP AROUND, USING 32K CARDS
```

```
55,L
56,L 177072  310                          .
57,L 177073  046 006 036 112                      LBA                         SET OPTIONS - DO BLANKS & WAIT ON DISPLAY
58,L 177077  066 352 056 375                      DE      74<8+BL-5           WRAPPED AROUND, USING 12K CARDS
59,L 177103  106 341 374                          HL      MTMSG64
60,L 177106  046 244                              CALL    DISPLAY             SO CHANGE THE MESSAGE
61,L                                              LE      SMST+SMSTL+UMSTL    AND START MEMORY SIZE TEST FROM THERE
62,L 177110  364                          .
63,L                                      MTSIZ32S LLE                        MOVE SIZE START TO L
64,L 177111                               *
65,L 177111  174 060                      MTSIZLP
66,L 177113  174 070                              POP     DE                  GET MEMORY POINTER
67,L 177115  070                                  PUSH    DE                  AND KEEP IT THERE
68,L 177116  066 357 056 375                      PUSH    HL                  SAVE THE TABLE POINTER
69,L 177122  250                                  HL      MTINC+1
70,L 177123  026 002                              XRA                         (CLEAR CARRY)
71,L 177125  016 060                              LC      2
72,L 177127  062 041                              LB      '0'                 UPDATE THE 'K' COUNTER IN FOUR'S
73,L 177131  100 140 376                          DFSB
74,L 177134  006 040                              JFC     MTSIZLP0            WILL NOT GET CARRY UNLESS CROSS 100K
75,L 177136  174 370                              LA      ' '
76,L                                              LMA     DE                  DONE FOR COSMETIC REASONS ONLY
77,L 177140  060                          .
78,L 177141  056 363                      MTSIZLP0 POP      HL
79,L 177143  035                                  LH      SMST>8              (SAFETY, DON'T TRUST MEMORY)
80,L 177144  176 074 233                          DECP    HL                  TRY THE NEXT STL ENTRY
81,L 177147  140 173 376                          CPL     SMST+6              MINIMAL SIZE AT 24K
82,L                                              JTC     MTSIZIT
83,L 177152  026 001                      .
84,L 177154  077                                  LC      1
85,L                                              STL
86,L 177155  016 377                      .
87,L 177157  321                                  LB      0377                LOAD NON-ZERO IN SOME MEMORY
88,L 177160  111 126 000                          LCB
89,L 177163  111 124 000                          DPS     BC,0                TO SEE IF IT IS THERE
90,L 177166  062 261                              DPL     BC,0                   (X-REG IS ZERO FROM ABOVE)
91,L 177170  150 111 376                          ORBC                        IF ZERO, NOT FOUND ANY YET
92,L                                              JTZ     MTSIZLP
93,L 177173  015                          .
94,L 177174  306                          MTSIZIT INCP      HL                  CORRECT POINTER TO FIRST MISSING SECTOR
95,L 177175  046 201 036 377                      LAL
96,L 177201  066 001 056 357                      DE      MERX
97,L 177205  027                                  HL      SVMEMP+1            SET THE VECTOR FOR THE MEMORY TEST
98,L 177206  060                                  DS      DE,HL
99,L 177207  174 060                              POP                         DON'T NEED THE DISPLAY POINTER
                                                  POP     DE                  RESTORE INCREMENT SIZE
```

```
100,L
101,L 177211                              +
102,L                                     MTMAINLP
103,L                                     .
104,L                                     . MAIN LOOP OF THE MEMORY TEST EXECUTED PER MEMORY BLOCK
105,L                                     .
106,L                                     . HERE, A = ADDRESS+1 OF LAST ENTRY IN STL TABLE THAT SHOWS MEMORY
107,L                                     .
                                          .
108,L 177211  024 004                     MTMNBLK   SU      MAXSTLS          BACK UP STL ENTRIES FOR NEXT BLOCK OF TABL
109,L                                     .                                  IF USED MAXSTLS=1 WILL GENERATE AN OVERLAP
110,L 177213  074 227                               CP      UMST             REACHED THE BEGINNING OF THE NORMAL BLOCK?
111,L 177215  100 275 376                           JFC     MTMNXT           NO, JUST ANOTHER SECTION
112,L                                     .
113,L 177220  074 226                               CP      UMST-1           BACKED UP INTO SPECIAL AREA?
114,L 177222  006 227                               LA      UMST             IF BACKUP WOULD CAUSE SKIP OF 12-15K PHYS
115,L 177224  150 275 376                           JTZ     MTMNXT           DO A SPECIAL PASS TO GET IT
116,L                                     .
117,L                                     .                                  REACHED SPECIAL LAST STL SECTION
118,L 177227  174 070                               PUSH    DE               SAVE ADDRESS INCREMENT SIZE
119,L 177231  066 231 056 363                       HL      SMST+MAXSTLS
120,L 177235  026 002                               LC      SMSTL            POINT TO FIRST UNUSED SECTOR
121,L 177237  077                                   STL                      THAT AVAILABLE IN SPECIAL PASS
122,L 177240  250                                   XRA
123,L 177241  310                                   LBA
124,L 177242  320                                   LCA
125,L 177243  330                                   LDA
126,L 177244  340                                   LEA
127,L 177245  056 320                               LH      0150000>8
128,L 177247  360                                   LLA
129,L 177250  076 040                               LX      0170000-0150000>8 COUNT BLOCKS OF 256
130,L
131,L 177252  021                         MTMSP0    BT                       TO BE MOVED TO SAFE PLACE
132,L 177253  117 024 001                           SUX     1
133,L 177256  110 252 376                           JFZ     MTMSP0
134,L
135,L 177261  174 060                               POP     DE               RESTORE DE FROM OLD AREA (SAFER?)
136,L 177263  066 231 056 363                       HL      SMST+MAXSTLS     MOVE SECTOR TABLES FOR SPECIAL AREA
137,L 177267  026 322                               LC      (SEDSPBF>8.AND.0360)+SMSTL
138,L 177271  062 077                               STLOC                    THAT USED IN SPECIAL PASS
139,L 177273  006 225                               LA      SMST
140,L
141,L 177275  076 000                     MTMNXT    LX      0                MOVE THE STL BLOCK # TO THE X-REGISTER
142,L 177277  117 200                               ADAX
143,L 177301  360                                   LLA
144,L 177302  056 363                               LH      SMST>8           POINT TO THE STL TABLE
145,L 177304  026 004                               LC      MAXSTLS
146,L 177306  077                                   STL                      WILL NOW WORK ON THOSE SECTORS
```

```
147,L
148,L 177307  106 102 373                *
                                                 CALL    DOSF60N          SHOULD THE MEMORY TEST BE ENDED?
149,L 177312  160 366 377                        JTS     MTEND            YES, JUST END IT HERE
150,L
                                         .
151,L 177315  250                                XRA                      ** START MEMORY TEST PHASE **
152,L 177316  310                                LBA
153,L 177317  320                                LCA
154,L 177320  066 376 056 077                    HL      MAXSZ-2
155,L 177324  111 027                    MTMNX0  DS      BC,HL            INITIALLY, THE MEMORY MUST BE ZERO
156,L 177326  117 035                            DECP    HL,2
157,L 177330  100 324 376                        JFC     MTMNX0
158,L                                    .
159,L 177333  026 240 016 377                    BC      MERPRF
160,L 177337  066 001 056 357                    HL      SVMEMP+1         SET AN ADDRESS FOR PARITY ERRORS IN MEMORY
161,L 177343  111 027                            DS      BC,HL
162,L 177345  310                                LBA
163,L                                    .
164,L 177346  026 001                    MFNXZ   LC      1
165,L 177350  360                        MFNXB   LLA
166,L 177351  350                                LHA
167,L                                    .
168,L                                    . NOW: HL, B, AND A ARE ZERO
169,L                                    .      C IS INITIALIZED TO FIRST BIT TO TEST
170,L                                    .      X IS STL TABLE POINTER
171,L                                    .      DE IS INCREMENT
172,L                                    .
```

```
173,L
174,L 177352                    +
175,L                           MFWDLP
176,L                           .
177,L                           . ASCENDING THROUGH THE MEMORY PART OF THE MEMORY TEST
178,L                           . NOTE:
179,L                           .      FORWARD ADDRESS SEQUENCE IS:
180,L                           .      0, 1, 2, 3, 4, ... -2, -1.    (ACTUALLY IN 'DE' STEPS)
181,L                           . NOTE:
182,L                           .      IT ALWAYS STARTS AT ZERO AND ENDS WHEN REACHS 'DE' WITH OVERFLOW
183,L                           .
184,L                           . INTERNALLY:
185,L                           . ALPHA    A         WORK REGISTER
186,L                           .          B         EXPECTED DATA
187,L                           .          C         BIT UNDER TEST
188,L                           .          DE        ADDRESS INCREMENT
189,L                           .          HL        CURRENT ADDRESS BEING TESTED
190,L                           .          X         STL SEGMENT BEING TESTED
191,L                           .
192,L                           . BETA     HL        PASS COUNTER
193,L                           .          D         ADDRESS ERROR DISPLAY POINTER (HORIZ POSN)
194,L 177352  307               .          LAM                          GET THE DATA (CAN CREATE PARITY FAULT)
195,L 177353  271                          CPB
196,L 177354  112 237 377                  CFZ       MERONG             WRONG, DATA CHANGED
197,L 177357  252                          XRC                          SET BIT BEING TESTED
198,L 177360  370                          LMA
199,L 177361  277                          CPM                          STILL SET? (PARITY FAULT POSSIBLE)
200,L 177362  112 237 377                  CFZ       MERONG             NO, DATA CHANGED RATHER QUICKLY
201,L                           .
202,L 177365  176 204          MFWDNXT     ADEL                         ERRORS CONTINUE FROM HERE
203,L 177367  115 213                      ACDH                         UPDATE HL USING DE
204,L 177371  115 074 100                  CPH       MAXSZ>8            WRAP AROUND?
205,L 177374  140 352 376                  JTC       MFWDLP             NO, GO BACK FOR MORE
206,L 177377  015                          INCP      HL                 CARRY WRAPAROUND FROM MSB
207,L 177400  115 044 077                  NDH       MAXSZ-1>8          AND KEEP THE ADDRESS IN RANGE
208,L 177403  115 273                      CPDH
209,L 177405  110 352 376                  JFZ       MFWDLP             MSB DOESN'T MATCH STEP SIZE, NOT THE END
210,L 177410  174 276                      CPLE
211,L 177412  110 352 376                  JFZ       MFWDLP             LSB DOESN'T MATCH STEP SIZE, NOT THE END
212,L                           .
213,L 177415  250                          XRA                          RESET THE ADDRESS TO ZERO
214,L 177416  111 252                      XRCB                         SET THE BIT FOR THE NEXT SUB-PASS
215,L 177420  062 202                      ADCC                         SHIFT FOR NEXT BIT (SAFER THAN SHIFT)
216,L 177422  110 350 376                  JFZ       MFNXB              CONTINUE TILL ALL BITS
217,L 177425  153                          EX        CLICK
218,L 177426  111 261                      ORBB
219,L 177430  160 346 376                  JTS       MFNXZ              GO FROM 0 TO 1 AND BACK TO ZERO
220,L                           .                                          (SAFER THAN JFZ)
```

```
221,L
222,L 177433                        +
                                    MTMDON
223,L
224,L                               .
225,L                               . A MAJOR PASS HAS COMPLETED,
226,L                               . CONTINUE TILL ALL MEMORY BLOCKS DONE
227,L                               . THEN INCREMENT PASS COUNTER, SET NEW ADDRESS INCREMENT AND DO NEXT
                                    .
228,L 177433  026 201 016 377             BC      MERX
229,L 177437  066 001 056 357             HL      SVMEMP+1          ANY MEMORY ERROR HERE IS FATAL
230,L 177443  111 027                     DS      BC,HL
231,L                               .
232,L 177445  022 070                     PUSH    XA                GET THE STL TABLE POINTER
233,L 177447  062 060                     POP     BC
234,L 177451  301                         LAB                       FROM THE X-REGISTER
235,L 177452  074 227                     CP      UMST              (JFC SAFER THAN JFZ)
236,L 177454  100 211 376                 JFC     MTMNBLK           CONTINUE TILL ALL BLOCKS DONE
237,L                               .
238,L 177457  174 070                     PUSH    DE                SAVE ADDRESS INCREMENT
239,L 177461  046 224 036 377             DE      MTDONMPF          IGNORE MEMORY FAULTS THAT HAPPEN HERE
240,L 177465  027                         DS      DE,HL
241,L                               .
242,L 177466  250                         XRA                       MOVE THE DISPLAY BUFFER, STACK SAVE AREA
243,L 177467  310                         LBA
244,L 177470  320                         LCA                       INTERRUPT VECTOR AND OTHER NECESSARY INFO
245,L 177471  330                         LDA
246,L 177472  340                         LEA                       BACK TO THE PHYSICAL MEMORY
247,L 177473  056 320                     LH      0150000>8
248,L 177475  360                         LLA                       IT NORMALLY USES
249,L 177476  076 040                     LX      0170000-0150000>8
250,L 177500  021                   MTMD0 BT
251,L 177501  117 024 001               SUX     1                 COUNT THE BLOCKS GOING BY
252,L 177504  110 100 377               JFZ     MTMD0
253,L 177507  106 003 373               CALL    MTSETUP           RESTORE SECTOR TABLE & INTERRUPT VECTOR
254,L 177512  174 060                   POP     DE
255,L                               .
256,L 177514                        MSTART
257,L 177514  026 201 016 377             BC      MERX
258,L 177520  066 001 056 357             HL      SVMEMP+1          ANY MEMORY ERROR HERE IS FATAL
259,L 177524  111 027                     DS      BC,HL
260,L                               .
261,L 177526  020                         BETA
262,L 177527  015                         INCP    HL                UPDATE PASS COUNTER
263,L 177530  315                         LBH
264,L 177531  326                         LCL
265,L 177532  066 006                     LL      6                 DISPLAY THE NEW PASS COUNTER VALUE
266,L 177534  046 005 036 117             DE      79<8+BL-6           (DESTROYS MOST REGISTERS)
267,L 177540  106 160 372                 CALL    DSPOCT
268,L 177543  111 020                     BETAL                     RE-LOAD BETA REGS, RESTORES WHAT DESTROYED
269,L 177545  015                         INCP    HL
270,L 177546  030                         ALPHA
271,L                               .
272,L 177547  303                         LAD
```

```
273,L 177550  174 204                    ADEE
274,L 177552  210                        ACA                          GET NEXT ASCENDING STEP SIZE
275,L 177553  004 300                    AD      MAXSZ-1,XOR,(-1)>8   GENERATE CORRECT NUMBER FOR CARRY AROUND
276,L 177555  174 014 000                ACE     0
277,L 177560  044 077                    ND      MAXSZ-1>8
278,L 177562  330                        LDA
279,L 177563  350                        LHA
280,L 177564  364                        LLE                          DOUBLE CHECK THAT ONLY ONE BIT SET
281,L 177565  035                        DECP    HL                   HL SHOULD BECOME (POWER OF 2) - 1
282,L 177566  140 201 377                JTC     MERX                 CATCH DE BECOMING ZERO AS A FATAL ERROR
283,L 177571  176 244                    NDEL
284,L 177573  115 243                    NDDH                         SHOULD RESULT IN ZERO IF WAS POWER OF 2
285,L 177575  035                        DECP     HL                  SHOULD SET CARRY IF WAS ZERO
286,L 177576  140 007 376                JTC      MTSIZE              OK, RE-CALCULATE MEMORY SIZE FOR NEXT PASS
287,L                                    .                            ** ERROR, WAS NOT POWER OF 2 **
```

```
                                *
                                . ERROR PROCESSING ROUTINES
                                .
177601  151                     MERX    EX      BEEP            EXCEPTIONAL MEMORY ERROR THAT DESTROYED
                                .                               OPERATION OF THE MEMORY TEST HAPPENED
177602  106 003 373                     CALL    MTSETUP
177605  106 242 372                     CALL    SSTATE          RESTORE STATE TO GOOD VALUES (I HOPE)
177610  066 252                         LL      MERXMSG         POINT TO MESSAGE (WITH OTHER MESSAGES)
177612  104 055 362                     JMP     ERROR
                                *
177615  174 060                 MTSIZMPF POP    DE              SKIP P-COUNTER
177617  174 060                         POP     DE              AND ERROR LOCATION
177621  104 173 376                     JMP     MTSIZIT         BUT KEEP TRYING
                                *
177624  302                     MTDONMPF LAC                    INTO A SAFE PLACE
177625  062 060                         POP     BC              GET WASTED INFO OUT OF THE WAY
177627  062 060                         POP     BC
177631  320                             LCA
177632  250                             XRA
177633  310                             LBA
177634  104 100 377                     JMP     MTMD0           AND TRY TO RECOVER
                                *
                                . ERROR IN DATA OR A PARITY FAULT WHILE ACTUALLY DOING MEMORY TEST OPERATIONS
                                .
177637  070                     MERONG  PUSH                    WRONG DATA, CORRECT STACK TO MATCH PARITY
                                .
177640  070                     MERPRF  PUSH    HL              COPY REAL ADDRESS OVER TO BETA REGS
177641  022 070                         PUSH    XA              AND THE STL TABLE POINTER
177643  020                             BETA
177644  062 060                         POP     BC
177646  301                             LAB                     NOW HAVE TABLE POINTER (FROM ALPHA = X)
177647  062 060                         POP     BC              AND ADDRESS TO CALCULATE PHYSICAL ADDRESS
177651  070                             PUSH    HL              SAVE PASS COUNTER
177652  056 363                         LH      SMST>8
177654  360                             LLA                     GET STL TABLE POINTER
                                .
177655  301                             LAB
177656  044 360                         ND      0360            GET STL INDEX BITS
177660  111 250                         XRAB                    CLEAR THEM OUT OF THE ADDRESS
177662  012 012 012 012                 SRN     4
177666  074 004                         CP      MAXSTLS
177670  100 201 377                     JFC     MERX            GOT ERROR OUTSIDE OF MEMORY BEING TESTED?
                                .
177673  017                             INCP    HL,A
177674  307                             LAM                     GET STL ENTRY THAT IN ERROR AREA
177675  044 360                         ND      0360
177677  111 260                         ORAB                    GENERATED PHYSICAL ADDRESS (LS15 BITS)
177701  307                             LAM
177702  002                             SLC
177703  044 003                         ND      3               GET BITS 16 & 17 OF PHYSICAL ADDRESS
177705  004 060                         AD      '0'
177707  022 070                         PUSH    XA
```

```
340,L 177711  046 013                 LE      BL
341,L 177713  066 005                 LL      5               OUTPUT 5 LS DIGITS OF THE PHYSICAL ADDRESS
342,L 177715  106 160 372             CALL    DSPOCT
343,L 177720  022 060                 POP     XA
344,L 177722  370                     LMA                     AND THE MS DIGIT ALSO
345,L                         .
346,L 177723  113 004 015             ADD     5+8             5 FOR DSPOCT & 8 FOR NEXT POSN
347,L 177726  113 074 107             CPD     80-8-1          GO TO NEXT DISPLAY POSITION
348,L 177731  140 347 377             JTC     MERR1
349.L                         .
350,L 177734  106 102 373 MERR0       CALL     DOSF60N        WAIT FOR THE DISPLAY KEY TO BE UP
351,L 177737  170 334 377             JTP     MERR0
352,L 177742  106 345 373             CALL    DOSF67          ROLL UP
353,L 177745  036 007                 LD      8-1             POSITION FOR NEXT ERROR ADDRESS
354,L                         .
355,L 177747  060 MERR1               POP     HL              RESTORE PASS COUNTER
356,L 177750  062 060                 POP     BC              IGNORE P-COUNTER
357,L 177752  062 060                 POP     BC              DISCARD LOCATION OF FAULT
358,L 177754  030                     ALPHA                   GO BACK TO NORMAL REGISTERS
359,L 177755  115 044 077             NDH     MAXSZ-1>8       AFTER BEING IN MEMORY, MAKE IT IN RANGE
360,L                                                         AS A MINIIMAL ASSURANCE OF ACCURACY
361,L 177760  301                     LAB             .
362,L 177761  252                     XRC
363,L 177762  370                     LMA                     SET MEMORY LOCATION TO WHAT IT SHOULD BE
364,L 177763  104 365 376             JMP     MFWDNXT         AND RETURN FROM THE ERROR
365,L                         .
366,L 177766  106 003 373 MTEND       CALL     MTSETUP        RESTORE THE SECTOR TABLE TO NORMAL STATE
367,L 177771  104 127 367             JMP     DEBUG
181.                          .
```

- END OF 1800 MACRO ROM -

```
184,
185,                                    .
186,                                            IFNE    $>8,0177777>8    (CHECK IF LOADED TO LAST PAGE)
187,                                            ERR     MACRO ROM NOT CORRECT SIZE, TOO BIG OR TOO SMALL
188,                                            XIF
189,                                    .
190,                                    . PAD TO THE END OF THE PAGE WITH ZEROS AND PUT THE VERSION OF THE MACRO
191,                                    .       ROM IN THE LAST BYTE IN THE PAGE AND
192,                                    .       THE ROM TYPE IN THE SECOND LAST BYTE
193,                                            LIST    -G
194,                                            RPT     -$-4,AND,255
195,  177774  000                               DC      000
196,  177775  000 000                           DA      *ROMCRC
197,  177777  011                     $MACVRS*  DC      $MACROM
198,                                    .
199.  170000                                    END     MACROM
```

```
177777  $MACVRS         *197
000215  $NL             *653:K   787:K
000203  $NS             *649:K   220:F   221:F   223:F   224:F   225:F   227:F   229:F   231:F   233:F   234:F   239:F
                         709:K   788:K
000233  $O              *658:K   711:K   792:K
000200  $OI             *660:K   450:K
170176  $ONLINE         *109:F
000100  $OS             *661:K   684:K
000040  $OW             *662:K   684:K   725:K   850:K
000024  $RD             *646:K   794:K
170170  $READ           *107:F
170201  $RESTORE        *110:F
173246  $RESTORL        *342:I   348:I
173233  $RESTORX        *333:I    24:I
000023  $RU             *645:K   793:K
156605  $SAVBC          *109      69:I    96:I   150:I   158:I   283:I   308:I
156603  $SAVXA          *108      68:I    97:I   281:I   282:I   309:I   310:I
156620  $SECTOR         *116     305:I   422:I   431:I
170162  $SEEK           *105:F
172637  $SEEKLOP         *45:I    36:I    49:I
172620  $SEEKOK          *30:I    22:I
172633  $SEEKOUT         *40:I   336:I
170215  $STATUS         *115:F
170165  $STEP           *106:F
172667  $STEPWTL         *77:I    90:I
172722  $STEPXIT         *96:I    86:I
156621  $TRACK          *117     136:H    20:I    34:I   334:I
000013  $V              *641:K   784:K
000213  $VA             *652:K    38:H   786:K
170173  $WRITE          *108:F
000000  A$L               *23:E
000000  A$N               *22:E
000140  A$RFC            *30:E
000150  A$RFE            *32:E
000170  A$RFP            *34:E
000160  A$RFS            *33:E
000150  A$RFZ            *31:E
000100  A$RTC            *25:E
000110  A$RTE            *27:E
000130  A$RTP            *29:E
000120  A$RTS            *28:E
000110  A$RTZ            *26:E
174610  ADRDE1          *669:J   667:J
174600  ADRDEV          *659:J   166:J   176:J   177:J
000000  ASCII            *23      52:J   114:J   197:J   200:J   342:J
170561  AVIOLM          *230:F   313:F
171020  AVIOLT          *312:F   179:F   180:F
000001  B0                *6:C    17:C    37:C    59:C    72:C    88:C   100:C   111:C    86:D    97:D   111:D   127:D
                         167:D
000002  B1                *7:C    18:C    38:C    60:C    73:C   101:C   112:C    87:D    98:D   112:D   128:D   168:D
000004  B2                *8:C    39:C    61:C    74:C   102:C   113:C    88:D    99:D   113:D   129:D   169:D
000010  B3                *9:C    21:C    40:C    51:C    75:C    91:C   103:C   114:C    89:D   100:D   114:D   130:D
```

```
000020  B4          *10:C   22:C   30:C   52:C   63:C   76:C  115:C   90:D  101:D  115:D  131:D
000040  B5          *11:C   31:C   53:C   64:C   77:C  116:C   91:D  102:D  116:D  662:K
000100  B6          *12:C   54:C   78:C  117:C   92:D  103:D  117:D  174:D  661:K  763:K
000200  B7          *13:C   44:C   66:C   79:C  118:C   93:D  104:D  118:D  176:D  660:K  749:K
174131  BACKSP      *311:J  127:J
174554  BASED       *641:J  632:J
174553  BASSES      *640:J  635:J
174533  BASSET      *627:J  214:J  266:J
170223  BEEPCODE    *141:F  202:F
170227  BEEPDO1     *147:F  148:F
170243  BEEPDO2     *151:F  153:F
170245  BEEPDO3     *152:F  152:F
000013  BL           *40    337:F  130:H   80:J   88:J   98:J  106:J  205:K  219:K  311:K  314:K  413:K
                    619:K   31:L   57:L  266:L  340:L
176114  BLANKIT     *435:K  343:K  485:K
176116  BLANKITB    *447:K  408:K
175644  BLINK       *256:K   85:F   92:K  127:K
175621  BLINKDE     *241:K   84:F  150:K
175722  BLINKOUT    *300:K  292:K
174466  BPCLR       *600:J  135:J
174472  BPCLRL      *605:J  612:J  614:J
174404  BPSET       *570:J  171:J
174412  BPSETG      *576:J  574:J
174425  BPSRCH      *580:J  586:J
153651  BPTABE       *79     17:J  582:J  605:J 1054:J
153613  BPTABL       *78     15:J  579:J  604:J 1050:J
175466  BPTCLL     *1052:J 1055:J
175460  BPTCLR     *1046:J  606:J
000003  BPTES        *77     78    15:J  579:J  580:J
000012  BPTLN        *72     78
174462  BPTSER     *597:J  578:J  583:J
173463  BRKPNL      *16:J   23:J   27:J
173520  BRKPNN      *36:J  344:F   18:J
173445  BRKPNR       *6:J  187:F  188:F
000010  BSP        *123:C  353:F  356:F  126:J
174112  C12345     *294:J   32:H  649:J   15:L
174124  C1234P     *302:J  299:J
174760  CALL       *765:J  172:J
175001  CALLKA     *784:J  781:J
000030  CAN        *125:C  355:F  124:J
000033  CAN3600    *126:C
175521  CCTAB60    *110:K   96:K   98:K
171207  CHARLOD    *411:F   77:F
171222  CHRCOL     *444:F  459:F
171223  CHRGO      *445:F  439:F
171207  CHRLOP     *434:F  467:F
171230  CHRROW     *448:F  454:F
170254  CLICKROM   *160:F   45:F
170252  CLIKCODE   *159:F  206:F
173742  CMDTNS     *161:J  143:J
174026  CMDTS      *241:J  140:J
000052  CODBP      *134:C
```

```
000061  CODMIN          *132:C
000071  CODMOUT         *131:C
000127  CODWRT          *133:C
153600  CURADR           *62     342:F   135:H   137:H    36:J    63:J    81:J   112:J   395:J   400:J   408:J   410:J
                         424:J   431:J   436:J   451:J   475:J   567:J   575:J   621:J   641:J   650:J   680:J   783:J
                         788:J
000010  CURINBUF         *91:C   276:K   290:K   572:K   579:K
000001  CURONOFF         *88:C   237:K   271:K   514:K
153602  CUROUT           *63     723:J   727:J
176237  CURSES          *553:K    83:F   253:K   688:K
176225  CURSESRS        *540:K   335:K   399:K   430:K   477:K   511:K
000000  CURSOR          *122:C   405:F
153603  CURSTA           *64     679:J
170110  DSBLINK          *85:F
170105  DSBLNKDE         *84:F
170077  DSCBL            *82:F
170102  DSCURSES         *83:F
173527  DEBUG            *40:J   617:F   789:J   367:L
173533  DEBUGI           *48:J   647:F
174257  DECADR          *427:J   173:J
174271  DECADV          *434:J   432:J
000177  DEL             *127:C
176647  DISPBEEP        *811:K   797:K
176651  DISPCLIK        *814:K   798:K
176352  DISPCPRP        *691:K   777:K
176353  DISPCPSL        *692:K   735:K   774:K
176447  DISPDOCC        *734:K   726:K
176337  DISPDOS         *684:K    80:F   131:H   161:H    33:L
176435  DISPOSPW        *728:K   732:K
176665  DISPENDL        *831:K   799:K
176670  DISPEXIT        *832:K   800:K
176413  DISPFCLP        *715:K   722:K
176462  DISPFORC        *743:K   706:K
176554  DISPFTAB        *779:K   714:K
176407  DISPFUNC        *714:K   708:K
176640  DISPGNSA        *803:K   788:K
176642  DISPGOPT        *806:K   792:K
176653  DISPHADJ        *817:K   785:K
176654  DISPHCOL        *819:K   783:K
176357  DISPLANC        *695:K   772:K
176341  DISPLAY         *686:K    79:F   341:F   820:J   851:K    59:L
175732  DISPNEWL        *309:K   787:K   831:K
176546  DISPOFFS        *776:K   755:K
176465  DISPOUT         *748:K   710:K   712:K   718:K
176525  DISPSKIP        *767:K   764:K
176660  DISPVADJ        *824:K   786:K
176661  DISPVCOL        *826:K   784:K
176524  DISPWRIT        *766:K   760:K
170113  DOSFNC60         *87:F
170116  DOSFNC61         *88:F
170121  DOSFNC62         *89:F
170124  DOSFNC63         *90:F
```

```
170127  DO$FNC64        *91:F
170132  DO$FNC65        *92:F
170135  DO$FNC66        *93:F
170140  DO$FNC67        *94:F
170143  DO$FNC68        *95:F
170146  DO$FNC69        *96:F
170151  DO$FNC6A        *97:F
170154  DO$FNC6B        *98:F
170157  DO$FNC6C        *99:F
173726  DOCMDL          *147:J   142:J
154000  DOSBUF           *37:B    97
156000  DOSEXT           *35:B    36:B    99      103
175477  DOSF60           *79:K    87:F
175502  DOSF60N          *93:K    53:G    73:H    144:H   148:L   350:L
175525  DOSF61          *115:K    88:F   811:J
176144  DOSF610         *465:K    97:F   794:K
176204  DOSF611         *498:K    98:F
176207  DOSF611N        *512:K   860:K
176216  DOSF612         *518:K    99:F
175540  DOSF62          *132:K    89:F   854:K
175543  DOSF62N         *151:K    74:F
175571  DOSF63          *165:K    90:F   858:K
175577  DOSF64          *195:K    91:F   378:K   789:K
175604  DOSF65          *209:K    92:F   790:K
175611  DOSF66          *223:K    93:F   853:K
175745  DOSF67          *317:K    94:F   793:K   352:L
176046  DOSF68          *381:K    95:F   796:K
176053  DOSF68LP        *402:K   809:J   414:K
176105  DOSF69          *418:K    96:F   795:K
000006  DSKWRT          *179:I   199:I
000007  DSKWV           *178:I
000012  DSPBLEN         *854:J   852:K
175141  DSPBLNK         *849:J   854:J   849:K
173533  DSPCAD           *59:J    38:J   100:J
176312  DSPCBL          *598:K    82:F    90:J   876:J   189:K   284:K   336:K   402:K   431:K   478:K   583:K   754:K
                         35:L
176004  DSPINIT         *356:K    78:F   394:F    34:H   805:J   813:J    17:L
176016  DSPINITL        *365:K   369:K
175154  DSPOC6          *871:J    83:J    99:J
175170  DSPOCL          *879:J   889:J
175156  DSPOCR          *872:J    95:J
175160  DSPOCT          *874:J   833:J   267:L   342:L
156662  DSPREGSA        *134     546:K   594:K
176303  DSPRESTR        *594:K   550:K
000015  ENTER           *124:C   357:F   122:J
171055  ERROR           *332:F    60:F   269:F   292:F   302:F   306:F   310:F   314:F   318:F   322:F   296:L
170414  ERRORM          *213:F   241:F
174740  EXECDM          *755:J   752:J
174730  EXECEM          *751:J   736:J   743:J
174721  EXECUI          *738:J   253:J
174726  EXECUT          *745:J   174:J
174714  EXEUSR          *731:J   225:J
```

```
172354  FALIGN          *62:H
172363  FALOOP          *70:H    74:H
172431  FAUTO           *98:H    52:H   127:H
172433  FAUTO0         *104:H   125:H
172435  FAUTO1         *110:H   117:H   120:H
172476  FAUTOER        *129:H   112:H
172572  FAUTOMSG       *167:H   129:H
172525  FCHECKR        *140:H    84:H    93:H    95:H   111:H   145:H
000040  FCINDP          *66:D
000020  FCINDT          *65:D
000000  FCINST          *64:D
000060  FCLEAR          *67:D   384:I
000260  FCOINT          *75:D
000220  FCOMOD          *73:D    74:I   127:I   366:I
000160  FCOSYL          *71:D
000140  FCOSYM          *70:D
000100  FCOTDP          *68:D
000120  FCOTUP          *69:D
000240  FCOUTC          *74:D
000300  FCRHDR          *76:D
000320  FCRTIM          *77:D
000200  FCWDCG          *72:D
000360  FCWDGP          *79:D
000340  FCWRTN          *78:D
000010  FDDBL          *192:D
172205  FDIAG            *6:H   184:J
172213  FDIAGDR         *36:H    42:H    46:H    59:H
172512  FDIAGEX        *134:H   146:H   149:H
000010  FDIND          *183:D   224:I
000000  FDINS          *182:D
000004  FDLTD          *180:D
000070  FDMASK         *195:D
000040  FDNINV         *194:D
000020  FDNOT          *193:D
000050  FDOTD          *187:D   189:I
000040  FDOTS          *186:D
000070  FDOUT          *189:D
000003  FDSMSK         *191:D
000030  FDVRD          *185:D   204:I
000020  FDVRS          *184:D
000060  FDWPI          *188:D   185:I
000010  FFDBL          *204:D   194:I   279:I
172401  FFIND           *78:H    56:H    85:H
000070  FFMASK         *209:D
000020  FFRDLTD        *200:D
000000  FFREAD         *199:D    70:H   279:I
000040  FFRMSK         *210:D
000000  FFSGL          *203:D
000010  FFSMSK         *212:D
000070  FFSYNC         *207:D
000060  FFWDCG         *206:D
000020  FFWMSK          *211:D
```

```
000040  FFWRITE         *201:D   194:I
172557  FGETRSP         *160:H    37:H     48:H
000017  FIADR            *135:D
000060  FIINDX           *138:D
000160  FINUM            *136:D
000160  FIPNTR           *142:D
000100  FISTEP           *139:D
000140  FITRER           *141:D
000120  FITROK           *140:D
000001  FKINDX           *111:D   120:D
000100  FKLOFF           *117:D   385:I
000200  FKLON            *118:D
000077  FKMAST           *120:D   385:I
000020  FKPNTR           *115:D   120:D
000040  FKRWMF           *116:D   120:D
000002  FKSTEP           *112:D   120:D
000010  FKTRER           *114:D   120:D
000004  FKTROK           *113:D   120:D
000001  FMINDX           *127:D
000020  FMPNTR           *131:D
000002  FMSTEP           *128:D
000010  FMTRER           *130:D
000004  FMTROK           *129:D
000200  FO43GT           *104:D   106:D   107:D
000100  FO43LE           *103:D   107:D
177524  FO43MGC          *106:D
000300  FO43MSK          *107:D
000001  FODR0             *97:D   622:F   434:I
000002  FODR1             *98:D   624:F   435:I
000004  FOLOAD            *99:D
000020  FOMVIN           *101:D    43:I
000040  FOMVOT           *102:D    35:I   342:I
000010  FOUNLD           *100:D
000400  FPLDTA           *154:D   157:D
000016  FPLHDR           *150:D
000406  FPLRDTA          *157:D   160:D
000424  FPLWDBL          *160:D   163:D
000430  FPLWSGL          *163:D
000200  FPTDTA           *153:D   156:D
000000  FPTHDR           *149:D   151:D
000176  FPTRDTA          *156:D   159:D
000002  FPTRKH           *151:D
000162  FPTWDBL          *159:D   162:D
000156  FPTWSGL          *162:D
000002  FRBUSY           *168:D   302:I
000001  FRDRV            *167:D
000004  FRINDX           *169:D   306:I
000100  FRIXCT           *174:D
000200  FRIXCX           *176:D
172413  FSEEK             *87:H    55:H
172414  FSEEK1            *92:H    96:H
000010  FSFPRO            *89:D   157:I
```

```
000020  FSGAP          *90:D
000200  FSNSYN         *93:D
000001  FSONLN         *86:D    84:I   119:I   120:I   153:I   297:I
000040  FSSTIP         *91:D    89:I
000002  FSTRIP         *87:D   119:I   120:I
000100  FSTRK0         *92:D    93:I    94:I
000004  FSWDIS         *88:D
173607  GETCMD        *102:J   125:J   316:J
174125  GETCME        *304:J   201:J   210:J   211:J   251:J   255:J   257:J   258:J   263:J   271:J   277:J   278:J
                       280:J   281:J   282:J   283:J   327:J   329:J   562:J   653:J
173616  GETCML        *109:J   309:J   319:J   339:J
174150  GETDIG        *321:J   137:J   139:J   145:J
174623  GETSTA        *678:J   672:J   721:J
171644  HMST          *552:F   552:F   553:F   554:F   555:F   556:F   557:F   558:F   559:F   560:F   561:F   562:F
                       563:F   564:F   565:F   566:F   567:F
175071  IDENT         *815:J   129:J
175115  IDENT0        *830:J   825:J
175130  IDENTM        *838:J   819:J
175131  IDENTS        *843:J   850:J   852:J
000017  IDMIC         *852:J   824:J
000013  IDPROC        *850:J   829:J
174211  INCADR        *391:J   195:J
174223  INCADV        *398:J   396:J
170505  INPARM        *225:F   301:F
170770  INPART        *300:F   173:F   174:F
170606  IVIOLM        *233:F   317:F
171030  IVIOLT        *316:F   181:F   182:F
174773  JUMP          *776:J   196:J
175033  JUMPIT        *794:J   792:J
171165  KBDSPILP      *397:F   402:F
171117  KBDSPINI      *367:F    75:F   265:F   612:F   665:F
176703  KEYCHAR       *839:K   163:H   110:J
176722  KEYCHARW      *854:K   855:K
153606  KEYINF         *66
000000  LC             *41     130:H   206:K   220:K   309:K   411:K
174233  LINK          *403:J   203:J
174241  LINKADR       *409:J   407:J
172115  LOADIPL        *64:G    48:G
170000  MACROM         *26:F   199
000030  MAXLIN         *83:C    99:B   205:K   348:K   368:K   490:K   620:K
000120  MAXPOS         *84:C   337:J   363:K   447:K   615:K   771:K
000032  MAXSECT        *27:D   116:H   119:H
000004  MAXSTLS         *6:L     7:L   108:L   119:L   136:L   145:L   328:L
040000  MAXSZ           *7:L    23:L   154:L   204:L   207:L   275:L   277:L   359:L
000114  MAXTRAK        *26:D   124:H   339:I
170470  MEMPAM        *224:F   279:F   336:F
170731  MEMPAT        *278:F   171:F   172:F
177637  MERONG        *312:L   196:L   200:L
177640  MERPRF        *314:L   159:L
177734  MERR0         *350:L   351:L
177747  MERR1         *355:L   348:L
170641  MERROR        *237:F   220:F   221:F   227:F   231:F   239:F
```

```
177601  MERX            *291:L    95:L   228:L   257:L   282:L   329:L
170652  MERXMSG         *239:F   295:L
177350  MFNXB           *165:L   216:L
177346  MFNXZ           *164:L   219:L
177352  MFWDLP          *174:L   205:L   209:L   211:L
177365  MFWDNXT         *202:L   364:L
170625  MINSTE          *236:F   233:F   234:F
174324  MODAGN          *456:J   133:J
174227  MODIFNXT        *400:J   469:J
174335  MODIFX          *463:J   461:J
174343  MODIFY          *471:J   204:J   454:J
174340  MODINC          *465:J   131:J
153604  MODVAL           *65     462:J   463:J
170533  MPARIT          *227:F   223:F   224:F   225:F
170573  MPROTE          *231:F   229:F
177514  MSTART          *256:L    24:L
170736  MSTPAT          *281:F   276:F
177624  MTDONMPF        *302:L   239:L
177766  MTEND           *366:L   149:L
176756  MTINC            *12:L    68:L
177211  MTMAINLP        *101:L
177500  MTMD0           *250:L   252:L   308:L
177433  MTMDON          *222:L
177211  MTMNBLK         *108:L   236:L
177324  MTMNX0          *155:L   157:L
177275  MTMNXT          *141:L   111:L   115:L
176743  MTMSG128         *10:L    32:L
176752  MTMSG64          *11:L    58:L
177252  MTMSP0          *131:L   133:L
167534  MTRFLG           *71:B
167535  MTRLEN           *72:B
167536  MTRPNT           *73:B
175403  MTSETUP         *994:J  1027:J   253:L   293:L   366:L
175405  MTSETX         *1000:J    39:F
177110  MTSIZ32S         *62:L    54:L
177007  MTSIZE           *26:L   286:L
177173  MTSIZIT          *93:L    81:L   300:L
177111  MTSIZLP          *64:L    91:L
177140  MTSIZLP0         *77:L    73:L
177615  MTSIZMPF        *298:L    43:L
167531  MTTFLG           *68:B
167532  MTTLEN           *69:B
167533  MTTPNT           *70:B
000106  NECP             *43     106:J   158:J   315:J   417:J   452:J   460:J   774:J
174247  NEWADR          *413:J   123:J
174253  NOSADR          *424:J   598:J
153611  OLDREGS          *69     157:J   938:J
153607  OLDTOS           *68     281:F   343:F    12:J   559:J   751:J   761:J   787:J   922:J
170040  ONEMSA           *50:F   183:F   184:F
173020  ONLINXIT        *158:I   155:I
000052  OPCODEBP         *44     577:J   592:J
000000  ORIGINI          *24      71      74      81      64:J   205:J   209:J   259:J   262:J   419:J   485:J  1033:J
```

```
170521  OUTPAM          *226:F   305:F
171000  OUTPAT          *304:F   175:F   176:F
174700  OUTSETUP        *715:J   687:J   691:J   695:J   699:J   703:J   707:J   711:J
174654  OUTXC1          *699:J   229:J
174661  OUTXC2          *703:J   230:J
174666  OUTXC3          *707:J   231:J
174673  OUTXC4          *711:J   227:J
174642  OUTXDA          *691:J   287:J
174635  OUTXST          *687:J   286:J
174647  OUTXWT          *695:J   228:J
000037  PISTL           *569:F   586:F    47:L
000002  PN1800           *18     616:F
171757  POWERDCL        *622:F   628:F
171673  POWERL1         *587:F   602:F
171705  POWERL2         *594:F   597:F
171666  POWERUP         *580:F    28:F
000002  RAMROM           *27      8:F    15:F    22:F
173100  READDATA        *226:I   205:I
174310  REGA            *448:J   245:J
174303  REGB            *443:J   246:J
174302  REGC            *442:J   247:J
174305  REGD            *445:J   248:J
174312  REGDSP          *451:J
174304  REGE            *444:J   249:J
174301  REGF            *441:J   250:J
174307  REGH            *447:J   252:J
174306  REGL            *446:J   256:J
174520  REGMOD          *616:J   222:J
174311  REGX            *449:J   284:J
172002  RESTART         *632:F    30:F
173420  RESTDSK         *462:I   667:F    36:H
172064  RESTIPL           *6:G    54:G    62:G
172067  RESTLOOP         *47:G    59:G
172064  RESTNRIM         *44:G
175344  RESTRLOC        *958:J   759:J   785:J
172033  RESWAIT         *652:F   653:F
000000  ROMCRC           *36     196
000000  ROMTYPE          *31     101:F   121:F   217:F   294:F   619:F   656:F   666:F   147      10:G    22:G    43:G
                          49:G    52:G   110:G   114:G   151     155     159     163     167     171     165:J   168:J
                         175:J   179:J   183:J   186:J   189:J   192:J   213:J   218:J   226:J   233:J   265:J   268:J
                         285:J   289:J   625:J   195
156644  RDWS            *133     446:F   453:F   462:F
175325  RSWAP           *941:J   927:J   929:J   967:J   969:J
175327  RSWL            *947:J   954:J
000307  SACLK            *44:D
167441  SACRC            *46:D
000376  SADTA            *43:D
172576  SADUAL           *45:D
175222  SBCRL3          *895:J   318:J   885:J
153600  SDBGWS           *38:B    59      91     914:J   918:J
000307  SDCLK            *51:D
137604  SDCRC            *53:D
```

```
000373  SDDTA          *50:D
172557  SDDUAL         *52:D
150000  SDSPBUF        *39:B    40:B   361:K
167540  SEACCNT        *84:B
167537  SEACFLG        *83:B
000100  SEATTKY        *117:C
167653  SEBRLS         *117:B   631:J
167644  SECCHAR        *109:B   291:K   376:K   587:K
167642  SECFLAGS       *107:B   236:K   270:K   279:K   288:K   289:K   374:K   513:K   515:K   568:K   571:K   578:K
                        580:K   859:K
167643  SECHIDE        *108:B   295:K   300:K   590:K
167645  SECPOS         *110:B   247:K   282:K   569:K   570:K   584:K   693:K   751:K   768:K
167634  SEDLBOT        *100:B   481:K   487:K   489:K
167556  SEDLN01         *99:B   339:K   345:K   347:K   362:K   368:K
167636  SEDNUB         *101:B   340:K   624:K
167554  SEDNUL          *98:B   482:K
167640  SEDOPTS        *103:B   808:J   372:K   449:K   689:K   703:K   808:K   833:K   835:K
150000  SEDSPBF         *40:B   605:F   650:F   154:H  1004:J   137:L
000001  SEDSPKY        *100:C   645:F   646:F    95:K   731:K
000001  SEFUNC1        *111:C   664:F
000002  SEFUNC2        *112:C
000004  SEFUNC3        *113:C
000010  SEFUNC4        *114:C
000020  SEFUNC5        *115:C
000200  SEINTKY        *118:C
167552  SEKBCH          *93:B   158:K
000002  SEKBDKY        *101:C   645:F    95:K
000010  SEKBDWN        *103:C
000004  SEKBRDY        *102:C   129:K   153:K
167551  SEKBS1          *92:B   644:F    94:K   128:K   152:K   730:K
167550  SEKBS2          *91:B   371:F   643:F   663:F   534:K
156000  SEKTRAN         *36:B   380:F   160:K
167553  SELFREQ         *97:B   274:K
167677  SEPSWSV        *124:B   620:J   770:J   931:J   965:J
000040  SERSTKY        *116:C
167666  SESAVA         *121:B
167676  SESAVAF        *122:B
167655  SESAVB         *119:B    91
167665  SESAVBF        *120:B
167700  SESTACK        *126:B    34:F   608:F  1029:J
175445  SETUP         *1018:J   610:F   655:F    16:L
167652  SEXADR         *116:B   668:J
000307  SMCLK           *58:D
107747  SMCRC           *60:D
000370  SMDTA           *57:D
172552  SMDUAL          *59:D
171625  SMST           *531:F   531:F   532:F   533:F   536:F   537:F   538:F   539:F   540:F   541:F   542:F   543:F
                        544:F   545:F   546:F   547:F   548:F   569:F   571:F   586:F   601:F   604:F   649:F  1003:J
                         47:L    60:L    78:L    80:L   119:L   136:L   139:L   144:L   321:L
000002  SMSTL          *533:F   605:F   650:F  1004:J    60:L   120:L   137:L
000252  SOCLK           *32:D
171272  SPLCS          *470:F   404:F
```

```
170027  SRCLICKR        *142:B
175311  SREGMOD         *932:J   623:J
170022  SRMEMPE         *141:B
170047  SRNEXTAL        *145:B    54:F
170000  SRPOWER         *137:B
170003  SRRSTRT         *138:B
170033  SRSTPE          *143:B
170006  SRSYSMF         *139:B
170044  SRTMOUT         *144:B
175321  SSTAT1          *938:J   934:J
175242  SSTATE          *908:J    58:F   267:F   274:F   278:F   300:F   304:F   308:F   312:F   316:F   320:F   324:F
                         640:F    10:J    46:J   294:L
000020  STA12            *76:C
000040  STA13            *77:C
000100  STA14            *78:C
000200  STA15            *79:C
000001  STA16            *72:C   552:F   553:F   554:F   555:F   556:F   557:F   558:F   559:F   560:F   561:F   562:F
                         563:F   564:F   565:F   566:F   567:F   576:F
174357  STACKD          *555:J   223:J
000000  STACKP           *25     272:J   276:J   520:J
000004  STAE             *74:C   531:F   532:F   536:F   537:F   538:F   539:F   540:F   541:F   542:F   543:F   544:F
                         545:F   546:F   547:F   548:F   552:F   553:F   554:F   555:F   556:F   557:F   558:F   559:F
                         560:F   561:F   562:F   563:F   564:F   565:F   566:F   567:F   575:F   576:F
172733  STATWAIT        *115:I   122:I
174560  STLOAD          *644:J   216:J
000002  STPAR            *73:C
170453  STPARM          *223:F   275:F
170721  STPART          *274:F   191:F   192:F
000010  STWE             *75:C   531:F   532:F   536:F   537:F   538:F   539:F   540:F   541:F   542:F   543:F   544:F
                         545:F   546:F   547:F   548:F   552:F   553:F   554:F   555:F   556:F   557:F   558:F   559:F
                         560:F   561:F   562:F   563:F   564:F   565:F   566:F   567:F   575:F   576:F
167524  SVAMLI          *67:B
167430  SVAVIOL         *51:B
167510  SVBEEP          *62:B
167460  SVBKPNT         *55:B
167516  SVCLIK          *63:B
167502  SVDISKWS        *59:B    77:I   121:I   291:I
167505  SVDISPWS        *60:B   728:K
167406  SVINP           *48:B
167436  SVIVIOL         *52:B
167400  SVMEMP          *47:B    42:F  1011:J    42:L    96:L   160:L   229:L   258:L
167546  SVMSECT         *86:B
167547  SVMTRAK         *87:B
167444  SVONEMS         *53:B
167414  SVOUTP          *49:B
167452  SVSCAL          *54:B
167474  SVSTPAR         *57:B    48:F
167466  SVUAINS         *56:B
167422  SVWVIOL         *50:B
000200  SWALBT          *66:C   622:J
000002  SWBASD          *60:C
000001  SWINTE          *59:C    68:C   735:J   742:J
```

```
175363  SWITZAF         *974:J   925:J
000040  SWRPT            *64:C    68:C
000045  SWSCF            *68:C
000020  SWSTDT            *63:C
000004  SWUSER           *61:C    68:C   735:J   772:J
000010  SXABCL           *40:C
000002  SXADLO           *38:C
000004  SXADSC           *39:C
000001  SXAPND           *37:C
000200  SXAPWR           *44:C
000327  SXCLK            *37:D
147543  SXCRC            *39:D
000040  SXDCRQ           *31:C
000020  SXDPR            *30:C
000374  SXDTA            *36:D
173572  SXDUAL           *38:D
000040  SXMCD            *53:C
000010  SXMCTS           *51:C
000020  SXMDSR           *52:C
000100  SXMRNG           *54:C
000002  SXSDTR           *18:C
000010  SXSNSY           *21:C
000020  SXSRDY           *22:C
000001  SXSRQS           *17:C
170614  SYSCAM          *234:F   325:F
171050  SYSCAT          *324:F   185:F   186:F
157000  SYSCOM           *34:B
167400  SYSESR           *31:B    32:B
167400  SYSIVR           *32:B    44:B   128:B
170674  SYSPAT          *251:F    38:F
170432  SYSPRM          *221:F   268:F
160000  SYSRAM           *33:B
170000  SYSROM           *30:B   134:B
175045  TDSTD           *804:J   802:J
175060  TDWAIT          *811:J   812:J
170044  TIMOUT           *58:F
170414  TIMOUTM         *220:F    59:F
173275  TOFFTL          *371:I    85:I   124:I   154:I
173264  TOFFTLDE        *352:I   623:F   625:F
171616  TRIANGLE        *523:F   398:F
175037  TSTDSP          *797:J   254:J
176760  TSTMEM           *14:L   224:J
171040  UAINST          *320:F   189:F   190:F
170622  UAINTM          *235:F   321:F
171627  UMST            *536:F   549:F  1000:J   110:L   113:L   114:L   235:L
000015  UMSTL           *549:F  1001:J    60:L
171077  UTRANT          *352:F   360:F   362:F   379:F   383:F
000020  UTRANTL         *360:F   378:F   384:F   391:F
170270  VECTI           *167:F   209:F  1010:J
000124  VECTIL          *209:F   259:F  1012:J
171664  WRAPST          *575:F    38:L
170545  WVIOLM          *229:F   309:F
```

```
 171010  WVIOLT          *308!F    177!F    178!F
```