HEATH ZENITH

H19 ROM

H-19-1 595-2465-01

```
                         2  ***     H19 TERMINAL FIRMWARE
                         3  *
                         4  *
                         5  *       COPYRIGHT 1978 BY HEATH COMPANY
                         6  *
                         7  *       WRITTEN BY R. N. BORCHARDT AUGUST 1, 1978


                         9  *       ASCII CHARACTER EQUIVALENCES
                        10  *
000.000                 11  NULL    EQU     00000000B       NULL
000.007                 12  BELL    EQU     00000111B       BELL
000.010                 13  BS      EQU     00001000B       BACKSPACE
000.011                 14  HT      EQU     00001001B       HORIZONTAL TAB
000.012                 15  LF      EQU     00001010B       LINE FEED
000.015                 16  CR      EQU     00001101B       CARRIAGE RETURN
000.021                 17  XON     EQU     00010001B       DC1 (X-ON)
000.023                 18  XOFF    EQU     00010011B       DC3(X-OFF)
000.030                 19  CAN     EQU     00011000B       CANCEL
000.033                 20  ESC     EQU     00011011B       ESCAPE
000.177                 21  RUBOUT  EQU     01111111B       RUBOUT/DELETE


                        23  **      I/O PORT EQUIVALENCES
                        24  *


                        26  *       8250 ACE PORTS
                        27  *
000.100                 28  AP.RBR  EQU     100Q            RECEIVER BUFFER REGISTER PORT
                        29
000.100                 30  AP.THR  EQU     100Q            TRANSMITTER HOLDING REGISTER PORT
                        31
000.101                 32  AP.IER  EQU     101Q            INTERRUPT ENABLE REGISTER PORT
000.001                 33  AB.ERDA EQU     00000001B       ENABLE RECEIVED DATA AVAILABE INTERRUPT
000.002                 34  AB.ETRE EQU     00000010B       ENABLE TRANSMITTER HOLDING REGISTER EMPTY
000.004                 35  AB.ERLS EQU     00000100B       ENABLE RECEIVER LINE STATUS INTERRUPT
000.010                 36  AB.EMS  EQU     00001000B       ENABLE MODEM STATUS INTERRUPT
                        37
000.102                 38  AP.IIR  EQU     102Q
000.004                 39  AB.RDAI EQU     100B            RECEIVED DATA AVAILABE INTERRUPT
000.002                 40  AB.TREI EQU     010B            TRANSMITTER REGISTER EMPTY INTERRUPT
000.001                 41  AB.IIP  EQU     00000001B       INVERTED INTERRUPT PENDING
000.006                 42  AB.IID  EQU     00000110B       INTERRUPT IDENTIFICATION BITS
                        43
000.103                 44  AP.LCR  EQU     103Q            LINE CONTROL REGISTER PORT
000.000                 45  AB.5BW  EQU     00000000B       FIVE BIT WIRD
000.001                 46  AB.6BW  EQU     00000001B       SIX BIT WORD
000.002                 47  AB.7BW  EQU     00000010B       SEVEN BIT WORD
000.003                 48  AB.8BW  EQU     00000011B       EIGHT BIT WORD
000.004                 49  AB.2SB  EQU     00000100B       TWO STOP BITS
```

```
000.010                  50  AB.PEN  EQU     00001000B       PARITY ENABLE
000.020                  51  AB.EPS  EQU     00010000B       EVEN PARITY SELECT
000.040                  52  AB.SP   EQU     00100000B       STICK PARITY
000.100                  53  AB.SBRK EQU     01000000B       SET BREAK
000.200                  54  AB.DLAB EQU     10000000B       DIVISOR LATCH ACCESS BIT
                         55
000.104                  56  AP.MCR  EQU     104Q            MODEM CONTROL REGISTER PORT
000.001                  57  AB.DTR  EQU     00000001B       DATA TERMINAL READY
000.002                  58  AB.RTS  EQU     00000010B       REQUEST TO SEND
000.004                  59  AB.OUT1 EQU     00000100B       OUTPUT #1
000.010                  60  AB.OUT2 EQU     00001000B       OUTPUT #2
000.020                  61  AB.LOOP EQU     00010000B       LOOP
                         62
000.105                  63  AP.LSR  EQU     105Q            LINE STATUS REGISTER PORT
000.001                  64  AB.DR   EQU     00000001B       DATA READY
000.002                  65  AB.OR   EQU     00000010B       OVERRUN ERROR
000.004                  66  AB.PE   EQU     00000100B       PARITY ERROR
000.010                  67  AB.FE   EQU     00001000B       FRAMING ERROR
000.020                  68  AB.BI   EQU     00010000B       BREAK INTERRUPT
000.040                  69  AB.THRE EQU     00100000B       TRANSMITTER HOLDING REGISTER EMPTY
000.100                  70  AB.TSRE EQU     01000000B       TRANSMITTER SHIFT REGISTER EMPTY
                         71
000.106                  72  AP.MSR  EQU     106Q            MODEM STATUS REGISTER PORT
000.001                  73  AB.DCTS EQU     00000001B       DELTA CLEAR TO SEND
000.002                  74  AB.DDSR EQU     00000010B       DELTA DATA SET READY
000.010                  75  AB.DRLS EQU     00001000B       DELTA RECEIVE LINE SIGNAL DETECT
000.020                  76  AB.CTS  EQU     00010000B       CLEAR TO SEND
000.040                  77  AB.DSR  EQU     00100000B       DATA SET READY
000.200                  78  AB.RLSD EQU     10000000B       RECEIVED LINE SIGNAL DETECT
                         79
000.100                  80  AP.DLL  EQU     100Q            DIVISOR LATCH LSB
                         81
000.101                  82  AP.DLM  EQU     101Q            DIVISOR LATCH MSB


                         84  *       CRT VIDEO CONTROLER PORTS
                         85  *
000.140                  86  VP.AR   EQU     140Q            VIDEO ADDRESS REGISTER OUTPUT PORT
000.001                  87  VA.HD   EQU     1               HORIZONTAL DISPLAYED REGISTER
000.002                  88  VA.HSP  EQU     2               HORIZONTAL SYNC POSITION REGISTER
000.003                  89  VA.HSW  EQU     3               HORIZONTAL SYNC WIDTH REGISTER
000.004                  90  VA.VT   EQU     4               VERTICAL TOTAL REGISTER
000.005                  91  VA.VTA  EQU     5               VERTICAL TOTAL ADJUST REGISTER
000.006                  92  VA.VD   EQU     6               VERTICAL DISPLAYED REGISTER
000.007                  93  VA.VSP  EQU     7               VERTICAL SYNC POSITION REGISTER
000.010                  94  VA.IM   EQU     8               INTERLACE MODE REGISTER
000.011                  95  VA.MSLA EQU     9               MAXIMUM SCAN LINE ADDRESS REGISTER
000.012                  96  VA.CS   EQU     10              CURSOR START REGISTER
000.100                  97  VB.CBE  EQU     01000000B       CURSOR BLINK ENABLE
000.040                  98  VB.CBPS EQU     00100000B       CURSOR BLINK PERIOD SLOW
000.040                  99  VB.CND  EQU     00100000B       CURSOR NOT DISPLAYED
000.013                 100  VA.CE   EQU     11              CURSOR END REGISTER
000.014                 101  VA.SAM  EQU     12              START ADDRESS (MSB) REGISTER
000.015                 102  VA.SAL  EQU     13              START ADDRESS (LSB) REGISTER
000.016                 103  VA.CAM  EQU     14              CURSOR ADDRESS (MSB) REGISTER
```

```
000.017                 104  VA.CAL  EQU     15              CURSOR ADDRESS (LSB) REGISTER
000.020                 105  VA.LPM  EQU     16              LIGHT PEN ADDRESS (MSB) REGISTER
000.021                 106  VA.LPL  EQU     17              LIGHT PEN ADDRESS (LSB) REGISTER
                        107
000.141                 108  VP.REGO EQU     141Q            VIDEO REGISTER OUTPUT PORT
                        109
000.143                 110  VP.REGI EQU     143Q            VIDEO REGISTER INPUT PORT
000.010                 111  VB.RBD  EQU     00001000B       REVERSE VIDEO DISABLE (WHEN ADDED TO CRTC PORT#)
000.004                 112  VB.NMI  EQU     00000100B       CAUSE NMI (MUST BE ADDED TO CRTC PORT#)


                        114  *       KEYBOARD PORTS
                        115  *
000.200                 116  KP.1    EQU     200Q            KEYBOARD PORT #1
000.177                 117  KB.CHAR EQU     01111111B       KEY VALUE
000.200                 118  KB.CTL  EQU     10000000B       CONTROL KEY
000.013                 119  KB.EX1  EQU     00001011B       EXTRA ENCODED KEY #1
000.014                 120  KB.EX2  EQU     00001100B       EXTRA ENCODED KEY #2
                        121
000.240                 122  KP.2    EQU     240Q            KEYBOARD PORT #2
000.001                 123  KB.SHFT EQU     00000001B       SHIFT KEY(S)
000.002                 124  KB.CPLK EQU     00000010B       CAPS LOCK KEY
000.004                 125  KB.BRK  EQU     00000100B       BREAK KEY
000.010                 126  KB.ONLN EQU     00001000B       ON-LINE KEY
000.100                 127  KB.RPT  EQU     01000000B       REPEAT KEY
000.200                 128  KB.STB  EQU     10000000B       KEYBOARD STROBE


                        130  *       MISC. I/O PORTS
                        131  *
000.300                 132  MP.TICK EQU     300Q            TICK PORT
                        133
000.340                 134  MP.BELL EQU     340Q            BELL PORT
                        135
000.000                 136  MP.PUP1 EQU     000Q            POWER UP CONFIGURATION PORT #1
000.017                 137  P1.BR   EQU     00001111B       BAUD RATE SELECTION SWITCHES
000.020                 138  P1.PEN  EQU     00010000B       PARITY ENABLE SWITCH
000.040                 139  P1.EPS  EQU     00100000B       EVEN PARITY SELECT SWITCH
000.100                 140  P1.SPS  EQU     01000000B       STICK PARITY SELECT SWITCH
000.200                 141  P1.FDX  EQU     10000000B       FULL DUPLEX SELECT SWITCH
                        142
000.040                 143  MP.PUP2 EQU     040Q            POWER UP CONFIGURATION PORT #2
000.001                 144  P2.CBLK EQU     00000001B       CURSOR = BLOCK
000.002                 145  P2.NOTK EQU     00000010B       NO TICK ON KEYBOARD STRIKES
000.004                 146  P2.WRAP EQU     00000100B       WRAP AROUND TO BEGINNING OF NEXT LINE
000.010                 147  P2.ALF  EQU     00001000B       AUTO LINE FEED ON CARRIAGE RETURN
000.020                 148  P2.NOSC EQU     00010000B       SCROLL KEY = NO SCROLL (EX HOLD SCREEN MODE)
000.040                 149  P2.VT52 EQU     00100000B       VT52 MODE
000.100                 150  P2.KPDS EQU     01000000B       KEYPAD SHIFTED
000.200                 151  P2.50HZ EQU     10000000B       50 HERTZ LINE FREQUENCY
```

```
                        153  **      INSTRUCTION EQUIVALENCES
                        154  *
000.313                 155  I.BITA  EQU     11001011B       BIT B,R (BYTE A)
000.100                 156  I.BITB  EQU     01000000B       BIT B,R (BYTE B) LESS BIT AND REGISTER
000.313                 157  I.BITHA EQU     11001011B       BIT B,(HL)  (BYTE A)
000.106                 158  I.BITHB EQU     01000110B       BIT B,(HL)  (BYTE B)
000.355                 159  I.IM1A  EQU     11101101B       IM1 (BYTE A)
000.126                 160  I.IM1B  EQU     01010110B       IM1 (BYTE B)
000.355                 161  I.NEGA  EQU     11101101B       NEG (BYTE A)
000.104                 162  I.NEGB  EQU     01000100B       NEG (BYTE B)
000.313                 163  I.SETA  EQU     11001011B       SET B,R (BYTE A)
000.300                 164  I.SETB  EQU     11000000B       SET B,R (BYTE B)
000.313                 165  I.SETHA EQU     11001011B       SET B,(HL)  (BYTE A)
000.306                 166  I.SETHB EQU     11000110B       SET B,(HL)  (BYTE B)
000.355                 167  I.SHDA  EQU     11101101B       SBC HL,DE (BYTE A)
000.122                 168  I.SHDB  EQU     01010010B       SBC HL,DE (BYTE B)
000.313                 169  I.RESA  EQU     11001011B       RES B,R (BYTE A)
000.200                 170  I.RESB  EQU     10000000B       RES B,R (BYTE B)
000.313                 171  I.RESHA EQU     11001011B       RES B,(HL)  (BYTE A)
000.206                 172  I.RESHB EQU     10000110B       RES B,(HL)  (BYTE B)
000.010                 173  I.EXAF  EQU     00001000B       EX AF,AF'
000.331                 174  I.EXX   EQU     11011001B       EXX
000.355                 175  I.LDEDA EQU     11101101B       LD DE,(NN) (BYTE A)
000.133                 176  I.LDEDB EQU     01011011B       LD DE,(NN) (BYTE B)
000.355                 177  I.LDDA  EQU     11101101B       LDD (BYTE A)
000.250                 178  I.LDDB  EQU     10101000B       LDD (BYTE B)
000.355                 179  I.LDDRA EQU     11101101B       LDDR (BYTE A)
000.270                 180  I.LDDRB EQU     10111000B       LDDR (BYTE B)
000.355                 181  I.LDIA  EQU     11101101B       LDI (BYTE A)
000.240                 182  I.LDIB  EQU     10100000B       LDI (BYTE B)
000.355                 183  I.LDIRA EQU     11101101B       LDIR (BYTE A)
000.260                 184  I.LDIRB EQU     10110000B       LDIR (BYTE B)
000.323                 185  I.OUT   EQU     11010011B       OUTPUT
000.311                 186  I.RET   EQU     11001001B       RETURN
000.303                 187  I.JMP   EQU     11000011B       JUMP
000.355                 188  I.RETNA EQU     11101101B       RETURN FROM NMI (BYTE A)
000.105                 189  I.RETNB EQU     01000101B       RETURN FROM NMI (BYTE B)


                        191  *       BIT INSTRUCTION BIT AND REGISTER DEFINITIONS
                        192  *
                        193  *       BYTE B OF BIT AND SET OPERATIONS ARE XXBBBRRR.  WHERE XX IS
                        194  *       DEFINED BY THE INSTRUCTION, BBB DEFINES THE BIT, AND RRR DEFINES
                        195  *       THE REGISTER
                        196  *
000.010                 197  IB.CPLK EQU     1*8             CAPS LOCK = BIT 1
000.070                 198  IB.ESCF EQU     7*8             ESCAPE CODE FLAG = BIT 7
000.010                 199  IB.ETRE EQU     1*8             ENABLE TRANSMITTER REGISTER EMPTY INTERRUPT
000.070                 200  IB.IFF  EQU     7*8             INPUT FIFO FLAG = BIT 7
000.070                 201  IB.KCB  EQU     7*8             KEYBOARD CONTROL KEY BIT = BIT 7
000.000                 202  IB.KSB  EQU     0*8             KEYBOARD SHIFT KEY BIT = BIT 0
000.000                 203  IB.HSM  EQU     0*8             HOLD SCREEN MODE = BIT 7
000.070                 204  IB.RV   EQU     7*8             REVERSE VIDEO MODE = BIT 7
000.060                 205  IB.ICM  EQU     6*8             INSERT CHARACTER MODE = BIT 6
000.070                 206  IB.KPDA EQU     7*8             KEYPAD ALTERNATE MODE = BIT 5
```

```
000.060                 207  IB.KPDS EQU     6*8             KEYPAD SHIFTED MODE = BIT 4
000.030                 208  IB.ONLN EQU     3*8             TERMINAL ON LINE FLAG = BIT 3
000.020                 209  IB.BRK  EQU     2*8             BREAK KEY ON = BIT 2
000.010                 210  IB.GRPH EQU     1*8             TERMINAL IN GRAPHICS MODE = BIT 1
000.000                 211  IB.PWE  EQU     0*8             PREVIOUS CHARACTER WAS AN ESCAPE = BIT 0
000.040                 212  IB.XOFF EQU     4*8             XOFF SENT TO HOST = BIT 4
000.010                 213  IB.XMTG EQU     1*8             TRANSMIT MODE = GRAPHICS = BIT 1
000.020                 214  IB.XMTR EQU     2*8             TRANSMIT MODE = REVERSE VIDEO = BIT 2
                        215
000.007                 216  IR.A    EQU     111B            REGISTER = A
000.000                 217  IR.B    EQU     000B            REGISTER = B
000.001                 218  IR.C    EQU     001B            REGISTER = C
000.002                 219  IR.D    EQU     010B            REGISTER = D
000.003                 220  IR.E    EQU     011B            REGISTER = E
000.004                 221  IR.H    EQU     100B            REGISTER = H
000.005                 222  IR.L    EQU     101B            REGISTER = L


                        224  **      LABEL EQUIVALENCES
                        225  *
000.200                 226  IFF     EQU     10000000B       INPUT FIFO FLAG
000.200                 227  ESCF    EQU     10000000B       ESCAPE FLAG
000.037                 228  SCRL    EQU     00011111B       KEYBOARD VALUE FOR SCROLL KEY
```

```
                                231  ***     INITIALIZE SYSTEM
                                232  *
                                233  *       JUMP TO THE ROUTINE WHICH SETS UP ALL PORTS, RAM, AND THE CRTC
                                234
000.000                         235          ORG     0
                                236
000.000  303 001 013            237  START   JMP     INIT
000.003  313                    238          DB      43H+41H+47H             ENGINEERING CODE



                                240  ***     MAIN - MAIN CONTROL LOOP
                                241  *
                                242  *       *MAIN* SETS ITSELF UP AS THE FINAL RETURN ADDRESS FOR ALL ROUTINES
                                243  *       AND IS IN CHARGE OF GIVING CONTROL TO THE PROPER ROUTINE WHENEVER
                                244  *       THERE IS A CHARACTER PRESENT IN ONE OF THE FIFOS
                                245  *
                                246  *       THE ORDER OF PRIORITY OF SERVICE IS IN THE ORDER IN WHICH THE
                                247  *       FIFOS ARE TESTED (KEYBOARD FIFO = #1, INPUT FIFO = #2, OUTPUT
                                248  *       FIFO = #3)
                                249  *
                                250  *
                                251  *       ENTRY   NONE
                                252  *
                                253  *       EXIT    NONE
                                254  *
                                255  *       USES    A,B,C,D,E,H,L,F
                                256
                                257
000.004  041 004 000            258  MAIN    LXI     H,MAIN          SET SELF AS RETURN ADDRESS ON STACK
000.007  345                    259          PUSH    H
                                260
000.010  315 367 011            261          CALL    FCIF            FETCH CHARACTER FROM INPUT FIFO
000.013  322 120 003            262          JNC     IFCP            IF PRESENT, PRECESS IT
                                263
000.016  315 107 012            264  MAIN.N  CALL    FVKF            FETCH VALUES FROM KEYBOARD FIFO
000.021  322 117 001            265          JNC     KCE             IF CHARACTER PRESENT, ENCODE IT
                                266
000.024  315 111 016            267          CALL    UCP             UPDATE CURSOR POSITION
                                268
000.027  072 307 100            269          LDA     MODEA           GET MODE FLAGS
000.032  346 040                270          ANI     MA.BRK          HAS BREAK BEEN SET?
000.034  304 070 000            271          CNZ     AKI             IF BREAK IS SET, SEE IF WE CAN TURN IT OFF
                                272
000.037  072 311 100            273  MAIN1   LDA     MODEI           GET CURRENT MODE FLAGS
000.042  346 010                274          ANI     MI.ONLN         IS TERMINAL ON-LINE?
000.044  310                    275          RZ                      IF NOT ON LINE
                                276
                                277  *       ALTERNATE ENTRY POINT FOR SETTING THE XMIT INTERRUPT ONLY
                                278  *
000.045  363                    279  MAINA   DI                      ELSE, LOCK OUT INTERRUPTS
000.046  072 243 100            280          LDA     OFC             SEE IF OUTPUT FIFO HAS ANY CHARACTERS
000.051  267                    281          ORA     A
000.052  050 012                282          JR      Z,MAIN2         IF NO CHARACTERS TO OUTPUT
                                283
```

```
000.054  333 101      284          IN      AP.IER          IF CHARACTERS PRESENT, SEE IF XMIT INT. IS ON
                       285  *       BIT     IB.ETRE,A
000.056  313 117      286          DB      I.BITA,I.BITB!IB.ETRE!IR.A
000.060  040 004      287          JR      NZ,MAIN2        IF INTERRUPT IS ENABLED
                       288
000.062  366 002      289          ORI     AB.ETRE         ELSE, ENABLE INTERRUPT SO OUTPUT CAN BEGIN
000.064  323 101      290          OUT     AP.IER
000.066  373          291  MAIN2   EI                      ENABLE INTERRUPTS
000.067  311          292          RET
                       293
```

```
                                296  ***     AKI - ACE/KEYBOARD INTERRUPT
                                297  *
                                298  *       *AKI* IS ACCESSED WHENEVER AN INTERRUPT IS RECEIVED FROM EITHER
                                299  *       THE ACE, THE KEYBOARD ENCODER, OR THE KEYBOARD BREAK KEY.  ALL
                                300  *       VALID INTERRUPT SOURCES ARE SAMPLED TO INSURE THAT NO FALSE
                                301  *       INTERRUPTS ARE SERVICED.
                                302  *
                                303  *
                                304  *       ENTRY   NONE
                                305  *
                                306  *       EXIT    NONE
                                307  *
                                308  *       USES    A',B',C',D',E',H',L',F'
                                309
000.000                         310          ERRNZ   *-70Q           Z80 MODE ONE INTERRUPT ADDRESS
                                311  *AKI    EX      AF,AF'          EXCHANGE ALL REGISTERS
000.070  010                    312  AKI     DB      I.EXAF
                                313  *       EXX
000.071  331                    314          DB      I.EXX
                                315
000.072  333 240                316  AKI1    IN      KP.2            READ KEYBOARD PORT #2
000.074  356 366                317          XRI     11110110B       INVERT SWITCHES (EXCEPT OFF-LINE)
000.076  127                    318          MOV     D,A             SAVE SWITCH VALUES
000.077  346 010                319          ANI     KB.ONLN         MASK FOR ON-LINE ACTIVE
000.101  117                    320          MOV     C,A             SAVE RESULT
000.102  072 311 100            321          LDA     MODEI           GET CURRENT MODE
000.105  346 367                322          ANI     377Q-MI.ONLN    CLEAR PREVIOUS ON-LINE STATUS
000.107  261                    323          ORA     C               REPLACE WITH NEW STATUS
000.000                         324          ERRNZ   KB.ONLN-MI.ONLN SWITCH AND MODE FLAG MUST BE THE SAME
000.110  062 311 100            325          STA     MODEI           UPDATE *MODE*
000.113  172                    326          MOV     A,D             GET SWITCH VALUES
000.114  346 004                327          ANI     KB.BRK          MASK FOR BREAK SWITCH
000.116  072 307 100            328          LDA     MODEA           GET MODE FLAGS
000.121  312 264 000            329          JZ      AKI1.5          IF NO BREAK KEY
                                330
000.124  107                    331          MOV     B,A             SAVE MODE A
000.125  072 311 100            332          LDA     MODEI           GET MODE I
000.130  346 004                333          ANI     MI.KID          SEE IF KEYBOARD IS DISABLED
000.132  170                    334          MOV     A,B             (A) = MODE A
000.133  302 264 000            335          JNZ     AKI1.5          IF CAN'T RESPOND TO BREAK
                                336
000.136  366 040                337          ORI     MA.BRK          SET BREAK FLAG
000.140  062 307 100            338          STA     MODEA
000.143  303 254 000            339          JMP     AKI1.3          CONTINUE PAST NMI ROUTINE
                                340
```

```
                                343  ***     NMI - NON MASKABLE INTERRUPT USED FOR CRTC HOME POSITION UPDATE
                                344  *
                                345  *       NMI OUTPUTS THE CURRENT DISPLAY INFORMATION TO THE CRTC WHEN
                                346  *       CAUSED BY VERT SYNC.
                                347  *
                                348  *
                                349  *       ENTRY   NONE
                                350  *
                                351  *       EXIT    NONE
                                352  *
                                353  *       USES    NONE
                                354
     000.000                    355          ERRNZ   *-146A
                                356
     000.146  365               357  NMI     PUSH    PSW             SAVE REGISTERS
     000.147  305               358          PUSH    B
     000.150  325               359          PUSH    D
     000.151  345               360          PUSH    H
                                361
                                362  *       UPDATE VERTICAL DISPLAYED REGISTER
                                363  *
     000.152  076 006           364          MVI     A,VA.VD         SET VERTICAL DISPLAYED ADDRESS
     000.154  323 140           365          OUT     VP.AR
     000.156  072 276 100       366          LDA     VI.VD           GET CURRENT # OF LINES TO DISPLAY
     000.161  323 141           367          OUT     VP.REGO
                                368
                                369  *       UPDATE CURSOR TYPE REGISTERS
                                370  *
     000.163  052 277 100       371          LHLD    VI.CSE          GET CURSOR START AND END PARAMETERS
     000.166  076 012           372          MVI     A,VA.CS         SET CRTC CURSOR START ADDRESS
     000.170  323 140           373          OUT     VP.AR
     000.172  174               374          MOV     A,H             OUTPUT CURSOR START INFO
     000.173  323 141           375          OUT     VP.REGO
     000.175  076 013           376          MVI     A,VA.CE         SET CURSOR END ADDRESS
     000.177  323 140           377          OUT     VP.AR
     000.201  175               378          MOV     A,L             OUTPUT CURSOR END INFO
     000.202  323 141           379          OUT     VP.REGO
                                380
                                381  *       UPDATE VIDEO HOME ADDRESS
                                382  *
     000.204  052 301 100       383          LHLD    VI.SA           GET START ADDRESS
     000.207  076 014           384          MVI     A,VA.SAM        START ADDRESS MSB
     000.211  323 140           385          OUT     VP.AR
     000.213  174               386          MOV     A,H             OUTPUT ADDRESS MSB
     000.214  323 141           387          OUT     VP.REGO
     000.216  076 015           388          MVI     A,VA.SAL        START ADDRESS LSB
     000.220  323 140           389          OUT     VP.AR
     000.222  175               390          MOV     A,L             OUTPUT ADDRESS LSB
     000.223  323 141           391          OUT     VP.REGO
                                392
                                393  *       UPDATE CURSOR ADDRESS
                                394  *
     000.225  052 303 100       395          LHLD    VI.CA           GET CURSOR ADDRESS
     000.230  076 016           396          MVI     A,VA,CAM        CURSOR ADDRESS MSB
     000.232  323 140           397          OUT     VP.AR
     000.234  174               398          MOV     A,H             CURSOR MSB
```

```
000.235  323 141        399          OUT     VP.REGO
000.237  076 017        400          MVI     A,VA.CAL        CURSOR ADDRESS LSB
000.241  323 140        401          OUT     VP.AR
000.243  175            402          MOV     A,L             CURSOR LSB
000.244  323 141        403          OUT     VP.REGO
                        404
                        405  *       ALLOW NEXT NMI
                        406  *
000.246  341            407          POP     H               RESTORE ALL REGISTERS
000.247  321            408          POP     D
000.250  301            409          POP     B
000.251  361            410          POP     PSW
                        411
                        412  *       RETN                    RETURN FROM NMI
000.252  355 105        413          DB      I.RETNA,I.RETNB


000.254  333 103        415  AKI1.3  IN      AP.LCR          SET BREAK SIGNAL
000.256  366 100        416          ORI     AB.SBRK
000.260  323 103        417          OUT     AP.LCR
000.262  030 013        418          JR      AKI1.7          CONTINUE
                        419
000.264  346 337        420  AKI1.5  ANI     377Q-MA.BRK     MAKE SURE BREAK FLAG IS OFF
000.266  062 307 100    421          STA     MODEA
000.271  333 103        422          IN      AP.LCR          CLEAR ANY BREAK
000.273  346 277        423          ANI     377Q-AB.SBRK
000.275  323 103        424          OUT     AP.LCR
                        425
000.277  333 102        426  AKI1.7  IN      AP.IIR          SEE IF ACE WAS SOURCE OF INTERRUPT
000.301  376 004        427          CPI     AB.RDAI         SEE IF THERE IS RECEIVED DATA AVAILABLE
000.303  040 055        428          JR      NZ,AKI2         IF NOT A DATA AVAILABLE INTERRUPT
                        429
                        430  *       INTERRUPT CAUSED BY DATA AVAILABLE IN ACE RECEIVER BUFFER
                        431  *
000.305  333 100        432          IN      AP.RBR          INPUT DATA
000.307  346 177        433          ANI     01111111B       TOSS PARITY BIT
000.311  312 113 001    434          JZ      AKI6            IF CHARACTER WAS A NULL, EXIT
                        435
                        436
                        437  *       BIT     IB.ONLN,C       SEE IF TERMINAL IS ON-LINE
000.314  313 131        438          DB      I.BITA,I.BITB!IB.ONLN!IR.C
000.316  050 173        439          JR      Z,AKI6          IF OFF LINE, EXIT
                        440
000.320  127            441          MOV     D,A             SAVE INPUT CHARACTER
000.321  072 241 100    442          LDA     IFC             GET INPUT FIFO COUNTER
000.324  376 160        443          CPI     IFMAX-16        SEE IF WITHIN 16 BYTES OF OVERFLOW
000.326  070 022        444          JR      C,AKI1.8        IF NOT < 16 BYTES LEFT
                        445
000.330  333 105        446  AKI1.75 IN      AP.LSR          SEE IF UART CAN TAKE A CHARACTER TO OUTPUT
000.332  346 040        447          ANI     AB,THRE
000.334  050 372        448          JR      Z,AKI1.75       IF HOLDING REGISTER NOT ENPTY
                        449
000.336  076 023        450          MVI     A,XOFF          ELSE, SEND CTL-S
000.340  323 100        451          OUT     AP.THR
```

```
000.342  072 311 100   452          LDA     MODEI           GET MODE FLAGS
000.345  366 020       453          ORI     MI.XOFF         SET XOFF SENT
000.347  062 311 100   454          STA     MODEI
                       455
000.352  172           456  AKI1.8  MOV     A,D             GET INPUT CHARACTER
000.353  315 251 013   457          CALL    PCIF            ELSE, PUT CHARACTER IN INPUT FIFO
000.356  070 131       458          JR      C,AKI5          IF FIFO IS FULL, TOSS CHARACTER AND DING BELL
000.360  030 131       459          JR      AKI6            IF FIFO NOT FULL, EXIT WITHOUT BELL
                       460
                       461
000.362  376 002       462  AKI2    CPI     AB.TREI         SEE IF INTERRUPT WAS FROM XMIT BUFFER EMPTY
000.364  040 025       463          JR      NZ,AKI4         IF NOT FROM ACE TRANSMITTER
                       464
                       465  *       INTERRUPT CAUSED BY ACE TRANSMITTER HOLDING REGISTER EMPTY
                       466  *
                       467  *       BIT     IB.ONLN,C       SEE IF ON-LINE
000.366  313 131       468          DB      I.BITA,I.BITB!IB.ONLN!IR.C
000.370  050 011       469          JR      Z,AKI3          IF OFF-LINE, DON'T OUTPUT CHARACTER
                       470
000.372  315 032 012   471          CALL    FCOF            ELSE, FETCH A CHARACTER FROM THE OUTPUT FIFO
000.375  070 004       472          JR      C,AKI3          IF NO CHARACTER IN FIFO
                       473
000.377  323 100       474          OUT     AP.THR          OUTPUT CHARACTER TO ACE
001.001  030 110       475          JR      AKI6            EXIT
                       476
001.003  333 101       477  AKI3    IN      AP.IER          INPUT ACE INTERRUPT ENABLE REGISTER
001.005  346 375       478          ANI     377Q-AB.ETRE    CLEAR TRANSMITTER REGISTER EMPTY INTERRUPT
001.007  323 101       479          OUT     AP.IER
001.011  030 100       480          JR      AKI6            EXIT
                       481
001.013  333 240       482  AKI4    IN      KP.2            READ SECOND KEYBOARD PORT
001.015  356 366       483          XRI     11110110B       INVERT SWITCHES (EXCEPT OFF-LINE)
001.017  107           484          MOV     B,A             SAVE INPUT
001.020  346 200       485          ANI     KB.STB          IS A KEYBOARD STROBE PRESENT?
001.022  050 067       486          JR      Z,AKI6          IF NO STROBE
                       487
                       488  *       INTERRUPT CAUSED BY A KEY STRIKE ON THE KEYBOARD
                       489  *
001.024  333 200       490          IN      KP.1            INPUT KEY VALUE
001.026  107           491          MOV     B,A
001.027  333 240       492          IN      KP.2            INPUT SECOND PORT
001.031  356 366       493          XRI     11110110B       INVERT SWITCHES (EXCEPT OFF LINE)
001.033  117           494          MOV     C,A
                       495
                       496  *       SEE IF CHARACTER IS FROM EXTRA KEY #1.  IF SO, INVERT KEYBOARD DISABLE
                       497  *
001.034  170           498          MOV     A,B             GET KEY VALUE
001.035  376 213       499          CPI     KB.CTL+KB.EX1   MUST BE CTL-EX1
001.037  072 311 100   500          LDA     MODEI           GET MODE FLAGS
001.042  040 007       501          JR      NZ,AKI4.3       IF NOT CONTROL AND EX1
                       502
001.044  356 004       503          XRI     MI.KID          INVERT KEYBOARD DISABLE FLAG
001.046  062 311 100   504          STA     MODEI           UPDATE FLAGS
001.051  030 004       505          JR      AKI4.4          SEE IF TO TICK-ET
                       506
                       507  *       SEE IF KEYBOARD IS DISABLED
```

```
                                508  *
001.053  346 004               509  AKI4.3  ANI     MI,KID          MASK FOR KEYBOARD INPUT DISABLED FLAG
001.055  040 034               510          JR      NZ,AKI6         IF DISABLED, EXIT
                                511
                                512  *       SEE IF TO TICK UPON KEYBOARD INPUT
                                513  *
001.057  072 310 100           514  AKI4.4  LDA     MODEB           GET PROPER FLAGS
001.062  346 002               515          ANI     MB.NOTK         NO TICK?
001.064  040 002               516          JR      NZ,AKI4.8       IF NOT TO MAKE NOISE
                                517
001.066  323 300               518          OUT     MP.TICK         TICK TICKER TO INDICATE KEY STRIKE
                                519
                                520  *       CAN PUT CHARACTER IN FIFO TO BE PROCESSED BY KCE
                                521  *
001.070  052 264 100           522  AKI4.8  LHLD    KBDFP           GET KEYBOARD FIFO POINTER
001.073  175                   523          MOV     A,L             GET POINTER LSB
001.074  376 264               524          CPI     KBDFMAX&377Q    SEE IF ROOM IN FIFO
001.076  050 011               525          JR      Z,AKI5          IF NO ROOM, DING BELL
                                526
001.100  160                   527          MOV     M,B             PLACE IN FIFO
001.101  043                   528          INX     H
001.102  161                   529          MOV     M,C             PLACE IN FIFO
001.103  043                   530          INX     H
001.104  042 264 100           531          SHLD    KBDFP           UPDATE FIFO POINTER
001.107  030 002               532          JR      AKI6            EXIT
                                533
001.111  323 340               534  AKI5    OUT     MP.BELL         DING THE DANG BELL
                                535
                                536  *AKI6   EX      AF,AF'          EXCHANGE ALL REGISTERS
001.113  010                   537  AKI6    DB      I.EXAF
                                538  *       EXX
001.114  331                   539          DB      I.EXX
001.115  373                   540          EI                      ENABLE NEW INTERRUPTS
001.116  311                   541          RET
```

```
                        544  ***     KCE - KEYBOARD CHARACTER ENCODER
                        545  *
                        546  *       *KCE* IS RESPONSIBLE FOR TAKING THE VALUE SUPPLIED BY THE HARDWARE
                        547  *       KEYBOARD ENCODER AND DETERMINING IF THE VALUE IS A LEGITAMATE ASCII
                        548  *       VALUE.  IF THE VALUE DOES NOT REPRESENT THE ACTUAL ASCII VALUE FOR
                        549  *       THE KEY STRUCK, *KCE* FORMS THE CORRECT VALUE.
                        550  *
                        551  *       THE HARDWARE ENCODER ALSO SIGNALS WHETHER THE 'CONTROL' OR 'SHIFT'
                        552  *       KEY WAS PRESSED.  IF THE HARDWARE ENCODER DOES NOT REFLECT THE
                        553  *       CONTROL OR SHIFT VALUE FOR THE KEY, *KCE* SETS THE CORRECT VALUE
                        554  *       PRIOR TO EXITING.
                        555  *
                        556  *       *KCE* ALSO PLACES THE ENCODED ASCII VALUE OF THE KEY INTO EITHER
                        557  *       THE INPUT OR OUTPUT FIFO.  IF THE KEY STRUCK IS A FUNCTION KEY
                        558  *       RATHER THAN AN ALPHABETICAL OR NUMERICAL KEY AND THE CONTROL KEY
                        559  *       WAS ALSO STRUCK, THE ENCODED VALUE IS PLACED IN THE INPUT FIFO.
                        560  *       ALL OTHER VALUES ARE PLACED IN THE OUTPUT FIFO.
                        561  *
                        562  *
                        563  *       ENTRY   (D,E) = VALUE OF KEY SUPPLIED BY THE HARDWARE ENCODER
                        564  *       EXIT    NONE
                        565  *       USES    A,B,C,D,E,H,L
                        566
                        567
001.117  172           568  KCE     MOV     A,D             PLACE KEY VALUE IN ACC
001.120  346 177       569          ANI     01111111B       MASK OFF ANY CONTROL BIT
                        570
                        571  *       ENCODE KEY VALUES 000Q THRU 017Q AND 033Q
                        572  *
001.122  376 033       573          CPI     033Q            CHECK FOR 'ESC' KEY
001.124  050 151       574          JR      Z,KCE3          IF KEY WAS THE 'ESC' KEY
                        575
001.126  376 020       576          CPI     020Q            KEY < 020Q ?
001.130  322 344 001   577          JNC     KCE6            IF KEY > 017Q
                        578
001.133  325           579  KCE1    PUSH    D               SAVE KEYBOARD VALUES
001.134  041 350 002   580          LXI     H,KAE1          POINT TO ASCII EQUIV. TABLE #1
001.137  026 011       581          MVI     D,KAE1L         (D) = LENGTH OF TABLE
001.141  036 002       582          MVI     E,KAE1W         (E) = WIDTH OF TABLE IN BYTES
001.143  315 363 015   583          CALL    STAB            SEARCH TABLE
001.146  321           584          POP     D               (D,E) = KEYBOARD VALUES
001.147  070 047       585          JR      C,KCE1.5        IF NO ENTRY WAS FOUND
                        586
001.151  376 312       587          CPI     ESCF+'J'        WAS IT THE ERASE KEY?
001.153  040 051       588          JR      NZ,KCE2         IF NOT
                        589
                        590  *       BIT     IB.KSB,E        ELSE, SEE IF SHIFT KEY ALSO
001.155  313 103       591          DB      I.BITA,I.BITB!IB.KSB!IR.E
001.157  050 045       592          JR      Z,KCE2          IF NO SHIFT KEY, JUST SEND AN ERM
                        593
001.161  072 310 100   594          LDA     MODEB           GET MODE FLAGS
001.164  346 040       595          ANI     MB.ANSI         IN ANSI MODE?
001.166  050 024       596          JR      Z,KCE1.4        IF NOT IN ANSI MODE
                        597
                        598  *       BIT     IB.KCB,D        SEE IF CONTROL KEY WAS DOWN
001.170  313 172       599          DB      I.BITA,I.BITB!IB.KCB!IR.D
```

```
001.172  050 010        600           JR      Z,KCE1.2        IF NO CONTROL KEY
                         601
001.174  315 051 015     602           CALL    PSIF            CONTROL DOWN, PUT STRING IN INPUT FIFO
001.177  033 133 062     603           DB      ESC,'[','2','J'+200Q
001.203  311             604           RET
                         605
001.204  315 100 015     606  KCE1.2   CALL    PSOF            NO CONTROL, PUT STRING IN OUTPUT FIFO
001.207  033 133 062     607           DB      ESC,'[','2','J'+200Q
001.213  311             608           RET
                         609
001.214  076 305         610  KCE1.4   MVI     A,ESCF+'E'      ELSE, SEND A CLR
001.216  030 006         611           JR      KCE2
                         612
001.220  172             613  KCE1.5   MOV     A,D             ELSE, PLACE ORIGINAL VALUE BACK IN ACC
001.221  346 177         614           ANI     377Q-KB.CTL     NO CONTROL = INPUT FIFO ON CONTROL CODES HERE
001.223  127             615           MOV     D,A             UPDATE (D)
001.224  030 051         616           JR      KCE3            GO PLACE ASCII VALUE IN APPROPRIATE FIFO
                         617
                         618
                         619  *       PLACE ASCII VALUE(S) IN THE APPROPRIATE FIFO
                         620  *       IF BIT 7 IS SET IN VALUE, FIRST PLACE AN 'ESC' FOLLOWED BY THE SEVEN LSB
                         621  *
                         622  *KCE2    BIT     IB.ESCF,A       SEE IF 'ESC' TO BE SENT BEFORE ALPHA CHARACTER
001.226  313 177         623  KCE2     DB      I,BITA,I.BITB!IB.ESCF!IR.A
001.230  050 045         624           JR      Z,KCE3          IF NO 'ESC' IS TO BE SENT
                         625
001.232  346 177         626           ANI     377Q-ESCF       REMOVE ESCAPE FLAG
001.234  365             627           PUSH    PSW             ELSE, SAVE VALUE FROM TABLE
001.235  076 033         628           MVI     A,ESC           SET (A) = 'ESC'
                         629
                         630  *       CHECK FOR "CONTROL" KEY.  IF STRUCK, PLACE CHARACTERS IN INPUT FIFO
                         631  *
                         632  *       BIT     IB.KCB,D        SEE IF CONTROL KEY WAS STRUCK ON KEYBOARD
001.237  313 172         633           DB      I,BITA,I.BITB!IB.KCB!IR.D
001.241  050 046         634           JR      Z,KCE4          IF CONTROL KEY NOT STRUCK
                         635
001.243  363             636           DI                      LOCK OUT OTHER INPUTS (*AKI*)
001.244  315 251 013     637           CALL    PCIF            PUT CHARACTER IN FIFO
001.247  373             638           EI                      ALLOW INPUTS FROM *AKI*
001.250  072 310 100     639           LDA     MODER           GET MODE FLAGS
001.253  346 040         640           ANI     MB.ANSI         SEE IF IN ANSI MODE
001.255  050 017         641           JR      Z,KCE2.7        IF HEATH MODE IS SELECTED
                         642
001.257  361             643           POP     PSW             GET CHARACTER
001.260  365             644           PUSH    PSW
001.261  376 120         645           CPI     'P'             SEE IF CHAR < P
001.263  076 133         646           MVI     A,'['           IF < P, OUTPUT A "["
001.265  070 002         647           JR      C,KCE2.3
                         648
001.267  076 117         649           MVI     A,'O'           ELSE, OUTPUT AN "O"
001.271  363             650  KCE2.3   DI
001.272  315 251 013     651           CALL    PCIF
001.275  373             652           EI
001.276  361             653  KCE2.7   POP     PSW             GET ASCII VALUE
                         654  *KCE3    BIT     IB.KCB,D        TEST CONTROL KEY FOR PLACEMENT OF THIS CHARACTER
001.277  313 172         655  KCE3     DB      I,BITA,I.BITB!IB.KCB!IR.D
```

```
001.301  050 036        656           JR      Z,KCE5          IF CONTROL KEY NOT STRUCK
                         657
001.303  363            658           DI                      LOCK OUT OTHER INPUTS
001.304  315 251 013    659           CALL    PCIF            PLACE CHARACTER IN INPUT FIFO
001.307  373            660           EI                      ALLOW *AKI* INPUTS
001.310  311            661           RET
                         662
001.311  315 375 013    663  KCE4     CALL    PCOFT           PLACE CHARACTER IN OUTPUT FIFO
001.314  072 310 100    664           LDA     MODEB           GET MODE FLAGS
001.317  346 040        665           ANI     MB.ANSI         SEE IF IN ANSI MODE
001.321  050 015        666           JR      Z,KCE4.7        IF IN HEATH MODE
                         667
001.323  361            668           POP     PSW             GET CHARACTER
001.324  365            669           PUSH    PSW
001.325  376 120        670           CPI     'P'             SEE IF < P
001.327  076 133        671           MVI     A,'['           IF < P, OUTPUT A '['
001.331  070 002        672           JR      C,KCE4.3        IF < P
                         673
001.333  076 117        674           MVI     A,'O'           ELSE, OUTPUT AN 'O'
001.335  315 375 013    675  KCE4.3   CALL    PCOFT
                         676
001.340  361            677  KCE4.7   POP     PSW             GET ASCII CHARACTER
001.341  303 375 013    678  KCE5     JMP     PCOFT           PLACE CHARACTER IN OUTPUT FIFO
                         679
                         680  *       ENCODE KEY VALUES 020Q THRU 034Q (EXCEPT 033Q)
                         681  *       THESE VALUES ARE FROM THE 12 KEY NUMERIC PAD
                         682  *
001.344  376 035        683  KCE6     CPI     035Q            KEY < 35Q ?
001.346  322 246 002    684           JNC     KCE12           IF KEY > 34Q
                         685
001.351  325            686           PUSH    D               SAVE KEYBOARD VALUES
001.352  041 372 002    687           LXI     H,KAE2          POINT TO SECOND ASCII EQUIVALENCE TABLE
001.355  026 014        688           MVI     D,KAE2L         GET TABLE LENGTH
001.357  036 005        689           MVI     E,KAE2W         GET TABLE WIDTH
001.361  315 363 015    690           CALL    STAB            SEARCH TABLE
001.364  321            691           POP     D               (D,E) = KEYBOARD VALUES
                         692
                         693  *       CHECK FOR SHIFT KEY AND/OR KEYPAD SHIFT MODE
                         694  *
                         695  *       BIT     IB.KSB,E        SHIFT KEY DOWN?
001.365  313 103        696           DB      I.BITA,I.BITB!IB.KSB!IR.E
001.367  072 310 100    697           LDA     MODEB           GET MODE FLAGS
001.372  050 006        698           JR      Z,KCE7          IF NO SHIFT KEY
                         699
                         700  *       BIT     IB.KPDS,A       TEST FOR KEYPAD SHIFT MODE
001.374  313 167        701           DB      I.BITA,I.BITB!IB.KPDS!IR.A
001.376  040 007        702           JR      NZ,KCE9         IF SHIFT KEY & SHIFT MODE, DON'T SHIFT!
002.000  030 004        703           JR      KCE8            ELSE SHIFT KEY & NO SHIFT MODE, SO SHIFT
                         704
                         705
                         706 *KCE7    BIT     IB.KPDS,A       TEST FOR KEYPAD SHIFT MODE
002.002  313 167        707  KCE7     DB      I.BITA,I.BITB!IB.KPDS!IR.A
002.004  050 001        708           JR      Z,KCE9          NO SHIFT KEY & NO SHIFT MODE
                         709
002.006  043            710  KCE8     INX     H               POINT TO SHIFTED BYTE IN TABLE
                         711
```

```
002.007  346 200        712  KCE9    ANI     MB.KPDA          TEST FOR KEYPAD IN ALTERNATE MODE
002.011  050 002        713           JR      Z,KCE10          IF NOT IN ALT MODE
                        714
002.013  043            715           INX     H                ELSE, POINT TO ALT MODE BYTE
002.014  043            716           INX     H
002.015  257            717  KCE10   XRA     A                CLEAR ACC AND GET BYTE FROM TABLE (SET FLAGS)
002.016  206            718           ADD     M
002.017  362 277 001     719           JP      KCE3             IF NOT AN ESC FUNCTION, GO PUT IN FIFO
                        720
002.022  376 315        721           CPI     ESCF+'M'         CHECK FOR ESC-M (COULD BE ESC-M OR ESC-?-M)
002.024  050 155        722           JR      Z,KCE10.5        IF ESC-M
                        723
002.026  376 340        724           CPI     11100000B        CHECK FOR LOWER CASE WITH 'ESC' (ALT MODE CHAR)
002.030  060 162        725           JR      NC,KCE11         IF NOT A CURSOR FUNCTION
                        726
002.032  376 316        727           CPI     ESCF+'N'         SEE IF DELETE CHARACTER
002.034  040 033        728           JR      NZ,KE10.07       IF NOT THE DC KEY
                        729
002.036  107            730           MOV     B,A              ELSE, SAVE CHARACTER
002.037  072 310 100     731           LDA     MODEB            GET MODE FLAGS
002.042  346 040        732           ANI     MB.ANSI          SEE IF IN ANSI MODE
002.044  170            733           MOV     A,B              (A) = CHARACTER
002.045  050 022        734           JR      Z,KE10.07        IF NOT IN ANSI MODE
                        735
                        736  *       BIT     IB.KCB,D         SEE IF CONTROL KEY PRESSED
002.047  313 172        737           DB      I.BITA,I.BITB!IB.KCB!IR.D
002.051  050 007        738           JR      Z,KE10.03        IF NO CONTROL KEY
                        739
002.053  315 051 015     740           CALL    PSIF             ELSE, PUT IN INPUT FIFO
002,056  033 133 320     741           DB      ESC,'[','P'+200Q
002.061  311            742           RET
                        743
002.062  315 100 015     744  KE10.03 CALL    PSOF             PUT STRING IN OUTPUT FIFO
002,065  033 133 320     745           DB      ESC,'[','P'+200Q
002.070  311            746           RET
                        747
002.071  376 300        748  KE10.07 CPI     ESCF+'@'         CHECK FOR INSERT CHARACTER CODE
002.073  302 226 001     749           JNZ     KCE2             IF NOT INSERT CHARACTER KEY
                        750
002.076  072 310 100     751           LDA     MODEB            GET MODE FLAGS
002.101  346 040        752           ANI     MB.ANSI          SEE IF IN ANSI MODE
002,103  050 057        753           JR      Z,KCE10.4        IF IN HEATH MODE
                        754
002,105  072 307 100     755           LDA     MODEA            ELSE, GET MODEA FLAGS
002.110  346 100        756           ANI     MA.ICM           SEE IF IN ICM
002.112  050 024        757           JR      Z,KCE10.2        IF NOT ALREADY IN INSERT MODE
                        758
                        759  *       BIT     IB.KCB,D         SEE IF CONTROL KEY WAS DOWN
002.114  313 172        760           DB      I.BITA,I.BITB!IB.KCB!IR.D
002.116  050 010        761           JR      Z,KCE10.1        IF NO CONTROL KEY
                        762
002.120  315 051 015     763           CALL    PSIF             ELSE, PUT STRING IN INPUT FIFO
002.123  033 133 064     764           DB      ESC,'[','4','1'+200Q
002.127  311            765           RET
                        766
002.130  315 100 015     767  KCE10.1 CALL    PSOF             PUT STRING IN OUTPUT FIFO
```

```
002.133  033 133 064   768           DB      ESC,'[','4','1'+200Q
002.137  311           769           RET
                       770
                       771  *KCE10.2  BIT     IB.KCB,D          CHECK FOR CONTROL KEY
002.140  313 172       772  KCE10.2  DB      I.BITA,I.BITB!IB.KCB!IR.D
002.142  050 010       773           JR      Z,KCE10.3         IF NO CONTROL KEY
                       774
002.144  315 051 015   775           CALL    PSIF              ELSE, PUT STRING IN INPUT FIFO
002.147  033 133 064   776           DB      ESC,'[','4','h'+200Q
002.153  311           777           RET
                       778
002.154  315 100 015   779  KCE10.3  CALL    PSOF              PUT STRIN IN OUTPUT FIFO
002.157  033 133 064   780           DB      ESC,'[','4','h'+200Q
002.163  311           781           RET
                       782
002.164  072 307 100   783  KCE10.4  LDA     MODEA             GET MODE FLAGS
002.167  346 100       784           ANI     MA.ICM            SEE IF ALREADY IN INSERT MODE
002.171  076 300       785           MVI     A,EICSEQ+ESCF     ENTER INSERT MODE
002.173  312 226 001   786           JZ      KCE2              IF NOT ALREADY IN INSERT MODE
                       787
002.176  076 317       788           MVI     A,XICSEQ+ESCF     ELSE, EXIT INSERT MODE
002.200  303 226 001   789           JMP     KCE2
                       790
002.203  172           791  KCE10.5  MOV     A,D               WAS 'M', SEE IF FROM THE 3 KEY (DELETE LINE)
002.204  346 177       792           ANI     01111111B         TOSS CONTROL BIT
002.206  376 023       793           CPI     023Q              3 KEY?
002.210  176           794           MOV     A,M               REPLACE TABLE VALUE IN A
002.211  312 226 001   795           JZ      KCE2              IF FROM 3 KEY GO PLACE 'ESC','M' IN FIFO
                       796
                       797  *        HAVE A "KEYPAD ALTERNATE MODE" CHARACTER
                       798  *        PLACE AN 'ESC' A '?' (OR 'O' IF IN ANSI MODE) AND 7 LSB
                       799  *        FROM TABLE IN FIFO
                       800  *
002.214  365           801  KCE11    PUSH    PSW               SAVE TABLE CHARACTER
002.215  076 033       802           MVI     A,ESC             PLACE 'ESC' IN FIFO
002.217  315 375 013   803           CALL    PCOFT
002.222  072 310 100   804           LDA     MODEB             GET MODE FLAGS
002.225  346 040       805           ANI     MB,ANSI           IN ANSI MODE?
002.227  076 077       806           MVI     A,'?'             FOR HEATH
002.231  050 002       807           JR      Z,KCE11.5         IF IN HEATH MODE
                       808
002.233  076 117       809           MVI     A,'O'             FOR ANSI
002.235  315 375 013   810  KCE11.5  CALL    PCOFT             PLACE IN FIFO
002.240  361           811           POP     PSW               GET TABLE VALUE BACK
002.241  346 177       812           ANI     01111111B         TOSS ESC BIT
002.243  303 375 013   813           JMP     PCOFT             PLACE LAST CHARACTER IN FIFO
                       814
                       815  *        ENCODE KEY VALUES FOR THE SCROLL KEY (037Q)
                       816  *
002.246  376 037       817  KCE12    CPI     037Q              WAS IT THE SCROLL KEY?
002.250  040 014       818           JR      NZ,KCE13          IF NOT THE SCROLL KEY
                       819
002.252  041 313 100   820           LXI     H,HSMLC           ELSE, POINT TO THE HSM LINE COUNTER
                       821  *        BIT     IB.KSB,E          SEE IF SHIFTED SCROLL
002.255  313 103       822           DB      I.BITA,I.BITB!IB.KSB!IR.E
002.257  040 002       823           JR      NZ,KCE12.5        IF SHIFTED
```

```
                                824
002.261  064                   825             INR     M               INCREMENT LINE COUNTER FOR ONE MORE LINE
002.262  311                   826             RET                     EXIT
                               827
002.263  066 030               828  KCE12.5    MVI     M,24            SET LINE COUNTER TO 24 LINES
002.265  311                   829             RET                     EXIT
                               830
                               831  *          AT LAST! SIMPLE ASCII KEYS!!!
                               832  *          ENCODE KEY VALUES FOR 040Q THRU 177Q
                               833  *
                               834  *KCE13     BIT     IB.KSB,E        SHIFT KEY STRUCK?
002.266  313 103               835  KCE13      DB      I.BITA,I.BITB!IB.KSB!IR.E
002.270  050 014               836             JR      Z,KCE13.5       IF NO SHIFT
                               837
002.272  325                   838             PUSH    D               SAVE KEYBOARD VALUES
002.273  041 066 003           839             LXI     H,KAE3          ELSE POINT TO EQUIV TABLE #3 FOR SPEC SHIFTS
002.276  026 015               840             MVI     D,KAE3L         SET TABLE LENGTH
002.300  036 002               841             MVI     E,KAE3W         SET TABLE WIDTH
002.302  315 363 015           842             CALL    STAB            SEARCH TABLE
002.305  321                   843             POP     D               (D,E) = KEYBOARD VALUES
                               844
                               845  *KCE13.5   BIT     IB.CPLK,E       TEST FOR 'CAPS LOCK' ON
002.306  313 113               846  KCE13.5    DB      I.BITA,I.BITB!IB.CPLK!IR.E
002.310  050 012               847             JR      Z,KCE14         IF CAPS LOCK NOT ON
                               848
002.312  376 141               849             CPI     'a'             TEST FOR < LOWER CASE A
002.314  070 006               850             JR      C,KCE14         IF LESS THAN A LOWER CASE CHARACTER
                               851
002.316  376 173               852             CPI     '{'             TEST FOR < LEFT BRACE
002.320  060 002               853             JR      NC,KCE14        IF GREATER THAN A LOWER CASE CHARACTER
                               854
002.322  346 337               855             ANI     11011111B       IS LOWER CASE, MAKE IT UPPER CASE
                               856
                               857  *KCE14     BIT     IB.KCB,D        TEST FOR 'CONTROL' KEY
002.324  313 172               858  KCE14      DB      I.BITA,I.BITB!IB.KCB!IR.D
002.326  312 375 013           859             JZ      PCOFT           IF CONTROL NOT STRUCK
                               860
002.331  376 100               861             CPI     '@'             NO CONTROL CODES IF < 100Q
002.333  332 375 013           862             JC      PCOFT           IF < 100Q
                               863
002.336  376 173               864             CPI     '{'             NO CONTROL CODES IF > 172Q
002.340  322 375 013           865             JNC     PCOFT           IF > 172Q
                               866
002.343  346 037               867             ANI     00011111B       ELSE FORM CONTROL CODE
002.345  303 375 013           868             JMP     PCOFT           GO PLACE CODE IN FIFO
```

```
                               870  **         KEYBOARD TO ASCII EQUIVALENCE TABLES
                               871  *
```

```
                                873  *       *KAE1* IS A LOOK UP TABLE FOR FUNCTION KEY VALUES
                                874  *
                                875  *       TABLE ENTRYS ARE: KEY VALUE FOLLOWED BY ASCII VALUE.  *ESCF* IS USED
                                876  *       TO SIGNAL THAT AN ESCAPE CODE IS TO PRECEED THE SEVEN BIT ASCII VALUE
                                877
                                878
002.350                         879  KAE1    EQU     *
002.350  000 323               880          DB      0,ESCF+'S'      KEY VALUE 0 = ESC-S
002.352  001 324               881          DB      1,ESCF+'T'      KEY VALUE 1 = ESC-T
002.354  002 325               882          DB      2,ESCF+'U'      KEY VALUE 2 = ESC-U
002.356  003 326               883          DB      3,ESCF+'V'      KEY VALUE 3 = ESC-V
002.360  004 327               884          DB      4,ESCF+'W'      KEY VALUE 4 = ESC-W
002.362  005 312               885          DB      5,ESCF+'J'      KEY VALUE 5 = ESC-J
002.364  006 320               886          DB      6,ESCF+'P'      KEY VALUE 6 = ESC-P
002.366  007 321               887          DB      7,ESCF+'Q'      KEY VALUE 7 = ESC-Q
002.370  016 322               888          DB      16Q,ESCF+'R'    KEY VALUE 13Q = ESC-R
000.002                         889  KAE1W   EQU     2               TABLE IS TWO BYTES WIDE
000.011                         890  KAE1L   EQU     *-KAE1/KAE1W


                                892  *       KAE2 CONTAINS THE ASCII VALUES FOR THE VARIOUS MODES OF
                                893  *       THE 12 KEY NUMERIC KEY PAD.  TABLE ENTRYS ARE: KEY VALUE,
                                894  *       UNSHIFTED ASCII, SHIFTED ASCII, KEYPAD ALTERNATE MODE ASCII,
                                895  *       AND KEYPAD ALTERNATE MODE SHIFTED ASCII.
                                896  *
002.372                         897  KAE2    EQU     *
002.372  020 060 060            898          DB      20Q,'0','0',ESCF+'P',ESCF+'P'           P = LOWER CASE P
002.377  021 061 314            899          DB      21Q,'1',ESCF+'L',ESCF+'Q',ESCF+'L'      Q = LOWER CASE Q
003.004  022 062 302            900          DB      22Q,'2',ESCF+'B',ESCF+'R',ESCF+'B'      R = LOWER CASE R
003.011  023 063 315            901          DB      23Q,'3',ESCF+'M',ESCF+'S',ESCF+'M'      S = LOWER CASE S
003.016  024 064 304            902          DB      24Q,'4',ESCF+'D',ESCF+'T',ESCF+'D'      T = LOWER CASE T
003.023  025 065 310            903          DB      25Q,'5',ESCF+'H',ESCF+'U',ESCF+'H'      U = LOWER CASE U
003.030  026 066 303            904          DB      26Q,'6',ESCF+'C',ESCF+'V',ESCF+'C'      V = LOWER CASE V
003.035  027 067 300            905          DB      27Q,'7',ESCF+'@',ESCF+'W',ESCF+'@'      W = LOWER CASE W
003.042  030 070 301            906          DB      30Q,'8',ESCF+'A',ESCF+'X',ESCF+'A'      X = LOWER CASE X
003.047  031 071 316            907          DB      31Q,'9',ESCF+'N',ESCF+'Y',ESCF+'N'      Y = LOWER CASE Y
003.054  032 056 056            908          DB      32Q,'.','.',ESCF+'N',ESCF+'N'           N = LOWER CASE N
003.061  034 015 015            909          DB      34Q,CR,CR,ESCF+'M',ESCF+'M'
000.005                         910  KAE2W   EQU     5               TABLE WIDTH = FIVE BYTES
000.014                         911  KAE2L   EQU     *-KAE2/KAE2W


                                913  *       *KAE3* CONTAINS SHIFT VALUES FOR KEYS WHERE THE SHIFTED VALUE OF THE
                                914  *       KEY CANNOT BE FORMED BY A NORMAL ASCII BIT MASK SHIFT
                                915  *
                                916  *       TABLE ENTRYS ARE: UNSHIFTED VALUE FOLLOWED BY THE SHIFTED VALUE
                                917
                                918
003.066                         919  KAE3    EQU     *
003.066  047 042               920          DB      ''',''"'
003.070  055 137               921          DB      '-','_'
003.072  060 051               922          DB      '0',')'
003.074  062 100               923          DB      '2','@'
003.076  066 136               924          DB      '6','^'
```

```
003.100  067 046      925          DB      '7','&'
003.102  070 052      926          DB      '8','*'
003.104  071 050      927          DB      '9','('
003.106  073 072      928          DB      ';',':'
003.110  075 053      929          DB      '=','+'
003.112  133 135      930          DB      '[',']'
003.114  140 176      931          DB      '`','~'
003.116  173 175      932          DB      '{','}'
000.002               933  KAE3W   EQU     2       TABLE WIDTH = 2 BYTES
000.015               934  KAE3L   EQU     *-KAE3/KAE3W
```

```
                                937  ***     IFCP - INPUT FIFO CHARACTER PROCESSOR
                                938  *
                                939  *       *IFCP* INTERPRETS THE CHARACTER WHICH HAS BEEN RECEIVED (OR STRUCK
                                940  *       ON THE KEYBOARD IF OFF-LINE OR KEYED AS A LOCAL FUNCTION) TO SEE IF
                                941  *       IT IS THE SECOND CHARACTER IN AN ESCAPE SEQUENCE, A CONTROL CHARACTER
                                942  *       OR A CHARACTER TO BE DISPLAYED.  IF THE CHARACTER IS PART OF AN
                                943  *       ESCAPE SEQUENCE WHICH CANNOT BE FOUND IN *ESCTAB* THE ESCAPE SEQUENCE
                                944  *       FLAG IS CLEARED AND THE CHARACTER IS DISPLAYED.  IF THE CHARACTER IS
                                945  *       A CONTROL CHARACTER, AND CANNOT BE FOUND IN *CTLTAB*, THE CHARACTER IS
                                946  *       DISCARDED.  IF THE CHARACTER IS FOUND IN THE APPROPRIATE TABLE,  THE
                                947  *       PROGRAM COUNTER IS SET TO THE ASSOCIATED ADDRESS.
                                948  *
                                949  *
                                950  *       ENTRY   (A) = CHARACTER FROM INPUT FIFO
                                951  *
                                952  *       EXIT    NONE
                                953  *
                                954  *       USES    A,B,C,D,E,H,L,F
                                955
                                956
003.120  127                    957  IFCP    MOV     D,A             SAVE CHARACTER
003.121  072 311 100            958          LDA     MODEI           GET MODE FLAGS
                                959  *       BIT     IB.XOFF,A       SEE IF XOFF SENT
003.124  313 147                960          DB      I.BITA,I.BITB!IB.XOFF!IR.A
003.126  050 025                961          JR      Z,IFCP0.5       IF NO XOFF HAS BEEN SENT
                                962
003.130  137                    963          MOV     E,A             SAVE MODE FLAGS
003.131  072 241 100            964          LDA     IFC             GET INPUT FIFO COUNTER
003.134  376 140                965          CPI     IFMAX-32        SEE IF AT LEAST 32 BYTES ARE FREE
003.136  173                    966          MOV     A,E             GET MODE FLAGS BACK
003.137  060 014                967          JR      NC,IFCP0.5      IF NOT ENOUGH BYTES OPEN
                                968
003.141  346 357                969          ANI     255-MI.XOFF     RESET XOFF FLAG
003.143  062 311 100            970          STA     MODEI
003.146  137                    971          MOV     E,A
003.147  076 021                972          MVI     A,XON           SEND XON TO HOST
003.151  315 375 013            973          CALL    PCOFT
003.154  173                    974          MOV     A,E             GET MODE FLAGS BACK
                                975
                                976  *IFCP0.5 BIT    IB.PWE,A        SEE IF PREVIOUS CHARACTER WAS AN ESCAPE CODE
003.155  313 107                977  IFCP0.5 DB      I.BITA,I.BITB!IB.PWE!IR.A
003.157  172                    978          MOV     A,D             (A) = CHARACTER
003.160  050 046                979          JR      Z,IFCP1         IF CHARACTER IS NOT PART OF AN ESCAPE SEQ.
                                980
                                981  *       CHARACTER IS PART OF AN ESCAPE SEQUENCE
                                982  *
003.162  072 311 100            983          LDA     MODEI           GET MODE AGAIN
003.165  346 376                984          ANI     377Q-MI.PWE     CLEAR 'PREVIOUS WAS AN ESCPAE' FLAG
003.167  062 311 100            985          STA     MODEI
003.172  072 310 100            986          LDA     MODEB           GET MODE B FLAGS
003.175  346 040                987          ANI     MB.ANSI         SEE IF IN ANSI MODE
003.177  172                    988          MOV     A,D             (A) = CHARACTER
003.200  050 011                989          JR      Z,IFCP0.7       IF NOT IN ANSI MODE
                                990
003.202  041 340 004            991          LXI     H,AESCT         (H,L) = ANSI ESCAPE TABLE
003.205  026 004                992          MVI     D,AESCTL        (D) = TABLE LENGTH
```

```
003.207  036 003        993          MVI     E,AESCTW        (E) = TABLE WIDTH
003.211  030 007        994          JR      IFCP0.9         SEARCH TABLE
                        995
003.213  041 134 004    996  IFCP0.7 LXI     H,ESCTAB        (H,L) = HEATH ESCAPE SEQUENCE TABLE
003.216  026 054        997          MVI     D,ESCTABL       (D) = TABLE LENGTH
003.220  036 003        998          MVI     E,ESCTABW       (E) = TABLE WIDTH
                        999
003.222  315 363 015  1000  IFCP0.9 CALL    STAB            SEARCH TABLE
003.225  060 160       1001          JR      NC,IFCP3        IF A MATCH WAS FOUND IN THE TABLE
                       1002
003.227  311          1003          RET                     ELSE, FORGET ABOUT THIS CHARACTER AND EXIT
                       1004
003.230  376 012      1005  IFCP1   CPI     LF              IS CHARACTER A LINE FEED?
003.232  040 130      1006          JR      NZ,IFCP1.9      IF NOT A LINE FEED
                       1007
                       1008  *       CHARACTER IS A LINE FEED.  CHECK FOR HOLD SCREEN MODE
                       1009  *
003.234  072 307 100  1010          LDA     MODEA           GET MODEA FLAGS
003.237  346 001      1011          ANI     MA.HSM
003.241  312 112 007  1012          JZ      PLFCR           IF NOT IN HOLD SCREEN MODE, GO DO LINE FEED
                       1013
                       1014  *       CHARACTER IS A LINE FEED AND TERMINAL IS IN THE HOLD SCREEN MODE.
                       1015  *       THE TERMINAL OPERATION WILL NOT CHANGE UNLESS OR UNTIL THE 24TH
                       1016  *       LINE IS REACHED.  A "SCROLL" KEY FROM THE KEYBOARD WILL CAUSE ONE
                       1017  *       MORE LINE TO BE DISPLAYED.  A SHIFTED "SCROLL" KEY WILL CAUSE 24
                       1018  *       LINES TO BE INPUT AND DISPLAYED.  IF THE INPUT FIFO IS FULL, THE
                       1019  *       CONTENTS WILL BE DISPLAYED AND ALL CHARACTERS UP TO THE NEXT LINE
                       1020  *       FEED WILL BE DISPLAYED.
                       1021  *
003.244  072 273 100  1022          LDA     CURVP           GET VERTICAL POSITION
003.247  376 027      1023          CPI     23              SEE IF ON LAST LINE
003.251  302 112 007  1024          JNZ     PLFCR           IF NOT LAST LINE, DON'T HOLD YET
                       1025
003.254  041 313 100  1026          LXI     H,HSMLC         POINT TO HOLD SCREEN MODE LINE COUNTER
003.257  065          1027          DCR     M               DECREMENT COUNTER
003.260  302 112 007  1028          JNZ     PLFCR           IF NOT ZERO, DO THIS LF AND WAIT FOR NEXT
                       1029
003.263  076 023      1030          MVI     A,XOFF          SEND CTL-S TO HOST
003.265  315 375 013  1031          CALL    PCOFT           PLACE IN OUTPUT FIFO
003.270  315 045 000  1032          CALL    MAINA           TURN ON XMIT INTERRUPT
                       1033
003.273  315 107 012  1034  IFCP1.1 CALL    FVKF            GET NEXT VALUE FROM KEYBOARD FIFO
003.276  060 022      1035          JR      NC,IFCP1.5      IF A KEY WAS STRUCK
                       1036
003.300  315 111 016  1037          CALL    UCP             UPDATE CURSOR
003.303  072 241 100  1038  IFCP1.3 LDA     IFC             SEE IF INPUT FIFO IS FULL
003.306  376 200      1039          CPI     IFMAX
003.310  040 361      1040          JR      NZ,IFCP1.1      IF NOT FULL, CHECK ON KEYBOARD AGAIN
                       1041
003.312  076 001      1042          MVI     A,1             ELSE, SET COUNTER TO INPUT ONE MORE LINE
003.314  062 313 100  1043          STA     HSMLC
003.317  303 112 007  1044          JMP     PLFCR           DO THE LINE FEED AND EMPTY THE FIFO
                       1045
003.322  172          1046  IFCP1.5 MOV     A,D             GET KEY VALUE
003.323  346 177      1047          ANI     01111111B       TOSS THE CONTROL BIT
003.325  376 037      1048          CPI     SCRL            WAS KEY THE SCROLL KEY?
```

```
003.327  050 010       1049            JR      Z,IFCP1.7       IF SCROLL KEY
                        1050
003.331  315 117 001   1051            CALL    KCE             ELSE ENCODE THE ASCII VALUE
003.334  315 045 000   1052            CALL    MAINA           SEE IF THERE IS A CHARACTER TO OUTPUT
003.337  030 342       1053            JR      IFCP1.3         SEE IF KEYSTRIKE FILLED THE INPUT FIFO
                        1054
003.341                 1055  IFCP1.7   EQU     *
                        1056  *         BIT     IB.KSB,E        SEE IF THIS WAS A SHIFTED SCROLL KEY
003.341  313 103       1057            DB      I.BITA,I.BITB!IB.KSB!IR.E
003.343  076 001       1058            MVI     A,1             SET LINE COUNTER TO ONE
003.345  050 002       1059            JR      Z,IFCP1.8       IF SCROLL KEY WAS NOT SHIFTED
                        1060
003.347  076 030       1061            MVI     A,24            ELSE, SET LINE COUNTER TO 24
003.351  062 313 100   1062  IFCP1.8   STA     HSMLC
003.354  315 112 007   1063            CALL    PLFCR           DO THE LINE FEED
003.357  076 021       1064            MVI     A,XON           SEND CTL-Q TO HOST FOR MORE CHARACTERS
003.361  303 375 013   1065            JMP     PCOFT
                        1066
003.364  376 177       1067  IFCP1.9   CPI     RUBOUT          CHECK FOR RUBOUT AND OTHER CONTROL CODES
003.366  050 004       1068            JR      Z,IFCP2         IF RUBOUT, SEE IF THERE IS A HANDLER
                        1069
003.370  376 040       1070            CPI     ' '             CHECK FOR OTHER CONTROL CODES
003.372  060 026       1071            JR      NC,IFCP4        IF NOT A CONTROL CODE
                        1072
                        1073  *         CHARACTER IS A CONTROL CHARACTER
                        1074  *
003.374  041 354 004   1075  IFCP2     LXI     H,CTLTAB        (H,L) = CONTROL CHARACTER TABLE
003.377  026 006       1076            MVI     D,CTLTABL       (B) = TABLE LENGTH
004.001  036 003       1077            MVI     E,CTLTABW       (C) = TABLE WIDTH
004.003  315 363 015   1078            CALL    STAB            SEARCH TABLE
004.006  330           1079            RC                      IF NOT IN TABLE, TOSS IT AND EXIT
                        1080
                        1081  *         CHARACTER FOUND IN TABLE, GO TO ASSOCIATED ADDRESS
                        1082  *
004.007  137           1083  IFCP3     MOV     E,A             (E) = ADDRESS LSB
004.010  043           1084            INX     H
004.011  126           1085            MOV     D,M             (D) = ADDRESS MSB
004.012  353           1086            XCHG                    (H,L) = ADDRESS OF SPECIAL ROUTINE
004.013  001 310 100   1087            LXI     B,MODEB         (B,C) = MODEB IN CASE OF A CHANGE
004.016  021 307 100   1088            LXI     D,MODEA         (D,E) = MODEA IN CASE OF A CHANGE
004.021  351           1089            PCHL                    (PC) = ROUTINE
                        1090
004.022  072 307 100   1091  IFCP4     LDA     MODEA           GET MODE FLAGS
004.025  346 002       1092            ANI     MA,GRPH         SEE IF IN GRAPHICS MODE
004.027  172           1093            MOV     A,D             (A) = CHARACTER
004.030  050 015       1094            JR      Z,IFCP6         IF NOT IN GRAPHICS MODE
                        1095
004.032  376 136       1096            CPI     136Q            IS CHARACTER IN THE GRAPHICS RANGE?
004.034  332 047 004   1097            JC      IFCP6           IF NOT IN GRAPHIC RANGE
                        1098
004.037  040 004       1099            JR      NZ,IFCP5        IF CHARACTER IS NOT 136Q
                        1100
004.041  076 177       1101            MVI     A,177Q          ELSE MAKE 136Q A 177Q
004.043  030 002       1102            JR      IFCP6           DISPLAY IT
                        1103
004.045  346 037       1104  IFCP5     ANI     00011111B       MAKE CHARACTER INTO A GRAPHICS DISPLAY CHAR.
```

```
                          1105
                          1106  *       CHARACTER IS A DISPLAYABLE CHARACTER
                          1107  *
004,047  127              1108  IFCP6   MOV     D,A             SAVE CHARACTER IN D
004.050  072 307 100      1109          LDA     MODEA           GET MODE FLAGS
004.053  346 200          1110          ANI     MA.RV           MASK FOR REVERSE VIDEO FLAG
004.055  262              1111          ORA     D               ADD FLAG STATUS TO CHARACTER
000.000                   1112          ERRNZ   MA.RV-10000000B M.RV MUST BE BIT 7
004.056  365              1113          PUSH    PSW             SAVE CHARACTER
004.057  072 307 100      1114          LDA     MODEA           GET MODE AGAIN
004.062  346 100          1115          ANI     MA.ICM          CHECK FOR INSERT CHARACTER MODE
004.064  304 165 014      1116          CNZ     PIC             IF SELECTED, DO AN INSERT CHARACTER
                          1117
004.067  361              1118          POP     PSW             ELSE, GET CHARACTER AND DISPLAY IT
004.070  052 274 100      1119          LHLD    CURAD           GET CURSOR ADDRESS
004,073  167              1120          MOV     M,A             PUT CHARACTER AT CURSOR
004.074  043              1121          INX     H               POINT TO NEXT COLUMN
004.075  174              1122          MOV     A,H             STAY IN VIDEO RAM
004.076  366 370          1123          ORI     VRAMS/256
004.100  147              1124          MOV     H,A
004.101  072 272 100      1125          LDA     CURHP           GET CURRENT HORIZONTAL POSITION
004.104  376 117          1126          CPI     79              AT END OF LINE?
004.106  040 014          1127          JR      NZ,IFCP7        IF NOT AT END OF LINE
                          1128
004.110  072 310 100      1129          LDA     MODEB           ELSE, GET MODE FLAGS
004.113  346 004          1130          ANI     MB.WRAP         SEE IF TO WRAP AROUND
                          1131
004.115  310              1132          RZ                      IF NO WRAP
                          1133
                          1134  *       LDA     CURVP           GET LINE NUMBER
                          1135  *       CPI     23              < 22?
                          1136  *       RNC                     IF ON LAST LINE(S)
                          1137
004.116  315 354 006      1138          CALL    PLF             ELSE, LFCR
004.121  303 013 014      1139          JMP     PCR
                          1140
004.124  074              1141  IFCP7   INR     A               ELSE, POINT TO NEXT COLUMN
004.125  062 272 100      1142          STA     CURHP
004.130  042 274 100      1143          SHLD    CURAD           UPDATE CURAD
                          1144
004.133  311              1145          RET                     EXIT
```

```
                          1147  **      ESCTAB - ESCAPE SEQUENCE TABLE
                          1148  *
                          1149  *       *ESCTAB* CONTAINS THE SECOND CHARACTER OF ALL ESCAPE SEQUENCES
                          1150  *       THE H19 WILL RESPOND TO.
                          1151  *
                          1152  *       TABLE ENTRYS ARE: THE SECOND CHARACTER OF THE ESCPAE SEQUENCE,
                          1153  *       FOLLOWED BY THE ADDRESS OF THE ROUTINE WHICH PERFORMS THE
                          1154  *       REQUESTED OPERATION
                          1155
                          1156
004.134                   1157  ESCTAB  EQU     $
```

```
                        1158
004.134  074           1159          DB      '<'             ESC <
004.135  310 010       1160          DW      EAM             ENTER ANSI MODE
                        1161
004.137  033           1162          DB      ESC             ESC ESC
004.140  005 016       1163          DW      SPWE            SET PREVIOUS WAS AN ESCAPE AGAIN
                        1164
004.142  075           1165          DB      '='             ESC =
004.143  113 011       1166          DW      EKAM            ENTER KEYPAD ALTERNATE MODE
                        1167
004.145  076           1168          DB      '>'             ESC >
004.146  311 016       1169          DW      XKAM            EXIT KEYPAD ALTERNATE MODE
                        1170
004.150  043           1171          DB      '#'             ESC #
004.151  252 017       1172          DW      XMTP            TRANSMIT PAGE
                        1173
000.100                 1174  EICSEQ  EQU     '@'             ENTER INSERT CHARACTER MODE SEQUENCE
004.153  100           1175          DB      EICSEQ          ESC @
004.154  107 011       1176          DW      EICM            SET INSERT CHARACTER MODE
                        1177
004.156  101           1178          DB      'A'             ESC A
004.157  227 007       1179          DW      CUP             CURSOR UP
                        1180
004.161  102           1181          DB      'B'             ESC B
004.162  346 006       1182          DW      CDN             CURSOR DOWN
                        1183
004.164  103           1184          DB      'C'             ESC C
004.165  201 007       1185          DW      CRT             CURSOR RIGHT
                        1186
004.167  104           1187          DB      'D'             ESC D
004.170  126 007       1188          DW      CLFT            CURSOR LEFT
                        1189
004.172  105           1190          DB      'E'             ESC E
004.173  020 010       1191          DW      CLR             CLEAR DISPLAY
                        1192
004.175  106           1193          DB      'F'             ESC F
004.176  061 011       1194          DW      EGM             ENTER GRAPHICS MODE
                        1195
004.200  107           1196          DB      'G'             ESC G
004.201  275 016       1197          DW      XGM             EXIT GRAPHICS MODE
                        1198
004.203  110           1199          DB      'H'             ESC H
004.204  271 015       1200          DW      SCH             PERFORM CURSOR HOME
                        1201
004.206  111           1202          DB      'I'             ESC I
004.207  234 007       1203          DW      PRLF            PERFORM REVERSE LINE FEED
                        1204
004.211  112           1205          DB      'J'             ESC J
004.212  317 011       1206          DW      ERM             ERASE TO END OF MEMORY (PAGE)
                        1207
004.214  113           1208          DB      'K'             ESC K
004.215  277 011       1209          DW      EOL             ERASE TO END OF LINE
                        1210
004.217  114           1211          DB      'L'             ESC L
004.220  234 014       1212          DW      PIL             PERFORM INSERT LINE
                        1213
```

```
004.222  115          1214          DB      'M'             ESC M
004.223  075 014      1215          DW      PDL             PERFORM DELETE LINE
                       1216
004.225  116          1217          DB      'N'             ESC N
004.226  027 014      1218          DW      PDC             PERFORM DELETE CHARACTER
                       1219
000.117               1220  XICSEQ  EQU     'O'             EXIT INSERT CHARACTER MODE SEQUENCE
004.230  117          1221          DB      XICSEQ          ESC O
004.231  305 016      1222          DW      XICM            EXIT INSERT CHARACTER MODE
                       1223
004.233  131          1224          DB      'Y'             ESC Y
004.234  162 013      1225          DW      PCA             PERFORM CURSOR ADDRESSING
                       1226
004.236  132          1227          DB      'Z'             ESC Z
004.237  366 012      1228          DW      IDT             IDENTIFY TERMINAL
                       1229
004.241  133          1230          DB      '['             ESC [
004.242  076 011      1231          DW      EHSM            ENTER HOLD SCREEN MODE
                       1232
004.244  134          1233          DB      '\'             ESC \
004.245  301 016      1234          DW      XHSM            EXIT HOLD SCREEN MODE
                       1235
004.247  135          1236          DB      ']'             ESC ]
004.250  326 016      1237          DW      XMT25           TRANSMIT 25TH LINE (IF ENABLED)
                       1238
004.252  142          1239          DB      'b'             ESC b (LOWER CASE B)
004.253  315 010      1240          DW      EBD             ERASE BEGINNING OF DISPLAY
                       1241
                       1242
004.255  152          1243          DB      'j'             ESC j (LOWER CASE J)
004.256  132 015      1244          DW      SCP             SAVE CURSOR POSITION
                       1245
004.260  153          1246          DB      'k'             ESC k (LOWER CASE K)
004.261  314 015      1247          DW      USCP            UNSAVE CURSOR POSITION
                       1248
004.263  154          1249          DB      'l'             ESC l (LOWER CASE L)
004.264  051 011      1250          DW      EEL             ERASE ENTIRE LINE
                       1251
004.266  156          1252          DB      'n'             ESC n (LOWER CASE N)
004.267  153 007      1253          DW      CPR             CURSOR POSITION REPORT
                       1254
004.271  157          1255          DB      'o'             ESC o (LOWER CASE O)
004.272  352 010      1256          DW      EBL             ERASE BEGINNING OF LINE
                       1257
004.274  160          1258          DB      'p'             ESC p  (LOWER CASE P)
004.275  361 011      1259          DW      ERVM            ENTER REVERSE VIDEO MODE
                       1260
004.277  161          1261          DB      'q'             ESC q  (LOWER CASE Q)
004.300  325 017      1262          DW      XRVM            EXIT REVERSE VIDEO MODE
                       1263
004.302  162          1264          DB      'r'             ESC r  (LOWER CASE R)
004.303  157 012      1265          DW      SBR             SET BAUD RATE
                       1266
                       1267
004.305  164          1268          DB      't'             ESC t (LOWER CASE T)
004.306  136 011      1269          DW      EKSM            ENTER KEYPAD SHIFTED MODE
```

```
                                1270
004.310  165                   1271            DB      'u'             ESC u (LOWER CASE U)
004.311  316 016               1272            DW      XKSM            EXIT KEYPAD SHIFTED MODE
                                1273
004.313  166                   1274            DB      'v'             ESC v (LOWER CASE V)
004.314  153 016               1275            DW      WEOL            WRAP AROUND AT END OF LINE
                                1276
004.316  167                   1277            DB      'w'             ESC w (LOWER CASE W)
004.317  216 010               1278            DW      DEOL            DISCARD AT END OF LINE
                                1279
004.321  170                   1280            DB      'x'             ESC x (LOWER CASE X)
004.322  141 015               1281            DW      SMS             HEATH SET MODE
                                1282
004.324  171                   1283            DB      'y'             ESC y (LOWER CASE Y)
004.325  122 015               1284            DW      RMS             HEATH RESET MODE
                                1285
004.327  172                   1286            DB      'z'             ESC z (LOWER CASE Z)
004.330  000 013               1287            DW      RAMP            RESET ALL MODES TO POWER UP CONFIGURATION
                                1288
004.332  173                   1289            DB      '{'             ESC { (LEFT BRACE)
004.333  125 011               1290            DW      EKI             ENABLE KEYBOARD INPUT
                                1291
004.335  175                   1292            DB      '}'             ESC } (RIGHT BRACE)
004.336  226 010               1293            DW      DKI             DISABLE KEYBOARD INPUT
                                1294
000.003                         1295  ESCTABW   EQU     3               TABLE IS THREE BYTES WIDE
000.054                         1296  ESCTABL   EQU     *-ESCTAB/ESCTABW


                                1298  *       AESCT - ANSI ESCAPE TABLE
                                1299  *
                                1300  *       *AESCT* CONTAINS THE SECOND CHARACTER OF THE ANSI ESCAPE TABLES
                                1301  *
                                1302  *       TABLE ENTRYS ARE THE SECOND CHARACTER FOLLOWED BY THE ADDRESS
                                1303  *       OF THE ROUTINE FOR THE REQUESTED FUNCTION
                                1304
004.340                         1305  AESCT     EQU     *
                                1306
004.340  075                   1307            DB      '='             ESC =
004.341  113 011               1308            DW      EKAM            ENTER KEYPAD ALTERNATE MODE
                                1309
004.343  076                   1310            DB      '>'             ESC >
004.344  311 016               1311            DW      XKAM            EXIT KEYPAD ALTERNATE MODE
                                1312
004.346  115                   1313            DB      'M'             ESC M
004.347  234 007               1314            DW      PRLF            REVERSE INDEX (REVERSE LINE FEED)
                                1315
004.351  133                   1316            DB      '['             ESC [
004.352  143 011               1317            DW      ELB             ESCAPE LEFT BRACKET
                                1318
000.003                         1319  AESCTW    EQU     3               TABLE IS THREE BYTES WIDE
000.004                         1320  AESCTL    EQU     *-AESCT/AESCTW
```

```
                        1322  *       CTLTAB - CONTROL CHARACTER TABLE
                        1323  *
                        1324  *       *CTLTAB* CONTAINS ALL CONTROL CODES WHICH THE H19 WILL RESPOND TO
                        1325  *
                        1326  *       TABLE ENTRYS ARE: THE CONTROL CHARACTER FOLLOWED BY THE ADDRESS
                        1327  *       OF THE ROUTINE WHICH PERFORM THE REQUESTED FUNCTION
                        1328
                        1329
004.354                 1330  CTLTAB  EQU     *
004.354  007           1331          DB      BELL            BELL
004.355  223 010       1332          DW      DING            DING THE BELL
                        1333
004.357  010           1334          DB      BS              BACKSPACE
004.360  126 007       1335          DW      PBS             PERFORM A BACKSPACE
                        1336
004.362  011           1337          DB      HT              HORIZONTAL TABULATION
004.363  036 016       1338          DW      TAB             TAB TO NEXT EIGTH COLUMN
                        1339
004.365  012           1340          DB      LF              LINE FEED
004.366  112 007       1341          DW      PLFCR           PERFORM LINE FEED AND/OR CARRIAGE RETURN
                        1342
004.370  015           1343          DB      CR              CARRIAGE RETURN
004.371  003 014       1344          DW      PCRLF           PERFORM CARRIAGE RETURN AND/OR LINE FEED
                        1345
004.373  033           1346          DB      ESC             ESCAPE
004.374  005 016       1347          DW      SPWE            SET PREVIOUS WAS AN ESCAPE FLAG
000.003                 1348  CTLTABW EQU     3               TABLE IS THREE BYTES WIDE
000.006                 1349  CTLTABL EQU     *-CTLTAB/CTLTABW
```

```
                        1352  ***     GENERAL USE SUBROUTINES



                        1354  **      A1M - ANSI MODE #1
                        1355  *
                        1356  *       *A1M* INPUTS THE PARAMETER STRING AND FINAL CHARACTER FOR THE
                        1357  *       ESC [ ?  SEQUENCE
                        1358  *
                        1359  *
                        1360  *       ENTRY   (B) = ZERO IF NO PN WAS INPUT
                        1361  *               (D,E) = PSDW
                        1362  *
                        1363  *       EXIT    (A) = FINAL CHARACTER
                        1364  *               (B) = ZERO IF NO PN WAS INPUT
                        1365  *               (D,E) = PSDW
                        1366  *
                        1367  *       USES    A,B,C,H,L,F
                        1368
                        1369
004.376  170           1370  A1M     MOV     A,B             SEE IF PN WAS INPUT
004.377  267           1371          ORA     A
005.000  300           1372          RNZ                     IF PN ALREADY INPUT, ILLEGAL, EXIT
                        1373
005.001  315 346 014   1374          CALL    PSD             INPUT PARAMETER STRING NOW
                        1375
                        1376  *       FINAL CHARACTER MUST BE A LOWER CASE H OR A LOWER CASE L
                        1377  *
005.004  376 150       1378          CPI     'h'             FINAL = LOWER CASE H?
005.006  050 005       1379          JR      Z,A1SM          IF H, IS SET MODE SEQUENCE
                        1380
005.010  376 154       1381          CPI     'l'             FINAL = LOWER CASE L?
005.012  050 033       1382          JR      Z,A1RM          IF L, IS RESET MODE SEQUENCE
                        1383
005.014  311           1384          RET                     IF NEITHER, EXIT



                        1386  *       A1SM - ANSI MODE #1 SET MODE SEQUENCE
                        1387  *
                        1388  *       *A1SM* SET THE SPECIFIED MODE(S)
                        1389  *
                        1390  *
                        1391  *       ENTRY   (B) = ZERO IF NO PN WAS INPUT
                        1392  *               (D,E) = PSDW
                        1393  *
                        1394  *       EXIT    NONE
                        1395  *
                        1396  *       USES    A,B,C,H,L,F
                        1397
                        1398
005.015  170           1399  A1SM    MOV     A,B             SEE IF PN WAS INPUT
005.016  267           1400          ORA     A
005.017  310           1401          RZ                      IF NO PN, EXIT
                        1402
```

```
005.020  353            1403           XCHG
005.021  001 310 100    1404           LXI     B,MODEB         (B,C) = MODEB
005.024  176            1405  A1SM1    MOV     A,M             GET PN
005.025  376 002        1406           CPI     2               PN = 2?
005.027  314 065 011    1407           CZ      EHM             IF 2, ENTER HEATH MODE
                        1408
005.032  176            1409           MOV     A,M             GET PN
005.033  376 007        1410           CPI     7               PN = 7?
005.035  314 153 016    1411           CZ      WEOL            IF 7, SET WRAP AROUND AT END OF LINE
                        1412
005.040  176            1413           MOV     A,M             GET PN
005.041  376 150        1414           CPI     'h'             PN = FINAL?
005.043  310            1415           RZ                      IF FINAL CHARACTER, EXIT
                        1416
005.044  043            1417           INX     H               ELSE, POINT TO NEXT PN
005.045  030 355        1418           JR      A1SM1           DECODE IT


                        1420  *        A1RM - ANSI MODE #1 RESET MODE SEQUENCE
                        1421  *
                        1422  *        *A1RM* RESETS THE MODE(S) SPECIFIED BY PN
                        1423  *
                        1424  *
                        1425  *        ENTRY   (B) = ZERO IF NO PN WAS INPUT
                        1426  *                (D,E) = PSDW
                        1427  *
                        1428  *        EXIT    NONE
                        1429  *
                        1430  *        USES    A,B,C,H,L,F
                        1431
                        1432
005.047  170            1433  A1RM     MOV     A,B             SEE IF PN WAS INPUT
005.050  267            1434           ORA     A
005.051  310            1435           RZ                      IF NO PN, EXIT
                        1436
005.052  353            1437           XCHG                    (H,L) = PSDW
005.053  001 310 100    1438           LXI     B,MODEB
005.056  176            1439  A1RM1    MOV     A,M
005.057  376 007        1440           CPI     7               PN = 7?
005.061  314 216 010    1441           CZ      DEOL            IF 7, SET DISCARD PAST END OF LINE
                        1442
005.064  176            1443           MOV     A,M             GET PN
005.065  376 154        1444           CPI     'l'             PN = FINAL?
005.067  310            1445           RZ                      IF FINAL, EXIT
                        1446
005.070  043            1447           INX     H               ELSE, POINT TO NEXT PN
005.071  030 363        1448           JR      A1RM1
```

```
                        1450  **      A2M - ANSI MODE #2
                        1451  *
                        1452  *       *A2M* INPUTS THE PARAMETER STRING AND THE FINAL CHARACTER FOR
                        1453  *       THE SEQUENCE  ESC [ >
                        1454  *
                        1455  *
                        1456  *       ENTRY   (B) = ZERO IF NO PN WAS INPUT
                        1457  *               (D,E) = PSDW
                        1458  *
                        1459  *       EXIT    (A) = FINAL CHARACTER
                        1460  *               (B) = ZERO IF NO PN WAS INPUT
                        1461  *               (D,E) = PSDW
                        1462  *
                        1463  *       USES    A,B,C,D,E,H,L,F
                        1464
                        1465
005.073  170            1466  A2M     MOV     A,B             SEE IF PN WAS INPUT
005.074  267            1467          ORA     A
005.075  300            1468          RNZ                     IF ALREADY INPUT, ILLEGAL, EXIT
                        1469
005.076  315 346 014    1470          CALL    PSD             INPUT PARAMETER STRING
005.101  376 150        1471          CPI     'h'             FINAL = LOWER CASE H?
005.103  050 005        1472          JR      Z,A2SM          IF H, GO SET MODE
                        1473
005.105  376 154        1474          CPI     'l'             FINAL = LOWER CASE L?
005.107  050 023        1475          JR      Z,A2RM          IF L, GO RESET MODE
                        1476
005.111  311            1477          RET                     ELSE, EXIT


                        1479  *       A2SM - ANSI MODE #2 SET MODE SEQUENCE
                        1480  *
                        1481  *       *A2SM* SETS THE MODE(S) SPECIFIED BY PN FOR THE SEQUENCE  ESC [ >
                        1482  *
                        1483  *
                        1484  *       ENTRY   (B) = ZERO IF NO PN WAS INPUT
                        1485  *               (D,E) = PSDW
                        1486  *
                        1487  *       EXIT    NONE
                        1488  *
                        1489  *       USES    A,B,C,D,E,H,L,F
                        1490
005.112  170            1491  A2SM    MOV     A,B             SEE IF PN WAS INPUT
005.113  267            1492          ORA     A
005.114  310            1493          RZ                      IF NO PN, EXIT
                        1494
005.115  041 205 015    1495  A2SM1   LXI     H,SMST          (H,L) = SET MODE SEQUENCE TABLE
005.120  032            1496          LDAX    D               GET PN
005.121  376 012        1497          CPI     10              < 10?
005.123  320            1498          RNC                     IF OUT OF RANGE OR FINAL
                        1499
005.124  325            1500          PUSH    D               SAVE PN POINTER
005.125  315 166 015    1501          CALL    SMSB            SET MODE
005.130  321            1502          POP     D
005.131  023            1503          INX     D               POINT TO NEXT PN
```

```
005.132  030 361       1504           JR      A2SM1


                       1506  *        A2RM - ANSI MODE #2 RESET MODE SEQUENCE
                       1507  *
                       1508  *        *A2RM* RESETS THE MODE(S) SPECIFIED BY PN FOR THE SEQUENCE  ESC [ >
                       1509  *
                       1510  *
                       1511  *        ENTRY   (B) = ZERO IF NO PN WAS INPUT
                       1512  *                (D,E) = PSDW
                       1513  *
                       1514  *        EXIT    NONE
                       1515  *
                       1516  *        USES    A,B,C,D,E,H,L,F
                       1517
                       1518
005.134  170           1519  A2RM     MOV     A,B             SEE IF PN WAS INPUT
005.135  267           1520           ORA     A
005.136  310           1521           RZ                      IF NO PN
                       1522
005.137  041 227 015   1523  A2RM1    LXI     H,RMST          (H,L) = RESET MODE SEQUENCE TABLE
005.142  032           1524           LDAX    D               GET PN
005.143  376 012       1525           CPI     10              <9?
005.145  320           1526           RNC                     IF OUT OF RANGE OR FINAL
                       1527
005.146  325           1528           PUSH    D               SAVE PN POINTER
005.147  315 166 015   1529           CALL    SMSB            RESET MODE
005.152  321           1530           POP     D
005.153  023           1531           INX     D               POINT TO NEXT PN
005.154  030 361       1532           JR      A2RM1


                       1534  **       ACDN - ANSI CURSOR DOWN
                       1535  *
                       1536  *        *ACDN* MOVES THE CURSOR TOWARD BOTTOM OF THE DISPLAY THE SPECIFIED
                       1537  *        NUMBER OF LINES
                       1538  *
                       1539  *
                       1540  *        ENTRY   (B) = ZERO IF NO PN WAS INPUT
                       1541  *                (D,E) = PSDW
                       1542  *
                       1543  *        EXIT    NONE
                       1544  *
                       1545  *        USES    A,B,C,D,E,H,L,F
                       1546
                       1547
005.156  170           1548  ACDN     MOV     A,B             SEE IF THERE WAS A PN INPUT
005.157  267           1549           ORA     A
005.160  312 346 006   1550           JZ      CDN             IF NO PN, DO ONLY ONE LINE
                       1551
005.163  032           1552           LDAX    D               GET PN
005.164  267           1553           ORA     A               SEE IF ZERO
005.165  312 346 006   1554           JZ      CDN             IF ZERO, DEFAULT TO ONE
```

```
                                1555
005.170  365                   1556  ACDN1   PUSH    PSW             SAVE PN
005.171  315 346 006           1557          CALL    CDN             MOVE CURSOR DOWN ONE LINE
005.174  361                   1558          POP     PSW
005.175  075                   1559          DCR     A               PN TIMES
005.176  040 370               1560          JR      NZ,ACDN1        IF NOT DONE
                                1561
005.200  311                   1562          RET


                                1564  **      ACLFT - ANSI CURSOR LEFT
                                1565  *
                                1566  *       *ACLFT* MOVES THE CURSOR TO THE BEGINNING OF THE LINE PN TIMES
                                1567  *
                                1568  *
                                1569  *       ENTRY   (B) = ZERO IF NO PN WAS INPUT
                                1570  *               (D,E) = PSDW
                                1571  *
                                1572  *       EXIT    NONE
                                1573  *
                                1574  *       USES    A,B,D,E,H,L,F
                                1575
                                1576
005.201  170                   1577  ACLFT   MOV     A,B             SEE IF HAD PN
005.202  267                   1578          ORA     A
005.203  312 126 007           1579          JZ      CLFT            IF NONE, DEFAULT TO ONE
                                1580
005.206  032                   1581          LDAX    D               GET PN
005.207  267                   1582          ORA     A               SEE IF ZERO
005.210  312 126 007           1583          JZ      CLFT            IF ZERO, DEFAULT TO ONE
                                1584
005.213  107                   1585          MOV     B,A             SAVE COUNT
005.214  315 126 007           1586  ACLFT1  CALL    CLFT            CURSOR LEFT ONE COLUMN
005.217  005                   1587          DCR     B
005.220  040 372               1588          JR      NZ,ACLFT1       TIL DONE
                                1589
005.222  311                   1590          RET


                                1592  **      ACPR - ANSI CURSOR POSITION REPORT
                                1593  *
                                1594  *       *ACPR* OUTPUTS THE CURRENT CURSOR POSITION IN THE ANSI FORM
                                1595  *               ** ESC [ Pl ; Pc R **
                                1596  *
                                1597  *
                                1598  *       ENTRY   NONE
                                1599  *
                                1600  *       EXIT    NONE
                                1601  *
                                1602  *       USES    A,B,C,D,E,H,L,F
                                1603
                                1604
```

```
005.223  170           1605  ACPR    MOV     A,B             SEE IF PN WAS INPUT
005.224  267           1606          ORA     A
005.225  310           1607          RZ                      IF NO PN, ILLEGAL, EXIT
                       1608
005.226  032           1609          LDAX    D               GET PN
005.227  376 006       1610          CPI     6               MUST BE 6 FOR ACPR
005.231  300           1611          RNZ                     IF NOT SIX, ILLEGAL, EXIT
                       1612
005.232  315 100 015   1613          CALL    PSOF            OUTPUT FIRST PART OF STRING
005.235  033 333       1614          DB      ESC,'['+200Q    ESC [
005.237  072 273 100   1615          LDA     CURVP           GET CURRENT LINE NUMBER
005.242  074           1616          INR     A               HOME LINE = 1 FOR ANSI
005.243  315 014 011   1617          CALL    EDD             ENCODE DECIMAL DIGITS
005.246  172           1618          MOV     A,D             GET TENS DIGIT
005.247  267           1619          ORA     A               SEE IF A LEADING ZERO
005.250  050 005       1620          JR      Z,ACPR1         IF ZERO DROP IT
                       1621
005.252  366 060       1622          ORI     00110000B       MAKE BCD INTO ASCII
005.254  315 375 013   1623          CALL    PCOFT           ELSE, PLACE IN FIFO
005.257  173           1624  ACPR1   MOV     A,E             GET ONES DIGIT
005.260  366 060       1625          ORI     00110000B       MAKE BCD INTO ASCII
005.262  315 375 013   1626          CALL    PCOFT           PUT IN FIFO
                       1627
005.265  315 100 015   1628          CALL    PSOF            SEPARATE LINE FROM COLUMN
005.270  273           1629          DB      ';'+200Q
                       1630
005.271  072 272 100   1631          LDA     CURHP           GET CURRENT COLUMN POSITION
005.274  074           1632          INR     A               HOME COLUMN = 1 FOR ANSI
005.275  315 014 011   1633          CALL    EDD             ENCODE DIGITS
005.300  172           1634          MOV     A,D             GET TENS
005.301  267           1635          ORA     A               SEE IF ZERO
005.302  050 005       1636          JR      Z,ACPR2         IF TENS IS ZERO, FORGET IT
                       1637
005.304  366 060       1638          ORI     00110000B       MAKE BCD INTO ASCII
005.306  315 375 013   1639          CALL    PCOFT           OUTPUT TENS
005.311  173           1640  ACPR2   MOV     A,E             GET ONES DIGIT
005.312  366 060       1641          ORI     00110000B       MAKE BCD INTO ASCII
005.314  315 375 013   1642          CALL    PCOFT           OUTPUT ONES
005.317  315 100 015   1643          CALL    PSOF            OUTPUT FINAL
005.322  322           1644          DB      'R'+200Q
005.323  311           1645          RET
```

```
                       1647  **      ACRT - ANSI CURSOR RIGHT
                       1648  *
                       1649  *       *ACRT* MOVES THE CURSOR ONE COLUMN TOWARD THE END OF THE LINE
                       1650  *
                       1651  *
                       1652  *       ENTRY   (B) = ZERO IF NO PN WAS INPUT
                       1653  *               (D,E) = PSDW
                       1654  *
                       1655  *       EXIT    NONE
                       1656  *
                       1657  *       USES    A,B,D,E,H,L,F
```

```
                        1658
                        1659
005.324  170           1660  ACRT    MOV     A,B             SEE IF PN WAS INPUT
005.325  267           1661          ORA     A
005.326  312 201 007    1662          JZ      CRT             IF NO PN, DEFAULT TO ONE
                        1663
005.331  032           1664          LDAX    D               GET PN
005.332  267           1665          ORA     A               SEE IF ZERO
005.333  312 201 007    1666          JZ      CRT             IF ZERO, DEFAULT TO ONE
                        1667
005.336  107           1668          MOV     B,A             SAVE COUNT
005.337  315 201 007    1669  ACRT1   CALL    CRT             CURSOR RIGHT ONE COLUMN
005.342  005           1670          DCR     B
005.343  040 372       1671          JR      NZ,ACRT1        'TIL DONE
                        1672
005.345  311           1673          RET
```

```
                        1675  **      ACUP - ANSI CURSOR UP
                        1676  *
                        1677  *       *ACUP* MOVES THE CURSOR TOWARD THE TOP OF THE SCREEN PN TIMES
                        1678  *
                        1679  *
                        1680  *       ENTRY   (B) = ZERO IF NO PN WAS INPUT
                        1681  *               (D,E) = PSDW
                        1682  *
                        1683  *       EXIT    NONE
                        1684  *
                        1685  *       USES    A,B,C,D,E,H,L,F
                        1686
                        1687
005.346  170           1688  ACUP    MOV     A,B             SEE IF A PN WAS INPUT
005.347  267           1689          ORA     A
005.350  312 227 007    1690          JZ      CUP             IF NONE, DEFAULT TO ONE
                        1691
005.353  032           1692          LDAX    D               GET PN
005.354  267           1693          ORA     A               ZERO?
005.355  312 227 007    1694          JZ      CUP             IF ZERO, DEFAULT TO ONE
                        1695
005.360  365           1696  ACUP1   PUSH    PSW             SAVE PSW
005.361  315 227 007    1697          CALL    CUP             CURSOR UP ONE LINE
005.364  361           1698          POP     PSW
005.365  075           1699          DCR     A               COUNT - 1
005.366  040 370       1700          JR      NZ,ACUP1        'TIL DONE
                        1701
005.370  311           1702          RET
```

```
                                1705  **      APCA - ANSI PERFORM CURSOR ADDRESSING
                                1706  *
                                1707  *       *APCA* CHECKS TO SEE THAT THE LINE AND COLUMN VALUES ARE LEGAL
                                1708  *       AND SETS THE CURSOR TO THE REQUESTED ADDRESS.  IF THE VALUES
                                1709  *       ARE OUT OF RANGE, THE OLD LINE NUMBER REMAINS AND/OR THE CURSOR
                                1710  *       GOES TO THE LAST COLUMN ON THE LINE
                                1711  *
                                1712  *
                                1713  *       ENTRY   (B) = ZERO IF NO PN WAS INPUT
                                1714  *               (D,E) = PSDW
                                1715  *
                                1716  *       EXIT    NONE
                                1717  *
                                1718  *       USES    A,B,C,D,E,H,L,F
                                1719
                                1720
005.371  170                   1721  APCA    MOV     A,B             SEE IF PS WAS INPUT
005.372  267                   1722          ORA     A
005.373  312 271 015           1723          JZ      SCH             IF NO ADDRESS SPECIFIED, PLACE CURSOR HOME
                                1724
005.376  353                   1725          XCHG                    (H,L) = PS WORK AREA
005.377  176                   1726          MOV     A,M             GET LINE NUMBER
006.000  267                   1727          ORA     A               SEE IF ZERO
006.001  050 001               1728          JR      Z,APCA1         IF ZERO, THIS IS FIRST LINE
                                1729
006.003  075                   1730          DCR     A               ELSE, LINE = LINE#-1
006.004  376 030               1731  APCA1   CPI     24              LINE IN RANGE?
006.006  070 013               1732          JR      C,APCA2         IF 0 TO 23
                                1733
006.010  040 014               1734          JR      NZ,APCA3        IF NOT REQUESTING 25TH LINE
                                1735
006.012  107                   1736          MOV     B,A             SAVE LINE NUMBER
006.013  072 311 100           1737          LDA     MODEI           GET MODE FLAGS
006.016  346 200               1738          ANI     MI.25L          25TH LINE ENABLED?
006.020  050 004               1739          JR      Z,APCA3         IF 25TH LINE IS OFF
                                1740
006.022  170                   1741          MOV     A,B
006.023  062 273 100           1742  APCA2   STA     CURVP           SET NEW LINE NUMBER
                                1743
006.026  043                   1744  APCA3   INX     H               GET NEXT PN
006.027  176                   1745          MOV     A,M             COLUMN NUMBER
006.030  267                   1746          ORA     A               ZERO?
006.031  050 001               1747          JR      Z,APCA4         IF ZERO
                                1748
006.033  075                   1749          DCR     A               ELSE, COLUMN = COLUMN#-1
006.034  376 120               1750  APCA4   CPI     80              SEE IF IN RANGE
006.036  070 002               1751          JR      C,APCA5         IF 0 TO 79
                                1752
006.040  076 117               1753          MVI     A,79            ELSE, SET LAST COLUMN
006.042  062 272 100           1754  APCA5   STA     CURHP           SET COLUMN POSITION
006.045  303 322 015           1755          JMP     SNCP            SET NEW CURSOR POSITION
```

```
                        1757  **      APDC - ANSI PERFORM DELETE CHARACTER
                        1758  *
                        1759  *       *APDC* DELETES THE SPECIFIED NUMBER OF CHARACTERS FROM THE LINE
                        1760  *
                        1761  *
                        1762  *       ENTRY   (B) = ZERO IF NO PN WAS INPUT
                        1763  *               (D,E) = PSDW
                        1764  *
                        1765  *       EXIT    NONE
                        1766  *
                        1767  *       USES    A,B,C,D,E,H,L,F
                        1768
                        1769
006.050  170           1770  APDC    MOV     A,B             SEE IF INPUT PN
006.051  267           1771          ORA     A
006.052  312 027 014   1772          JZ      PDC             IF NO PN, DEFAULT TO ONE
                        1773
006.055  032           1774          LDAX    D               GET PN
006.056  267           1775          ORA     A               ZERO?
006.057  312 027 014   1776          JZ      PDC             IF ZERO, DEFAULT TO ONE
                        1777
006.062  365           1778  APDC1   PUSH    PSW             SAVE PN
006.063  315 027 014   1779          CALL    PDC
006.066  361           1780          POP     PSW
006.067  075           1781          DCR     A
006.070  040 370       1782          JR      NZ,APDC1        'TIL DONE
                        1783
006.072  311           1784          RET


                        1786  **      APDL - ANSI PERFORM DELETE LINE
                        1787  *
                        1788  *       *APDL* DELETES THE SPECIFIED NUMBER OF FOLLOWING LINES
                        1789  *
                        1790  *
                        1791  *       ENTRY   (B) = ZERO IF NO PN WAS INPUT
                        1792  *               (D,E) = PSDW
                        1793  *
                        1794  *       EXIT    NONE
                        1795  *
                        1796  *       USES    A,B,C,D,E,H,L,F
                        1797
                        1798
006.073  170           1799  APDL    MOV     A,B             SEE IF PN WAS INPUT
006.074  267           1800          ORA     A
006.075  312 075 014   1801          JZ      PDL             IF NO PN, DEFAULT TO ONE
                        1802
006.100  032           1803          LDAX    D               GET PN
006.101  267           1804          ORA     A               ZERO?
006.102  312 075 014   1805          JZ      PDL             IF ZERO, DEFAULT TO ONE
                        1806
006.105  365           1807  APDL1   PUSH    PSW             SAVE COUNT
006.106  315 075 014   1808          CALL    PDL             DELETE ONE LINE
006.111  361           1809          POP     PSW
```

```
006.112  075           1810          DCR     A               COUTN -1
006.113  040 370       1811          JR      NZ,APDL1        'TIL DONE
                       1812
006.115  311           1813          RET


                       1815  **      APIL - ANSI PERFORM INSERT LINE
                       1816  *
                       1817  *       *APIL* INSERTS THE SPECIFIED NUMBER OF LINES AT THE CURRENT LINE
                       1818  *       OF THE CURSOR POSITION
                       1819  *
                       1820  *
                       1821  *       ENTRY   (B) = ZERO IF NO PN WAS INPUT
                       1822  *               (D,E) = PSDW
                       1823  *
                       1824  *       EXIT    NONE
                       1825  *
                       1826  *       USES    A,B,C,D,E,H,L,F
                       1827
                       1828
006.116  170           1829  APIL    MOV     A,B             SEE IF PN WAS INPUT
006.117  267           1830          ORA     A
006.120  312 234 014   1831          JZ      PIL             IF NO PN, INSERT ONE LINE
                       1832
006.123  032           1833          LDAX    D               GET PN
006.124  267           1834          ORA     A               ZERO?
006.125  312 234 014   1835          JZ      PIL             IF ZERO, DEFAULT TO ONE LINE
                       1836
006.130  365           1837  APIL1   PUSH    PSW             SAVE COUNT
006.131  315 234 014   1838          CALL    PIL             INSERT ONE LINE
006.134  361           1839          POP     PSW
006.135  075           1840          DCR     A
006.136  040 370       1841          JR      NZ,APIL1        'TIL DONE
                       1842
006.140  311           1843          RET


                       1845  **      ARM - ANSI RESET MODE
                       1846  *
                       1847  *       *ARM* RESETS THE MODE(S) SPECIFIED BY PN FOR THE SEQUENCE  ESC [ Pn l
                       1848  *
                       1849  *
                       1850  *       ENTRY   (B) = ZERO IF NO PN WAS INPUT
                       1851  *               (D,E) = PSDW
                       1852  *
                       1853  *       EXIT    NONE
                       1854  *
                       1855  *       USES    A,B,C,D,E,H,L,F
                       1856
                       1857
006.141  170           1858  ARM     MOV     A,B             SEE IF PN WAS INPUT
006.142  267           1859          ORA     A
```

```
006.143  310            1860            RZ                      IF NO PN, EXIT
                         1861
006.144  353            1862            XCHG                    (H,L) = PSDW
006.145  001 310 100    1863            LXI     B,MODEB         (B,C) = MODEB
006.150  021 307 100    1864  ARM1      LXI     D,MODEA         (D,E) = MODEA
006.153  176            1865            MOV     A,M             GET PN
006.154  376 002        1866            CPI     2               PN = 2?
006.156  314 125 011    1867            CZ      EKI             IF 2, ENABLE KEYBOARD INPUT
                         1868
006.161  176            1869            MOV     A,M             GET PN
006.162  376 004        1870            CPI     4               PN = 4?
006.164  345            1871            PUSH    H               SAVE PN POINTER
006.165  314 305 016    1872            CZ      XICM            IF 4, EXIT INSERT CHARACTER MODE
006.170  341            1873            POP     H
                         1874
006.171  176            1875            MOV     A,M             GET PN
006.172  376 024        1876            CPI     20              PN = 20?
006.174  314 263 016    1877            CZ      XACR            IF 20, EXIT AUTO CARRIAGE RETURN
                         1878
006.177  176            1879            MOV     A,M             GET PN
006.200  376 154        1880            CPI     '1'             SEE IF PN = FINAL
006.202  310            1881            RZ                      IF FINAL
                         1882
006.203  043            1883            INX     H               ELSE, POINT TO NEXT PN
006.204  030 342        1884            JR      ARM1
```

```
                         1886  **        ASBR - ANSI SET BAUD RATE
                         1887  *
                         1888  *         *ASBR* SETS THE BAUD RATE TO THAT SPECIFIED BY PN
                         1889  *
                         1890  *
                         1891  *         ENTRY   (B) = ZERO IF NO PN WAS INPUT
                         1892  *                 (D,E) = PSDW
                         1893  *
                         1894  *         EXIT    NONE
                         1895  *
                         1896  *         USES    A,B,C,D,E,H,L,F
                         1897
                         1898
006.206  170            1899  ASBR      MOV     A,B             SEE IF PN WAS INPUT
006.207  267            1900            ORA     A
006.210  050 005        1901            JR      Z,ASBR1         IF NONE SPECIFIED, DEFAULT TO 110 BAUD
                         1902
006.212  353            1903            XCHG                    (H,L) = PSDW
006.213  176            1904            MOV     A,M             GET PN
006.214  376 016        1905            CPI     14              IN RANGE?
006.216  320            1906            RNC                     IF NOT, EXIT
                         1907
006.217  303 172 012    1908  ASBR1     JMP     SBR.
```

```
                                1910  **      ASGM - ANSI SET GRAPHICS MODE
                                1911  *
                                1912  *       *ASGM* SETS OR RESETS THE GRAPHICS MODE AND/OR THE REVERSE VIDEO MODE
                                1913  *
                                1914  *
                                1915  *       ENTRY   (B) = ZERO IF NO PN WAS INPUT
                                1916  *               (D,E) = PSDW
                                1917  *
                                1918  *       EXIT    NONE
                                1919  *
                                1920  *       USES    A,D,E,H,L,F
                                1921
                                1922
006.222  170                   1923  ASGM    MOV     A,B             SEE IF PN WAS INPUT
006.223  267                   1924          ORA     A
006.224  050 011               1925          JR      Z,ASGM1.5       IF NO PN, DEFAULT TO REVERSE VIDEO OFF
                                1926
006.226  325                   1927          PUSH    D               SAVE PN POINTER
                                1928
006.227  321                   1929  ASGM1   POP     D               GET PN POINTER
006.230  032                   1930          LDAX    D               GET PN
006.231  023                   1931          INX     D               POINT TO NEXT
006.232  325                   1932          PUSH    D               SAVE FOR NEXT PASS(S)
006.233  041 227 006           1933          LXI     H,ASGM1         SET RETURN ADDRESS
006.236  345                   1934          PUSH    H
                                1935
006.237  041 265 006           1936  ASGM1.5 LXI     H,ASGMT         (H,L) = GRAPHIC MODE TABLE ADDRESS
006.242  026 004               1937          MVI     D,ASGMTL        (D) = TABLE LENGTH
006.244  036 003               1938          MVI     E,ASGMTW        (E) = TABLE WIDTH
006.246  315 363 015           1939          CALL    STAB            SEARCH TABLE FOR PN VALUE
006.251  070 007               1940          JR      C,ASGM2         IF NOT IN TABLE, MUST BE BAD NO. OR FINAL
                                1941
006.253  043                   1942          INX     H               ELSE, IN TABLE, GO TO ROUTINE
006.254  146                   1943          MOV     H,M             GET MSB
006.255  157                   1944          MOV     L,A             LSB
006.256  021 307 100           1945          LXI     D,MODEA         (D,E) = MODEA FOR ROUTINES
006.261  351                   1946          PCHL                    GO TO ADDRESS IN TABLE
                                1947
006.262  341                   1948  ASGM2   POP     H               TOSS FORCED RETURN ADDRESS
006.263  321                   1949          POP     D               EVEN STACK
006.264  311                   1950          RET


                                1952  *       ASGMT - ANSI SET GRAPHICS MODE TABLE
                                1953  *
                                1954  *       *ASGMT* CONTAINS THE ADDRESS OF THE ROUTINES WHICH SET
                                1955  *       THE REQUESTED GRAPHIC OR REVERSE VIDEO MODE
                                1956
                                1957
006.265                         1958  ASGMT   EQU     *
                                1959
006.265  000                   1960          DB      0               ZERO
006.266  325 017               1961          DW      XRVM            EXIT REVERSE VIDEO MODE
                                1962
006.270  007                   1963          DB      7               SEVEN
```

```
006.271  361 011       1964            DW      ERVM            ENTER REVERSE VIDEO MODE
                       1965
006.273  012           1966            DB      10              TEN
006.274  061 011       1967            DW      EGM             ENTER GRAPHICS MODE
                       1968
006.276  013           1969            DB      11              ELEVEN
006.277  275 016       1970            DW      XGM             EXIT GRAPHICS MODE
                       1971
000.003                1972  ASGMTW    EQU     3               TABLE IS THREE BYTES WIDE
000.004                1973  ASGMTL    EQU     *-ASGMT/ASGMTW  TABLE LENGTH


                       1975  **        ASM - ANSI SET MODE
                       1976  *
                       1977  *         *ASM* SETS THE MODE(S) SPECIFIED BY PN
                       1978  *
                       1979  *
                       1980  *         ENTRY   (B) = ZERO IF NO PN WAS INPUT
                       1981  *                 (D,E) = PSDW
                       1982  *
                       1983  *         EXIT    NONE
                       1984  *
                       1985  *         USES    A,B,C,D,E,H,L,F
                       1986
                       1987
006.301  170           1988  ASM       MOV     A,B             SEE IF PN WAS INPUT
006.302  267           1989            ORA     A
006.303  310           1990            RZ                      IF NO PN
                       1991
006.304  353           1992            XCHG                    (H,L) = PSDW
006.305  001 310 100   1993            LXI     B,MODEB         (B,C) = MODEB
006.310  021 307 100   1994  ASM1      LXI     D,MODEA         (D,E) = MODEA
006.313  176           1995            MOV     A,M             GET PN
006.314  376 002       1996            CPI     2               PN = 2?
006.316  314 226 010   1997            CZ      DKI             IF 2, DISABLE KEYBOARD INPUT
                       1998
006.321  176           1999            MOV     A,M             GET PN
006.322  345           2000            PUSH    H               SAVE PN POINTER
006.323  376 004       2001            CPI     4               PN = 4?
006.325  314 107 011   2002            CZ      EICM            IF 4, ENTER INSERT CHARACTER MODE
006.330  341           2003            POP     H
                       2004
006.331  176           2005            MOV     A,M             GET PN
006.332  376 024       2006            CPI     20              PN = 20?
006.334  314 276 010   2007            CZ      EACR            IF 20, ENTER AUTO CARRIAGE RETURN
                       2008
006.337  176           2009            MOV     A,M             GET PN
006.340  376 150       2010            CPI     'h'             SEE IF PN = FINAL
006.342  310           2011            RZ                      IF FINAL
                       2012
006.343  043           2013            INX     H               ELSE, POINT TO NEXT PN
006.344  030 342       2014            JR      ASM1
```

```
                                2017  **      CDN - CURSOR DOWN
                                2018  *
                                2019  *       *CDN* MOVES THE CURSOR DOWN ONE LINE ON THE DISPLAY, BUT DOES
                                2020  *       NOT CAUSE A SCROLL PAST THE LAST LINE
                                2021  *
                                2022  *
                                2023  *       ENTRY   NONE
                                2024  *
                                2025  *       EXIT    NONE
                                2026  *
                                2027  *       USES    A,B,C,D,E,H,L,F
                                2028
                                2029
006.346  072 273 100           2030  CDN     LDA     CURVP           GET CURRENT VERTICAL POSITION
006.351  376 027               2031          CPI     23              ON LAST LINE(S)?
006.353  320                   2032          RNC                     IF SO, EXIT
                                2033
                                2034  *       JMP     PLF             ELSE, DO A LINE FEED
000.000                         2035          ERRNZ   *-PLF



                                2037  **      PLF - PERFORM LINE FEED
                                2038  *
                                2039  *       *PLF* MOVES THE CURSOR DOWN ONE LINE IN THE DISPLAY MEMORY.  IF
                                2040  *       THE CURSOR WAS ON THE 23RD LINE, THE DISPLAYED VIDEO IS SCROLLED
                                2041  *       UP ONE LINE AND THE NEW LINE IS WRITTEN FULL OF SPACES.
                                2042  *
                                2043  *
                                2044  *       ENTRY   NONE
                                2045  *
                                2046  *       EXIT    NONE
                                2047  *
                                2048  *       USES    A,B,C,D,E,H,L,F
                                2049
                                2050
006.354  072 273 100           2051  PLF     LDA     CURVP           GET VERTICAL POSITION
006.357  376 030               2052          CPI     24              ON 25TH LINE?
006.361  310                   2053          RZ                      IF SO, EXIT WITHOUT ANY ACTION
                                2054
                                2055
006.362  021 120 000           2056  PLF0.2  LXI     D,80            LINES = 80 CHARACTERS
006.365  052 270 100           2057          LHLD    CLSA            GET CURRENT LINE STARTING ADDRESS
006.370  031                   2058          DAD     D               ADD A LINE
006.371  174                   2059          MOV     A,H             STAY IN VIDEO RAM
006.372  366 370               2060          ORI     VRAMS/256
006.374  147                   2061          MOV     H,A
006.375  042 270 100           2062          SHLD    CLSA            SET NEW LINE VALUE
007.000  072 273 100           2063          LDA     CURVP           GET CURSOR VERTICAL POSITION
007.003  376 027               2064          CPI     23              ON LAST LINE?
007.005  040 061               2065          JR      NZ,PLF1         IF NOT ON LAST LINE
                                2066
                                2067  *       SCROLL VIDEO TO DISPLAY A NEW LINE
                                2068  *
007.007  072 311 100           2069          LDA     MODEI           GET MODE FLAGS
```

```
007.012  346 200      2070          ANI     MI.25L          SEE IF 25TH LINE IS ENABLED
007.014  050 016      2071          JR      Z,PLF0.5        IF NOT TO WORRY ABOUT 25TH LINE
                      2072
007.016  345          2073          PUSH    H               SAVE NEW LINE STARTING ADDRESS
007.017  021 120 000  2074          LXI     D,80            ADD 80 FOR BEGINNING OF 26TH LINE
007.022  031          2075          DAD     D
007.023  353          2076          XCHG                    (D,E) = BEGINNING OF 26TH LINE
007.024  341          2077          POP     H               (H,L) = BEGINNING OF 25TH LINE
007.025  345          2078          PUSH    H               SAVE AGAIN FOR LATER
007.026  006 005      2079          MVI     B,80/16         SET MOD 16 COUNT FOR ONE LINE
007.030  315 040 010  2080          CALL    CPM16           COPY LINE
007.033  341          2081          POP     H               (H,L) = BEGINNING OF 25TH LINE
                      2082
007.034  006 005      2083  PLF0.5  MVI     B,80/16         SET MOD-16 ERASE COUNT TO 5 (80 CHARACTERS)
007.036  315 212 016  2084          CALL    WSVA            WRITE 80 SPACES TO VIDEO TO BLANK THIS LINE
007.041  052 266 100  2085          LHLD    SHOME           GET OLD HOME POSITION
007.044  021 120 000  2086          LXI     D,80            ADD ONE LINE
007.047  031          2087          DAD     D
007.050  174          2088          MOV     A,H             STAY IN VIDEO RAM
007.051  366 370      2089          ORI     VRAMS/256
007.053  147          2090          MOV     H,A
007.054  042 266 100  2091          SHLD    SHOME           UPDATE HOME POSITION
007.057  174          2092          MOV     A,H             SET VALUE OF HOME POSITION FOR CRTC
007.060  346 007      2093          ANI     HOMAX/256
007.062  147          2094          MOV     H,A
007.063  042 301 100  2095          SHLD    VI.SA
007.066  030 004      2096          JR      PLF2            UPDATE CURSOR ADDRESS
                      2097
007.070  074          2098  PLF1    INR     A               ADD ONE TO LINE COUNTER
007.071  062 273 100  2099          STA     CURVP
                      2100
007.074  052 274 100  2101  PLF2    LHLD    CURAD           GET CURRENT CURSOR POSITION
007.077  031          2102          DAD     D               ADD ONE LINE
007.100  174          2103          MOV     A,H             STAY IN VIDEO RAM
007.101  366 370      2104          ORI     VRAMS/256
007.103  147          2105          MOV     H,A
007.104  042 274 100  2106          SHLD    CURAD           UPDATE CURSOR ADDRESS
                      2107
                      2108
007.107  303 146 016  2109          JMP     UCP.            CAUSE NMI TO UPDATE SCREEN IN CASE OF SCROLL



                      2111  **      PLFCR - PERFORM LINE FEED AND/OR CARRIAGE RETURN
                      2112  *
                      2113  *       *PLFCR* PERFORMS A CARRIAGE RETURN PRIOR TO PERFORMING A LINE
                      2114  *       FEED IF THE AUTO CARRIAGE RETURN FUNCTION IS SELECTED
                      2115  *
                      2116  *
                      2117  *       ENTRY   NONE
                      2118  *
                      2119  *       EXIT    NONE
                      2120  *
                      2121  *       USES    A,B,C,D,E,H,L,F
                      2122
```

```
                                2123
007.112  315 354 006           2124  PLFCR   CALL    PLF             DO LINE FEED
007.115  072 310 100           2125          LDA     MODEB           GET MODE FLAGS
007.120  346 020               2126          ANI     MB.ACR          SEE IF ACR SELECTED
007.122  302 013 014           2127          JNZ     PCR             IF SELECTED
                                2128
007.125  311                   2129          RET                     ELSE, JUST RETURN


                                2131  **      PBS - PERFORM BACKSPACE
                                2132  *
                                2133  *       CLFT - CURSOR LEFT
                                2134  *
                                2135  *       *PBS*-*CLFT* STEPS THE CURSOR ONE POSITION TO THE LEFT, BUT DOES NOT
                                2136  *       WRAP AROUND TO THE PREVIOUS LINE AFTER REACHING COLUMN ZERO
                                2137  *
                                2138  *
                                2139  *       ENTRY   NONE
                                2140  *
                                2141  *       EXIT    NONE
                                2142  *
                                2143  *       USES    A,H,L,F
                                2144
                                2145
007.126                         2146  PBS     EQU     *
007.126                         2147  CLFT    EQU     *
007.126  072 272 100           2148          LDA     CURHP           GET CURRENT POSITION ON LINE
007.131  267                   2149          ORA     A               SEE IF COLUMN ZERO
007.132  310                   2150          RZ                      IF AT BEGINNING OF LINE
                                2151
007.133  075                   2152          DCR     A               ELSE, DECREMENT CURSOR POSITION
007.134  062 272 100           2153          STA     CURHP
007.137  052 274 100           2154          LHLD    CURAD           GET CURSOR ADDRESS
007.142  053                   2155          DCX     H               DECREMENT ADDRESS
007.143  174                   2156  PBS1    MOV     A,H             STAY IN VIDEO RAM
007.144  366 370               2157          ORI     VRAMS/256
007.146  147                   2158          MOV     H,A
007.147  042 274 100           2159          SHLD    CURAD
                                2160
                                2161
007.152  311                   2162          RET                     EXIT WITH OR WITHOUT


                                2164  **      CPR - CURSOR POSITION REPORT
                                2165  *
                                2166  *       *CPR* OUTPUTS THE CURSOR POSITION IN THE HEATH FORMAT
                                2167  *               ** ESC Y Pl Pc **
                                2168  *
                                2169  *
                                2170  *       ENTRY   NONE
                                2171  *
                                2172  *       EXIT    NONE
```

```
                                2173  *
                                2174  *       USES    A,B,C,H,L,F
                                2175
                                2176
007.153  315 100 015           2177  CPR     CALL    PSOF            OUTPUT FIRST PART OF REPORT
007.156  033 331               2178          DB      ESC,'Y'+200Q    ESC Y
007.160  072 273 100           2179          LDA     CURVP           GET CURRENT LINE NUMBER
007.163  306 040               2180          ADI     40Q             MAKE ASCII
007.165  315 375 013           2181          CALL    PCOFT           PLACE LINE NUMBER IN FIFO
007.170  072 272 100           2182          LDA     CURHP           GET CURRENT COLUMN NUMBER
007.173  306 040               2183          ADI     40Q
007.175  315 375 013           2184          CALL    PCOFT           PUT IN FIFO
007.200  311                   2185          RET


                                2187  **      CRT - CURSOR RIGHT
                                2188  *
                                2189  *       *CRT* MOVES THE CURSOR RIGHT ONE COLUMN.  IF ALREADY AT THE LAST
                                2190  *       COLUMN, THE CURSOR IS ADVANCED TO THE BEGINNING OF THE NEXT LINE.
                                2191  *       *CRT* WILL NOT ADVANCE THE CURSOR PAST COLUMN 79 OF LINE 23.
                                2192  *
                                2193  *
                                2194  *       ENTRY   NONE
                                2195  *
                                2196  *       EXIT    NONE
                                2197  *
                                2198  *       USES    A,H,L,F
                                2199
                                2200
007.201  072 272 100           2201  CRT     LDA     CURHP           GET CURRENT HORIZONTAL POSITION
007.204  376 117               2202          CPI     79              CURSOR AT LAST COLUMN?
007.206  310                   2203          RZ                      EXIT IF AT END OF LINE
                                2204
007.207  074                   2205          INR     A               ELSE, INCREMENT CURSOR POSITION
007.210  062 272 100           2206          STA     CURHP
007.213  052 274 100           2207          LHLD    CURAD           GET CURSOR ADDRESS
007.216  043                   2208          INX     H               INCREMENT ADDRESS
007.217  174                   2209          MOV     A,H             STAY IN VIDEO RAM
007.220  366 370               2210          ORI     VRAMS/256
007.222  147                   2211          MOV     H,A
007.223  042 274 100           2212          SHLD    CURAD           UPDATE CURAD
                                2213
                                2214
007.226  311                   2215          RET                     EXIT REGARDLESS
```

```
                                2217  **      CUP - CURSOR UP
                                2218  *
                                2219  *       *CUP* MOVES THE CURSOR UP ONE LINE ON THE DISPLAY.  *CUP* WILL
                                2220  *       NOT MOVE THE CURSOR PAST LINE ZERO
                                2221  *
                                2222  *
                                2223  *       ENTRY   NONE
                                2224  *
                                2225  *       EXIT    NONE
                                2226  *
                                2227  *       USES    A,B,C,D,E,H,L,F
                                2228
                                2229
007.227  072 273 100            2230  CUP     LDA     CURVP           GET CURRENT VERTICAL POSITION
007.232  267                    2231          ORA     A               CHECK FOR LINE ZERO
007.233  310                    2232          RZ                      IF LINE ZERO, EXIT WITHOUT MOVING CURSOR
                                2233
                                2234  *       JMP     PRLF            DO A REVERSE LINE FEED
000.000                         2235          ERRNZ   *-PRLF



                                2237  **      PRLF - PERFORM A REVERSE LINE FEED
                                2238  *
                                2239  *       *PRLF* MOVES THE CURSOR UP ONE LINE IN THE VIDEO MEMORY.  IF THE
                                2240  *       CURSOR WAS ALREADY ON THE TOP LINE OF THE DISPLAY, THE DISPLAY IS
                                2241  *       SCROLLED DOWN AND THE NEW LINE IS FILLED WITH SPACES.
                                2242  *
                                2243  *
                                2244  *       ENTRY   NONE
                                2245  *
                                2246  *       EXIT    NONE
                                2247  *
                                2248  *       USES    A,B,C,D,E,H,L,F
                                2249
                                2250
007.234  072 273 100            2251  PRLF    LDA     CURVP           GET VERTICAL POSITION
007.237  376 030                2252          CPI     24              25TH LINE?
007.241  310                    2253          RZ                      IF SO, FORGET RLF
                                2254
                                2255
007.242  021 260 377            2256  PRLF0.2 LXI     D,-80           (D,E) = TWOS COMPLEMENT OF 80
007.245  052 270 100            2257          LHLD    CLSA            GET CURRENT LINE ADDRESS
007.250  031                    2258          DAD     D               SUBTRACT ONE LINE
007.251  174                    2259          MOV     A,H             STAY IN VIDEO RAM
007.252  366 370                2260          ORI     VRAMS/256
007.254  147                    2261          MOV     H,A
007.255  042 270 100            2262          SHLD    CLSA            UPDATE LINE ADDRESS
007.260  072 273 100            2263          LDA     CURVP           GET CURRENT DISPLAYED LINE NUMBER
007.263  267                    2264          ORA     A               CHECK FOR LINE ZERO
007.264  040 067                2265          JR      NZ,PRLF1        IF NOT ON LINE ZERO
                                2266
007.266  072 311 100            2267          LDA     MODE1           GET MODE FLAGS
007.271  346 200                2268          ANI     M1.25L          IS 25TH LINE ON?
007.273  050 021                2269          JR      Z,PRLF0.5       IF NOT ON
```

```
                        2270
                        2271  *       25TH LINE IS ON, COPY TO 24TH BEFORE DISPLAY IS MOVED
                        2272  *
007.275  052 266 100    2273          LHLD    SHOME           GET HOME ADDRESS
007.300  021 060 007    2274          LXI     D,23*80         FIND ADDRESS OF LINE #23
007.303  031            2275          DAD     D
007.304  353            2276          XCHG                    (D,E) = BEGINNING OF LINE 23
007.305  041 120 000    2277          LXI     H,80            FIND ADDRESS OF LINE #24
007.310  031            2278          DAD     D
007.311  006 005        2279          MVI     B,80/16         COPY ONE LINE
007.313  315 040 010    2280          CALL    CPM16
                        2281
                        2282  *       ERASE TOP LINE
                        2283  *
007.316  052 270 100    2284  PRLF0.5 LHLD    CLSA            LINE ZERO ADDRESS TO (H,L) FOR ERASE
007.321  006 005        2285          MVI     B,5             ERASE LINE (5*16 SPACES)
007.323  315 212 016    2286          CALL    WSVA
007.326  052 266 100    2287          LHLD    SHOME           GET OLD HOME POSITION
007.331  021 260 377    2288          LXI     D,-80
007.334  031            2289          DAD     D               SUBTRACT ONE LINE
007.335  174            2290          MOV     A,H             STAY IN VIDEO RAM
007.336  366 370        2291          ORI     VRAMS/256
007.340  147            2292          MOV     H,A
007.341  042 266 100    2293          SHLD    SHOME           UPDATE HOME POSITION
007.344  174            2294          MOV     A,H             SET HOME POSITION FOR CRTC
007.345  346 007        2295          ANI     HOMAX/256
007.347  147            2296          MOV     H,A
007.350  042 301 100    2297          SHLD    VI.SA
007.353  030 004        2298          JR      PRLF2           UPDATE CURSOR
                        2299
007.355  075            2300  PRLF1   DCR     A               DECREMENT LINE COUNTER
007.356  062 273 100    2301          STA     CURVP
                        2302
007.361  052 274 100    2303  PRLF2   LHLD    CURAD           GET CURRENT CURSOR ADDRESS
007.364  031            2304          DAD     D               SUBTRACT ONE LINE
007.365  174            2305          MOV     A,H             STAY IN VIDEO RAM
007.366  366 370        2306          ORI     VRAMS/256
007.370  147            2307          MOV     H,A
007.371  042 274 100    2308          SHLD    CURAD           UPDATE CURSOR ADDRESS
                        2309
                        2310
007.374  303 146 016    2311          JMP     UCP.            EXIT


                        2313  **      EID - ERASE IN DISPLAY
                        2314  *
                        2315  *       *EID* ERASES THE AMOUNT OF THE SCREEN SPECIFIED BY PN
                        2316  *
                        2317  *
                        2318  *       ENTRY   (B) = ZERO IF NO PN WAS INPUT
                        2319  *               (D,E) = PSDW
                        2320  *
                        2321  *       EXIT    NONE
                        2322  *
```

```
                        2323  *       USES    A,B,C,D,E,H,L,F
                        2324
                        2325
007.377  170           2326  EID     MOV     A,B             SEE IF PN WAS INPUT
010.000  267           2327          ORA     A
010.001  312 317 011   2328          JZ      ERM             IF NO PN, ERASE TO END OF SCREEN
                        2329
010.004  353           2330          XCHG                    (H,L) = PSDW
010.005  176           2331          MOV     A,M             GET PN
010.006  267           2332          ORA     A               ZERO?
010.007  312 317 011   2333          JZ      ERM             IF ZERO, ERASE TO END OF PAGE
                        2334
010.012  075           2335          DCR     A               ONE?
010.013  312 315 010   2336          JZ      EBD             IF ONE, ERASE BEGINNING OF DISPLAY
                        2337
010.016  075           2338          DCR     A               TWO?
010.017  300           2339          RNZ                     IF NOT TWO, EXIT
                        2340
                        2341  *       JMP     CLR             ELSE, CLEAR DISPLAY
000.000                 2342          ERRNZ   *-CLR
```

```
                        2344  **      CLR - CLEAR
                        2345  *
                        2346  *       *CLR* PLACES THE SOFTWARE HOME POSITION BACK TO THE HARDWARE
                        2347  *       HOME POSITION (BEGINNING OF VIDEO RAM) AND WRITES ASCII SPACES
                        2348  *       INTO THE FIRST 1920 BYTES (24 LINES X 80 CHARACTERS)
                        2349  *
                        2350  *
                        2351  *       ENTRY   NONE
                        2352  *
                        2353  *       EXIT    NONE
                        2354  *
                        2355  *       USES    A,B,C,D,E,H,L,F
                        2356
                        2357
010.020  072 273 100   2358  CLR     LDA     CURVP           GET VERTICAL POSITION
010.023  376 030       2359          CPI     24              SEE IF ON 25TH LINE
010.025  312 051 011   2360          JZ      EEL             IF SO, JUST ERASE LINE
                        2361
                        2362
010.030  315 271 015   2363          CALL    SCH             SET CURSOR TO HOME POSITION (H,L) = CURAD
010.033  006 170       2364          MVI     B,1920/16       ERASE 24 LINES WORTH OF CHARACTERS
010.035  303 212 016   2365          JMP     WSVA
```

```
                        2367  **      CPM16 - COPY MEMORY - MODULO SIXTEEN
                        2368  *
                        2369  *       *CPM16* COPYS SIXTEEN BYTES FROM AND TO VIDEO RAM.  FROM AND TO
                        2370  *       ADDRESSES MUST BE AT THE BEGINNING OF A LINE.
                        2371  *
                        2372  *
                        2373  *       ENTRY   (H,L) = ADDRESS TO COPY FROM
                        2374  *               (D,E) = ADDRESS TO COPY TO
                        2375  *                 (B) = NUMBER OF BYTES TO COPY /16
                        2376  *
                        2377  *       EXIT    NONE
                        2378  *
                        2379  *       USES    A,B,D,E,H,L,F
                        2380
010.040  172           2381  CPM16   MOV     A,D             KEEP BOTH ADDRESSES IN VIDEO RAM AREA
010.041  366 370       2382          ORI     VRAMS/256
010.043  127           2383          MOV     D,A
010.044  174           2384          MOV     A,H
010.045  366 370       2385          ORI     VRAMS/256
010.047  147           2386          MOV     H,A
                        2387
                        2388  *       COPY SIXTEEN BYTES
                        2389  *
010.050  176           2390          MOV     A,M             GET BYTE FROM FIRST LINE
010.051  022           2391          STAX    D               PUT IN SAME COLUMN ON OTHER LINE
010.052  043           2392          INX     H               POINT TO NEXT COLUMN ON BOTH LINES
010.053  023           2393          INX     D
                        2394
010.054  176           2395          MOV     A,M             *2
010.055  022           2396          STAX    D
010.056  043           2397          INX     H
010.057  023           2398          INX     D
                        2399
010.060  176           2400          MOV     A,M             *3
010.061  022           2401          STAX    D
010.062  043           2402          INX     H
010.063  023           2403          INX     D
                        2404
010.064  176           2405          MOV     A,M             *4
010.065  022           2406          STAX    D
010.066  043           2407          INX     H
010.067  023           2408          INX     D
                        2409
010.070  176           2410          MOV     A,M             *5
010.071  022           2411          STAX    D
010.072  043           2412          INX     H
010.073  023           2413          INX     D
                        2414
010.074  176           2415          MOV     A,M             *6
010.075  022           2416          STAX    D
010.076  043           2417          INX     H
010.077  023           2418          INX     D
                        2419
010.100  176           2420          MOV     A,M             *7
010.101  022           2421          STAX    D
010.102  043           2422          INX     H
```

```
010.103  023          2423          INX     D
                       2424
010.104  176          2425          MOV     A,M             *8
010.105  022          2426          STAX    D
010.106  043          2427          INX     H
010.107  023          2428          INX     D
                       2429
010.110  176          2430          MOV     A,M             *9
010.111  022          2431          STAX    D
010.112  043          2432          INX     H
010.113  023          2433          INX     D
                       2434
010.114  176          2435          MOV     A,M             *10
010.115  022          2436          STAX    D
010.116  043          2437          INX     H
010.117  023          2438          INX     D
                       2439
010.120  176          2440          MOV     A,M             *11
010.121  022          2441          STAX    D
010.122  043          2442          INX     H
010.123  023          2443          INX     D
                       2444
010.124  176          2445          MOV     A,M             *12
010.125  022          2446          STAX    D
010.126  043          2447          INX     H
010.127  023          2448          INX     D
                       2449
010.130  176          2450          MOV     A,M             *13
010.131  022          2451          STAX    D
010.132  043          2452          INX     H
010.133  023          2453          INX     D
                       2454
010.134  176          2455          MOV     A,M             *14
010.135  022          2456          STAX    D
010.136  043          2457          INX     H
010.137  023          2458          INX     D
                       2459
010.140  176          2460          MOV     A,M             *15
010.141  022          2461          STAX    D
010.142  043          2462          INX     H
010.143  023          2463          INX     D
                       2464
010.144  176          2465          MOV     A,M             *16
010.145  022          2466          STAX    D
010.146  043          2467          INX     H
010.147  023          2468          INX     D
                       2469
010.150  020 266      2470          DJNZ    CPM16
010.152  311          2471          RET             TIL DONE
```

```
                        2473  **      D25L - DISABLE 25TH LINE
                        2474  *
                        2475  *       *D25L* DISABLES THE DISPLAY OF THE 25TH LINE
                        2476  *
                        2477  *
                        2478  *       ENTRY   NONE
                        2479  *
                        2480  *       EXIT    NONE
                        2481  *
                        2482  *       USES    A,F
                        2483
                        2484
010.153  072 311 100   2485  D25L    LDA     MODEI           GET MODE FLAGS
010.156  346 177       2486          ANI     255-MI.25L      CLEAR 25TH LINE FLAG
010.160  062 311 100   2487          STA     MODEI
010.163  076 030       2488          MVI     A,24
010.165  062 276 100   2489          STA     VI.VD           UPDATE VIDEO DISPLAYED INFO FOR NMI
010.170  311           2490          RET


                        2492  **      DALF - DISABLE AUTO LINE FEED
                        2493  *
                        2494  *       *DALF* DISABLES A CARRIAGE RETURN CAUSING AN AUTOMATIC LINE FEED
                        2495  *
                        2496  *
                        2497  *       ENTRY   (B,C) = MODEB
                        2498  *
                        2499  *       EXIT    NONE
                        2500  *
                        2501  *       USES    A,F
                        2502
                        2503
010.171  012           2504  DALF    LDAX    B               GET MODE FLAGS
010.172  346 367       2505          ANI     255-MB.ALF      CLEAR AUTO LINE FEED
010.174  002           2506          STAX    B
010.175  311           2507          RET


                        2509  **      DC - DISABLE CURSOR
                        2510  *
                        2511  *       *DC* INHIBITS THE DISPLAY OF THE CURSOR.  ALL OTHER FUNCTIONS
                        2512  *       PERFORM AS THEY WOULD IF THE CURSOR WERE ON
                        2513  *
                        2514  *
                        2515  *       ENTRY   NONE
                        2516  *
                        2517  *       EXIT    NONE
                        2518  *
                        2519  *       USES    A
                        2520
                        2521
010.176  072 307 100   2522  DC      LDA     MODEA           GET MODE FLAGS
```

```
010.201  366 020      2523          ORI     MA.CD           SET CURSOR DISABLED
010.203  062 307 100  2524          STA     MODEA
010.206  046 040      2525          MVI     H,VB.CND        CURSOR NOT DISPLAYED
010.210  056 000      2526          MVI     L,0             SET CURSOR END ADDRESS TO ZERO
010.212  042 277 100  2527          SHLD    VI.CSE
010.215  311          2528          RET


                      2530  **      DEOL - DISCARD AT END OF LINE
                      2531  *
                      2532  *       *DEOL* RESETS THE WRAP AROUND AT END OF LINE FLAG.  CHARACTERS
                      2533  *       INCOMING CHARACTERS PAST COLUMN 79 ARE PLACED IN COLUMN 79 UNTIL
                      2534  *       A CARRIAGE RETURN IS RECEIVED
                      2535  *
                      2536  *
                      2537  *       ENTRY   (B,C) = MODEB
                      2538  *
                      2539  *       EXIT    NONE
                      2540  *
                      2541  *       USES    A,F
                      2542
                      2543
010.216  012          2544  DEOL    LDAX    B               GET MODE FLAGS
010.217  346 373      2545          ANI     255-MB.WRAP     CLEAR WRAP AROUND FLAG
010.221  002          2546          STAX    B
010.222  311          2547          RET


                      2549  **      DING - DING BELL
                      2550  *
                      2551  *
                      2552  *       ENTRY   NONE
                      2553  *
                      2554  *       EXIT    NONE
                      2555  *
                      2556  *       USES    NONE
                      2557
                      2558
010.223               2559  DING    EQU     *
010.223  323 340      2560          OUT     MP.BELL
010.225  311          2561          RET


                      2563  **      DKI - DISABLE KEYBOARD INPUT
                      2564  *
                      2565  *       *DKI* CAUSES ALL KEYBOARD ENTRYS TO BE IGNORED
                      2566  *
                      2567  *
                      2568  *       ENTRY   NONE
                      2569  *
```

```
                        2570  *       EXIT    NONE
                        2571  *
                        2572  *       USES    A,F
                        2573
                        2574
010.226  072 311 100   2575  DKI     LDA     MODEI           GET PROPER MODE FLAGS
010.231  366 004       2576          ORI     MI.KID          SET KEYBOARD INPUT DISABLED
010.233  062 311 100   2577          STA     MODEI
010.236  311           2578          RET


                        2580  **      E25L - ENABLE 25TH LINE
                        2581  *
                        2582  *       *E25L* ENABLES THE DISPLAY OF THE 25TH LINE ONLY IF IT HAS NOT
                        2583  *       BEEN PREVIOUSLY ENABLED.  THE 25TH LINE IS CLEARED AT THE TIME
                        2584  *       IT IS ENABLED.
                        2585  *
                        2586  *
                        2587  *       ENTRY   NONE
                        2588  *
                        2589  *       EXIT    NONE
                        2590  *
                        2591  *       USES    A,B,C,D,E,H,L,F
                        2592
                        2593
010.237  072 311 100   2594  E25L    LDA     MODEI           GET MODE FLAGS
010.242  107           2595          MOV     B,A             SAVE
010.243  346 200       2596          ANI     MI.25L          SEE IF 25TH LINE ALREADY ON
010.245  300           2597          RNZ                     IF ON
                        2598
010.246  076 200       2599          MVI     A,MI.25L        ELSE, SET FLAG
010.250  260           2600          ORA     B
010.251  062 311 100   2601          STA     MODEI
010.254  052 266 100   2602          LHLD    SHOME           GET CURRENT HOME POSITION
010.257  021 200 007   2603          LXI     D,1920          ADD 24 LINE SIZE FOR BEGINNING OF 25TH
010.262  031           2604          DAD     D               (H,L) = STARTING ADDRESS OF LINE 25
010.263  006 005       2605          MVI     B,80/16         SET MODULO 16 COUNT FOR ERASE
010.265  315 212 016   2606          CALL    WSVA.           ERASE LINE
                        2607
010.270  076 031       2608          MVI     A,25            SET VERTICAL DISPLAYED INFO TO 25 LINES
010.272  062 276 100   2609          STA     VI.VD
010.275  311           2610          RET


                        2612  **      EACR - ENTER AUTO CARRIAGE RETURN
                        2613  *
                        2614  *       *EACR* ENABLES THE TERMINAL TO PERFORM AN AUTOMATIC CARRIAGE
                        2615  *       RETURN UPON RECEIPT OF A LINE FEED CHARACTER
                        2616  *
                        2617  *
                        2618  *       ENTRY   (B,C) = MODEB
                        2619  *
```

```
                        2620  *       EXIT    NONE
                        2621  *
                        2622  *       USES    A,F
                        2623
                        2624
010.276  012            2625  EACR    LDAX    B               GET MODE FLAGS
010.277  366 020        2626          ORI     MB.ACR          SET AUTO CARRIAGE RETURN
010.301  002            2627          STAX    B
010.302  311            2628          RET


                        2630  **      EALF - ENABLE AUTO LINE FEED
                        2631  *
                        2632  *       *EALF* ENABLES THE TERMINAL TO PERFORM AN AUTOMATIC LINE FEED
                        2633  *       UPON RECEIPT OF A CARRIAGE RETURN
                        2634  *
                        2635  *
                        2636  *       ENTRY   (B,C) = MODEB
                        2637  *
                        2638  *       EXIT    NONE
                        2639  *
                        2640  *       USES    A,F
                        2641
                        2642
010.303  012            2643  EALF    LDAX    B               GET MODE FLAGS
010.304  366 010        2644          ORI     MB.ALF
010.306  002            2645          STAX    B
010.307  311            2646          RET


                        2648  **      EAM - ENTER ANSI MODE
                        2649  *
                        2650  *       *EAM* PLACES THE TERMINAL IN THE ANSI MODE FOR ESCAPE CODES.
                        2651  *
                        2652  *
                        2653  *       ENTRY   (B,C) = MODEB
                        2654  *
                        2655  *       EXIT    NONE
                        2656  *
                        2657  *       USES    A,F
                        2658
                        2659
010.310  012            2660  EAM     LDAX    B               GET MODE FLAGS
010.311  366 040        2661          ORI     MB.ANSI         SET ANSI MODE FLAG
010.313  002            2662          STAX    B
010.314  311            2663          RET
```

```
                                2665  **      EBD - ERASE BEGINNING OF DISPALY
                                2666  *
                                2667  *       *EBD* ERASES THE SCREEN FROM 'HOME' TO THE CURRENT CURSOR POSITION
                                2668  *
                                2669  *
                                2670  *       ENTRY   NONE
                                2671  *
                                2672  *       EXIT    NONE
                                2673  *
                                2674  *       USES    A,B,C,D,E,H,L,F
                                2675
                                2676
010.315                         2677  EBD     EQU     *
010.315  072 273 100           2678          LDA     CURVP           GET VERTICAL POSITION
010.320  376 030               2679          CPI     24              SEE IF ON 25TH LINE
010.322  050 026               2680          JR      Z,EBL           IF SO, JUST DO BEGINNING OF THIS LINE
                                2681
                                2682  *       ERASE ALL LINES ABOVE CURRENT LINE
                                2683  *
010.324  072 273 100           2684          LDA     CURVP           GET CURRENT LINE NUMBER
010.327  267                   2685          ORA     A               SEE IF ALREADY ON HOME LINE
010.330  050 020               2686          JR      Z,EBD1          IF SO, NO FULL LINES TO ERASE
                                2687
010.332  157                   2688          MOV     L,A             MULTIPLY LINE COUNT BY 5 FOR MODULO 16 COLUMNS
010.333  137                   2689          MOV     E,A
010.334  046 000               2690          MVI     H,0
010.336  127                   2691          MOV     D,A
010.337  031                   2692          DAD     D               *2
010.340  031                   2693          DAD     D               *3
010.341  031                   2694          DAD     D               *4
010.342  031                   2695          DAD     D               *5
010.343  105                   2696          MOV     B,L             (B) = RESULT
010.344  052 266 100           2697          LHLD    SHOME           (H,L) = FIRST ADDRESS TO SPACE
010.347  315 212 016           2698          CALL    WSVA            WRITE SPACES
                                2699
010.352                         2700  EBD1    EQU     *
                                2701  *       JMP     EBL             ERASE BEGINNING OF CURRENT LINE
000.000                         2702          ERRNZ   *-EBL


                                2704  **      EBL - ERASE BEGINNING OF LINE
                                2705  *
                                2706  *       *EBL* ERASES FROM THE BEGINNING OF THE CURRENT LINE TO THE CURSOR
                                2707  *
                                2708  *
                                2709  *       ENTRY   NONE
                                2710  *
                                2711  *       EXIT    NONE
                                2712  *
                                2713  *       USES    A,B,H,L,F
                                2714
                                2715
010.352  052 270 100           2716  EBL     LHLD    CLSA            GET ADDRESS OF FIRST COLUMN ON THIS LINE
010.355  072 272 100           2717          LDA     CURHP           GET COLUMN COUNT
```

```
010.360  074            2718          INR     A               ADD ONE FOR THE CURSOR POSITION
010.361  107            2719          MOV     B,A             (B) = COUNT
                        2720
                        2721
010.362  076 040       2722          MVI     A,' '           WRITE SPACES
010.364  167            2723  EBL1    MOV     M,A             PUT SPACE IN DISPLAY
010.365  043            2724          INX     H               POINT TO NEXT
010.366  020 374        2725          DJNZ    EBL1            UNTIL DONE
                        2726
010.370  311            2727          RET
                        2728


                        2730  **      EC - ENABLE CURSOR
                        2731  *
                        2732  *       *EC* ENABLES THE DISPLAY OF THE CURSOR
                        2733  *
                        2734  *
                        2735  *       ENTRY   NONE
                        2736  *
                        2737  *       EXIT    NONE
                        2738  *
                        2739  *       USES    A,F
                        2740
                        2741
010.371  072 307 100    2742  EC      LDA     MODEA           GET MODE FLAGS
010.374  346 357        2743          ANI     255-MA.CD       SHOW CURSOR AS NOT DISABLED
010.376  062 307 100    2744          STA     MODEA
011.001  072 310 100    2745          LDA     MODEB           GET OTHER MODE FLAGS
011.004  346 001        2746          ANI     MB.CBLK         BLOCK CURSOR SELECTED?
011.006  302 261 015    2747          JNZ     SBC.            IF BLOCK SELECTED, SET IT
                        2748
011.011  303 026 016    2749          JMP     SUC.            ELSE, SET UNDERSCORE CURSOR


                        2751  **      EDD - ENCODE DECIMAL DIGITS
                        2752  *
                        2753  *       *EDD* ENCODES TWO DECIMAL DIGITS FROM THE BINARY VALUE SUPPLIED.
                        2754  *       THE DECIMAL DIGITS ARE IN THE FORM OF TWO BCD BYTES (NOT ASCII).
                        2755  *
                        2756  *
                        2757  *       ENTRY   (A) = BINARY VALUE TO BE ENCODED
                        2758  *
                        2759  *       EXIT    (D) = DECIMAL TENS DIGIT (IN BCD)
                        2760  *               (E) = DECIMAL ONES DIGIT (IN BCD)
                        2761  *
                        2762  *       USES    A,D,E
                        2763
                        2764
011.014  026 000        2765  EDD     MVI     D,0             CLEAR DECIMAL COUNTERS
011.016  376 012        2766  EDD1    CPI     10              SEE IF BINARY IS MORE THAN NINE
011.020  070 005        2767          JR      C,EDD2          IF LESS THAN TEN
```

```
                                2768
011.022  326 012                2769            SUI     10              ELSE, SUBTRACT TEN FROM BINARY
011.024  024                    2770            INR     D               ADD ONE TO TENS DIGIT
011.025  030 367                2771            JR      EDD1            SEE IF MORE TENS
                                2772
011.027  137                    2773  EDD2      MOV     E,A             LEFTOVERS ARE ONES DIGIT
011.030  311                    2774            RET


                                2776  **        EIL - ERASE IN LINE
                                2777  *
                                2778  *         *EIL* ERASES THE PORTION OF THE CURRENT LINE AS SPECIFIED BY PN
                                2779  *
                                2780  *
                                2781  *         ENTRY   (B) = ZERO IF NO PN WAS INPUT
                                2782  *                 (D,E) = PSDW
                                2783  *
                                2784  *         EXIT    NONE
                                2785  *
                                2786  *         USES    A,B,C,D,E,H,L,F
                                2787
                                2788
011.031  170                    2789  EIL       MOV     A,B             SEE IF PN WAS INPUT
011.032  267                    2790            ORA     A
011.033  312 277 011            2791            JZ      EOL             IF NO PN, ERASE TO END OF LINE
                                2792
011.036  353                    2793            XCHG                    (H,L) = PSDW
011.037  176                    2794            MOV     A,M             GET PN
011.040  267                    2795            ORA     A               ZERO?
011.041  312 277 011            2796            JZ      EOL             IF ZERO, ERASE TO END OF LINE
                                2797
011.044  075                    2798            DCR     A               ONE?
011.045  050 303                2799            JR      Z,EBL           IF ONE, ERASE BEGINNING OF LINE
                                2800
011.047  075                    2801            DCR     A               TWO?
011.050  300                    2802            RNZ                     IF NOT TWO, ILLEGAL, EXIT
                                2803
                                2804  *         JMP     EEL             ELSE, ERASE ENTIRE LINE
000.000                         2805            ERRNZ   *-EEL


                                2807  **        EEL - ERASE ENTIRE LINE
                                2808  *
                                2809  *         *EEL* WRITES SPACES INTO THE ENTIRE LINE WHERE THE CURSOR RESIDES
                                2810  *
                                2811  *
                                2812  *         ENTRY   NONE
                                2813  *
                                2814  *         EXIT    NONE
                                2815  *
                                2816  *         USES    A,B,C,H,L,F
                                2817
```

```
                                2818
011.051  052 270 100  2819  EEL     LHLD    CLSA            START ERASING AT BEGINNING OF CURRENT LINE
011.054  006 005      2820          MVI     B,80/16         MODULO 16 COUNT FOR NUMBER TO WRITE
011.056  303 212 016  2821          JMP     WSVA            WRITE SPACES AND EXIT


                                2823  **      EGM - ENTER GRAPHICS MODE
                                2824  *
                                2825  *       *EGM* SETS THE GRAPHICS MODE FLAG
                                2826  *
                                2827  *
                                2828  *       ENTRY   (D,E) = MODEA
                                2829  *
                                2830  *       EXIT    NONE
                                2831  *
                                2832  *       USES    A,F
                                2833
                                2834
011.061  353                    2835  EGM     XCHG                            (H,L) = MODE
                                2836  *       SET     IB.GRPH,(HL)    SET GRAPHICS MODE FLAG
011.062  313 316                2837          DB      I.SETHA,I.SETHB!IB.GRPH
011.064  311                    2838          RET


                                2840  **      EHM - ENTER HEATH MODE
                                2841  *
                                2842  *       *EHM* TAKES THE TERMINAL OUT OF THE ANSI MODE AND INTO THE HEATH MODE
                                2843  *
                                2844  *
                                2845  *       ENTRY   NONE
                                2846  *
                                2847  *       EXIT    NONE
                                2848  *
                                2849  *       USES    A,F
                                2850
                                2851
011.065  072 310 100  2852  EHM     LDA     MODEB           GET MODE FLAGS
011.070  346 337      2853          ANI     255-MB.ANSI
011.072  062 310 100  2854          STA     MODEB
011.075  311          2855          RET


                                2857  **      EHSM - ENTER HOLD SCREEN MODE
                                2858  *
                                2859  *
                                2860  *       *EHSM* SETS THE HOLD SCREEN MODE FLAG
                                2861  *
                                2862  *
                                2863  *       ENTRY   (D,E) = MODEA
                                2864  *
```

```
                        2865  *       EXIT    NONE
                        2866  *
                        2867  *       USES    A,F
                        2868
                        2869
011.076  353           2870  EHSM    XCHG                    (H,L) = MODEA
                        2871  *       SET     IB.HSM,(H,L)    SET HOLD SCREEN MODE FLAG
011.077  313 306       2872          DB      I.SETHA,I.SETHB!IB.HSM
011.101  076 001       2873          MVI     A,1             SET LINE COUNTER TO ONE
011.103  062 313 100   2874          STA     HSMLC
011.106  311           2875          RET


                        2877  **      EICM - ENTER INSERT CHARACTER MODE
                        2878  *
                        2879  *       *EICM* SETS THE INSERT CHARACTER MODE FLAG
                        2880  *
                        2881  *
                        2882  *       ENTRY   (D,E) = MODEA
                        2883  *
                        2884  *       EXIT    NONE
                        2885  *
                        2886  *       USES    A,F
                        2887
                        2888
011.107  353           2889  EICM    XCHG                    (H,L) = MODEA
                        2890  *       SET     IB.ICM,(HL)     SET INSERT CHARACTER MODE FLAG
011.110  313 366       2891          DB      I.SETHA,I.SETHB!IB.ICM
011.112  311           2892          RET


                        2894  **      EKAM - ENTER KEYPAD ALTERNATE MODE
                        2895  *
                        2896  *       *EKAM* SETS THE KEYPAD ALTERNATE MODE FLAG IN *MODE*
                        2897  *
                        2898  *
                        2899  *       ENTRY   (B,C) = MODEB
                        2900  *
                        2901  *       EXIT    NONE
                        2902  *
                        2903  *       USES    A,F
                        2904
                        2905
011.113  012           2906  EKAM    LDAX    B               GET MODEB FLAGS
011.114  366 200       2907          ORI     MB.KPDA         SET KEYPAD ALTERNATE MODE
011.116  002           2908          STAX    B
011.117  311           2909          RET
```

```
                        2911  **      EKC - ENABLE KEYBOARD CLICK
                        2912  *
                        2913  *       *EKC* RESETS THE FLAG WHICH INHIBITS THE KEY CLICK
                        2914  *
                        2915  *
                        2916  *       ENTRY   (B,C) = MODEB
                        2917  *
                        2918  *       EXIT    NONE
                        2919  *
                        2920  *       USES    A,F
                        2921
                        2922
011.120  012           2923  EKC     LDAX    B               GET MODE FLAGS
011.121  346 375       2924          ANI     255-MB.NOTK     CLEAR NO TICK FLAG
011.123  002           2925          STAX    B
011.124  311           2926          RET


                        2928  **      EKI - ENABLE KEYBOARD INPUT
                        2929  *
                        2930  *       *EKI* RESETS THE FLAG WHICH DISABLES INPUT FROM THE KEYBOARD
                        2931  *
                        2932  *
                        2933  *       ENTRY   NONE
                        2934  *
                        2935  *       EXIT    NONE
                        2936  *
                        2937  *       USES    A,F
                        2938
                        2939
011.125  072 311 100    2940  EKI     LDA     MODEI           GET MODE FLAGS
011.130  346 373       2941          ANI     255-MI.KID      TOSS KEYBOARD DISABLE FLAG
011.132  062 311 100    2942          STA     MODEI
011.135  311           2943          RET


                        2945  **      EKSM - ENTER KEYPAD SHIFTED MODE
                        2946  *
                        2947  *       *EKSM* SETS THE KEYPAD SHIFTED MODE FLAG
                        2948  *
                        2949  *
                        2950  *       ENTRY   (B,C) = MODEB
                        2951  *
                        2952  *       EXIT    NONE
                        2953  *
                        2954  *       USES    A,F
                        2955
                        2956
011.136  012           2957  EKSM    LDAX    B               GET MODEB FLAGS
011.137  366 100       2958          ORI     MB.KPDS
011.141  002           2959          STAX    B
011.142  311           2960          RET
```

```
                                2962  **      ELB - ESCAPE LEFT BRACKET
                                2963  *
                                2964  *       *ELB* IS A CONTINUATION OF THE ESCAPE SEQUENCE PROCESSING FOR
                                2965  *       ANSI ESCPAE SEQUENCES.  THE FINAL CHARACTER OF THE SEQUENCE IS
                                2966  *       DECODED WITH OR WITHOUT A PRECEEDING PARAMETER STRING AND
                                2967  *       CONTROL IS PASSED ON TO THE ASSOCIATED ROUTINE
                                2968  *
                                2969  *
                                2970  *       ENTRY   NONE
                                2971  *
                                2972  *       EXIT    TO REQUESTED ROUTINE WITH
                                2973  *                       (B) = ZERO IF NO PARAMETER STRING WAS INPUT
                                2974  *
                                2975  *       USES    A,B,C,D,E,H,L,F
                                2976
                                2977
011.143  315 346 014           2978  ELB     CALL    PSD             DECODE PARAMETER STRING
011.146  305                   2979          PUSH    B               SAVE (B)
                                2980
                                2981  *       SEARCH TABLE FOR FINAL CHARACTER OF SEQUENCE
                                2982  *
011.147  026 027               2983          MVI     D,ELBTL         SET TABLE LENGTH
011.151  036 003               2984          MVI     E,ELBTW         SET TABLE WIDTH
011.153  041 172 011           2985          LXI     H,ELBT          SET TABLE ADDRESS
011.156  315 363 015           2986          CALL    STAB            SEARCH TABLE
011.161  301                   2987          POP     B               RESTORE (B)
011.162  330                   2988          RC                      IF CHARACTER NOT FOUND IN TABLE, EXIT NOW
                                2989
011.163  043                   2990          INX     H               ELSE, GET ADDRESS OF ROUTINE
011.164  146                   2991          MOV     H,M             GET MSB
011.165  157                   2992          MOV     L,A             GET LSB
011.166  021 314 100           2993          LXI     D,PSDW          (D,E) = PSD WORK AREA
011.171  351                   2994          PCHL                    GO TO ROUTINE



                                2996  *       ELBT - ESCAPE LEFT BRACKET TABLE
                                2997  *
                                2998  *       *ELBT* CONTAINS THE THIRD AND/OR FINAL CHARACTERS OF THE ANSI
                                2999  *       ESCAPE SEQUENCES
                                3000
                                3001
011.172                         3002  ELBT    EQU     *
                                3003
011.172  076                   3004          DB      '>'             ESC [ >
011.173  073 005               3005          DW      A2M             ANSI SET MODE #2
                                3006
011.175  077                   3007          DB      '?'             ESC [ ?
011.176  376 004               3008          DW      A1M             ANSI SET MODE #1
                                3009
011.200  101                   3010          DB      'A'             ESC [ A
011.201  346 005               3011          DW      ACUP            ANSI CURSOR UP
                                3012
011.203  102                   3013          DB      'B'             ESC [ B
011.204  156 005               3014          DW      ACDN            ANSI CURSOR DOWN
                                3015
```

```
011.206  103          3016          DB      'C'             ESC [ C
011.207  324 005      3017          DW      ACRT            ANSI CURSOR RIGHT
                      3018
011.211  104          3019          DB      'D'             ESC [ D
011.212  201 005      3020          DW      ACLFT           ANSI CURSOR LEFT
                      3021
011.214  110          3022          DB      'H'             ESC [ H
011.215  371 005      3023          DW      APCA            ANSI PERFORM CURSOR ADDRESSING
                      3024
011.217  112          3025          DB      'J'             ESC [ J
011.220  377 007      3026          DW      EID             ERASE IN DISPLAY
                      3027
011.222  113          3028          DB      'K'             ESC [ K
011.223  031 011      3029          DW      EIL             ERASE IN LINE
                      3030
011.225  114          3031          DB      'L'             ESC [ L
011.226  116 006      3032          DW      APIL            ANSI PERFORM INSERT LINE
                      3033
011.230  115          3034          DB      'M'             ESC [ M
011.231  073 006      3035          DW      APDL            ANSI PERFORM DELETE LINE
                      3036
011.233  120          3037          DB      'P'             ESC [ P
011.234  050 006      3038          DW      APDC            ANSI PERFORM DELETE CHARACTER
                      3039
                      3040
011.236  146          3041          DB      'f'             ESC [ f         (LOWER CASE F)
011.237  371 005      3042          DW      APCA            ANSI PERFORM CURSOR ADDRESSING
                      3043
011.241  150          3044          DB      'h'             ESC [ h         (LOWER CASE H)
011.242  301 006      3045          DW      ASM             ANSI SET MODE
                      3046
011.244  154          3047          DB      'l'             ESC [ l         (LOWER CASE L)
011.245  141 006      3048          DW      ARM             ANSI RESET MODE
                      3049
011.247  155          3050          DB      'm'             ESC [ m         (LOWER CASE M)
011.250  222 006      3051          DW      ASGM            ANSI SET GRAPHICS MODE
                      3052
011.252  156          3053          DB      'n'             ESC [ n         (LOWER CASE N)
011.253  223 005      3054          DW      ACPR            ANSI CURSOR POSITION REPORT
                      3055
011.255  160          3056          DB      'p'             ESC [ p         (LOWER CASE P)
011.256  247 017      3057          DW      AXMTP           ANSI TRANSMIT PAGE
                      3058
011.260  161          3059          DB      'q'             ESC [ q         (LOWER CASE Q)
011.261  323 016      3060          DW      AXMT25          ANSI TRANSMIT 25TH LINE
                      3061
011.263  162          3062          DB      'r'             ESC [ r         (LOWER CASE R)
011.264  206 006      3063          DW      ASBR            ANSI SET BAUD RATE
                      3064
011.266  163          3065          DB      's'             ESC [ s         (LOWER CASE S)
011.267  127 015      3066          DW      ASCP            ANSI SAVE CURSOR POSITION
                      3067
011.271  165          3068          DB      'u'             ESC [ u         (LOWER CASE U)
011.272  311 015      3069          DW      AUSCP           ANSI UNSAVE CURSOR POSITION
                      3070
011.274  172          3071          DB      'z'             ESC [ z         (LOWER CASE Z)
```

```
011.275  375 012        3072           DW      ARAMP           ANSI RESET ALL MODES TO POWER UP CONFIGURATION
                        3073
000.003                 3074  ELBTW    EQU     3               TABLE WIDTH IS 3
000.027                 3075  ELBTL    EQU     *-ELBT/ELBTW    TABLE LENGTH



                        3077  **       EOL - ERASE TO END OF LINE
                        3078  *
                        3079  *        *EOL* PLACES SPACES IN VIDEO RAM FROM THE CURRENT CURSOR POSITION
                        3080  *        TO THE END OF THE CURRENT LINE.  CURSOR POSITION DOES NOT CHANGE.
                        3081  *
                        3082  *
                        3083  *        ENTRY   NONE
                        3084  *
                        3085  *        EXIT    NONE
                        3086  *
                        3087  *        USES    A,B,C,D,E,H,L,F
                        3088
                        3089
011.277  052 274 100    3090  EOL      LHLD    CURAD           GET CURSOR ADDRESS
011.302  072 272 100    3091           LDA     CURHP           GET CURRENT COLUMN POSITION
                        3092  *        NEG                     SUBTRACT COLUMN COUNTER FROM 80
011.305  355 104        3093           DB      I.NEGA,I.NEGB
011.307  306 120        3094           ADI     80
011.311  137            3095           MOV     E,A             PLACE COUNT IN D & E
011.312  026 000        3096           MVI     D,0
011.314  303 160 016    3097           JMP     WSV             WRITE SPACES ON REST OF LINE



                        3100  **       ERM - ERASE REST OF MEMORY
                        3101  *
                        3102  *        *ERM* ERASES THE SCREEN FROM THE CURRENT CURSOR POSITION TO
                        3103  *        THE END OF THE SCREEN
                        3104  *
                        3105  *
                        3106  *        ENTRY   NONE
                        3107  *
                        3108  *        EXIT    NONE
                        3109  *
                        3110  *        USES    A,B,C,D,E,H,L,F
                        3111
                        3112
011.317  072 273 100    3113  ERM      LDA     CURVP           GET CURRENT VERTICAL POSITION
011.322  376 030        3114           CPI     24              SEE IF ON 25TH LINE
011.324  050 351        3115           JR      Z,EOL           IF SO, JUST SETTLE FOR END OF THIS LINE
                        3116
011.326  052 266 100    3117           LHLD    SHOME           GET HOME ADDRESS
011.331  021 177 007    3118           LXI     D,1919          ADD DISPLAY SIZE TO FIND END
```

```
011.334  031              3119           DAD     D
011.335  067              3120           STC                     CLEAR CARRY BIT FOR NEXT OPERATION
011.336  077              3121           CMC
                          3122  *        LD      DE,(CURAD)
011.337  355 133          3123           DB      I.LDEDA,I.LDEDB
011.341  274 100          3124           DW      CURAD
                          3125  *        SBC     HL,DE           SUBTRACT CURSOR ADDRESS FROM END OF DISPLAY
011.343  355 122          3126           DB      I.SHDA,I.SHDB
011.345  043              3127           INX     H               ADD ONE TO ERASE CHARACTER AT (CURAD)
011.346  353              3128           XCHG                    (D,E) = COUNT TO END OF DISPLAY
011.347  172              3129           MOV     A,D             MASK TO KEEP COUNT UNDER 2K
011.350  346 007          3130           ANI     00000111B
011.352  127              3131           MOV     D,A
011.353  052 274 100      3132           LHLD    CURAD           (H,L) = ADDRESS OF FIRST SPACE TO WRITE
011.356  303 160 016      3133           JMP     WSV             WRITE SPACES


                          3135  **       ERVM - ENTER REVERSE VIDEO MODE
                          3136  *
                          3137  *        *ERVM* SETS THE REVERSE VIDEO MODE FLAG
                          3138  *
                          3139  *
                          3140  *        ENTRY   (D,E) = MODEA
                          3141  *
                          3142  *        EXIT    NONE
                          3143  *
                          3144  *        USES    A,F
                          3145
                          3146
011.361                   3147  ERVM     EQU     *
                          3148
                          3149
011.361  353              3150           XCHG                    (H,L) = MODE
011.362  176              3151           MOV     A,M             GET MODE FLAGS
011.363  366 210          3152           ORI     MA.RV+MA.RVP    SET REVERSE VIDEO AND RV PRESENT FLAGS
011.365  167              3153           MOV     M,A
011.366  311              3154           RET


                          3156  **       FCIF - FETCH CHARACTER FROM INPUT FIFO
                          3157  *
                          3158  *        *FCIF* IS USED TO FETCH A SINGLE CHARACTER FROM THE INPUT FIFO
                          3159  *        IF ANY ARE AVAILABLE.
                          3160  *
                          3161  *
                          3162  *        ENTRY   NONE
                          3163  *
                          3164  *        EXIT    (A) = CHARACTER IF ONE IS AVAILABLE
                          3165  *                'C' = SET IF NO CHARACTER
                          3166  *                'C' = CLEAR IF CHARACTER IN ACCUMULATOR
                          3167  *
                          3168  *        USES    A,B,H,L,F
```

```
                         3169
                         3170
011.367  072 241 100    3171  FCIF    LDA     IFC             GET INPUT FIFO COUNTER
011.372  267            3172          ORA     A               SEE IF FIFO IS EMPTY
011.373  067            3173          STC                     SET 'C' FOR NO CHARACTER
011.374  310            3174          RZ                      IF NO CHARACTER IN FIFO, EXIT
                         3175
011.375  363            3176          DI                      LOCK OUT ANY ENTRYS FOR NOW
011.376  072 241 100    3177          LDA     IFC             DECREMENT INPUT FIFO COUNTER
012.001  075            3178          DCR     A
012.002  062 241 100    3179          STA     IFC
012.005  072 240 100    3180          LDA     IFP             GET INPUT FIFO POINTER
012.010  107            3181          MOV     B,A             SAVE VALUE
012.011  041 000 100    3182          LXI     H,INF           POINT TO INPUT FIFO
012.014  205            3183          ADD     L               ADD POINTER
012.015  157            3184          MOV     L,A
012.016  170            3185          MOV     A,B             INCREMENT POINTER
012.017  074            3186          INR     A
012.020  346 177        3187          ANI     IFCMSK          KEEP POINTER IN FIFO
012.022  062 240 100    3188          STA     IFP             UPDATE POINTER
012.025  176            3189          MOV     A,M             READ CHARACTER FROM FIFO
012.026  067            3190          STC                     CLEAR 'C'
012.027  077            3191          CMC
012.030  373            3192          EI                      ALLOW NEW ENTRYS TO FIFO
012.031  311            3193          RET


                         3195  **      FCOF - FETCH CHARACTER FROM OUTPUT FIFO
                         3196  *
                         3197  *       *FCOF* FETCHES A CHARACTER FROM THE OUTPUT FIFO IF ANY ARE AVAILABLE
                         3198  *
                         3199  *
                         3200  *       ENTRY   NONE
                         3201  *
                         3202  *       EXIT    (A) = CHARACTER IF AVAILABLE
                         3203  *               'C' = SET IF NO CHARACTERS IN FIFO
                         3204  *               'C' = CLEAR IF CHARACTER IN ACCUMULATOR
                         3205  *
                         3206  *       USES    A,B,H,L,F
                         3207
                         3208
012.032  072 243 100    3209  FCOF    LDA     OFC             GET OUTPUT FIFO COUNTER
012.035  267            3210          ORA     A               SEE IF FIFO IS EMPTY
012.036  067            3211          STC                     SET 'C' FOR NO CHARACTERS
012.037  310            3212          RZ                      IF NO CHARACTERS IN FIFO
                         3213
012.040  075            3214          DCR     A               DECREMENT COUNTER
012.041  062 243 100    3215          STA     OFC
012.044  072 242 100    3216          LDA     OFP             GET OUTPUT FIFO POINTER
012.047  107            3217          MOV     B,A             SAVE FOR LATER
012.050  041 200 100    3218          LXI     H,OUTF          POINT TO OUTPUT FIFO
012.053  205            3219          ADD     L               ADD POINTER
012.054  157            3220          MOV     L,A
012.055  170            3221          MOV     A,B             INCREMENT POINTER
```

```
012.056  074            3222          INR     A
012.057  346 037        3223          ANI     OFCMSK          KEEP POINTER IN FIFO
012.061  062 242 100    3224          STA     OFP             UPDATE OUTPUT FIFO POINTER
012.064  176            3225          MOV     A,M             READ CHARACTER FROM FIFO
012.065  067            3226          STC                     CLEAR 'C'
012.066  077            3227          CMC
012.067  311            3228          RET


                        3230  **      FNCP - FETCH NEXT CHARACTER FOR PRIVATE PROCESSING
                        3231  *
                        3232  *       FNCP GETS THE NEXT CHARACTER AVAILABLE FROM THE INPUT FIFO.
                        3233  *       IF NONE ARE AVAILABLE, FNCP CONTINUES TO PROCESS KEYBOARD
                        3234  *       CHARACTERS AND OUTPUT FIFO CHARACTERS
                        3235  *
                        3236  *
                        3237  *       ENTRY   NONE
                        3238  *
                        3239  *       EXIT    (A) = CHARACTER FROM INPUT FIFO
                        3240  *               'C' = CLEARED
                        3241  *
                        3242  *       USES    A,F
                        3243
                        3244
012.070  305            3245  FNCP    PUSH    B               SAVE REGISTERS
012.071  325            3246          PUSH    D
012.072  345            3247          PUSH    H
012.073  315 016 000    3248  FNCP1   CALL    MAIN.N          SERVICE KEYBOARD AND OUTPUT FIFO
012.076  315 367 011    3249          CALL    FCIF            SEE IF THERE IS A CHARACTER IN THE INPUT FIFO
012.101  070 370        3250          JR      C,FNCP1         IF NO CHARACTER YET
                        3251
012.103  341            3252          POP     H               ELSE, RESTORE REGISTERS AND EXIT
012.104  321            3253          POP     D
012.105  301            3254          POP     B
012.106  311            3255          RET


                        3258  **      FVKF - FETCH VALUE FROM KEYBOARD FIFO
                        3259  *
                        3260  *       *FVKF* FETCHES A TWO BYTE VALUE FROM THE KEYBOARD FIFO
                        3261  *       AND THEN DOES A BUBBLE DOWN ON THE REST OF THE CONTENTS OF THE FIFO
                        3262  *       AND UPDATES THE KEYBOARD FIFO POINTER
                        3263  *
                        3264  *
                        3265  *       ENTRY   NONE
                        3266  *
                        3267  *       EXIT    (D,E) = TWO BYTE VALUE FROM KEYBOARD FIFO IF AVAILABLE
                        3268  *               'C' = SET IF FIFO WAS EMPTY
```

```
                                3269  *               'C' = CLEAR IF VALUE WAS PLACED IN (D,E)
                                3270  *
                                3271  *       USES    A,D,E,F
                                3272
                                3273
012.107  305                    3274  FVKF    PUSH    B               SAVE (B,C,H,L)
012.110  345                    3275          PUSH    H
012.111  363                    3276          DI                      INHIBIT ANY NEW ENTRYS WHILE REMOVING OLD
012.112  052 264 100            3277          LHLD    KBDFP           (H,L) = KEYBOARD FIFO POINTER
012.115  175                    3278          MOV     A,L             CHECK LSB TO SEE IF THERE ARE ANY ENTRYS
012.116  376 244                3279          CPI     KBDFMIN&377Q
012.120  067                    3280          STC                     SET CARRY IN CASE NO ENTRY
012.121  050 030                3281          JR      Z,FVKF1         IF FIFO EMPTY
                                3282
012.123  053                    3283          DCX     H               ELSE, UPDATE FIFO POINTER
012.124  053                    3284          DCX     H
012.125  042 264 100            3285          SHLD    KBDFP
012.130  041 244 100            3286          LXI     H,KBDF          (H,L) = BEGINNIG OF FIFO
012.133  126                    3287          MOV     D,M             (D) = VALUE FROM IP.KBD1
012.134  043                    3288          INX     H
012.135  136                    3289          MOV     E,M             (E) = VLAUE FROM IP.KBD2
012.136  043                    3290          INX     H               POINT TO NEXT ENTRY
012.137  325                    3291          PUSH    D               SAVE KEYBOARD VALUES
012.140  021 244 100            3292          LXI     D,KBDF          (D,E) = BEGINNING OF FIFO FOR BUBBLE DOWN
012.143  001 016 000            3293          LXI     B,KBDFL-2       (B,C) = NUMBER OF BYTES TO BUBBLE
                                3294  *       LDIR                    BUBBLE UNTIL (B,C) = ZERO
012.146  355 260                3295          DB      I.LDIRA,I.LDIRB
012.150  067                    3296          STC                     CLEAR CARRY BIT TO INDICAT VALUE FOUND
012.151  077                    3297          CMC
012.152  321                    3298          POP     D               (D,E) = KEYBOARD VALUES
012.153  341                    3299  FVKF1   POP     H               RESTORE REGISTERS
012.154  301                    3300          POP     B
012.155  373                    3301          EI
012.156  311                    3302          RET
```

```
                                3304  **      SBR - SET BAUD RATE
                                3305  *
                                3306  *       *SBR* ALLOWS THE BUAD RATE TO BE SET INDEPENDANT OF THE POWER-UP
                                3307  *       SWITCH CONFIGURATION
                                3308  *
                                3309  *
                                3310  *       ENTRY   NONE
                                3311  *
                                3312  *       EXIT    NONE
                                3313  *
                                3314  *       USES    A,B,C,D,E,H,L,F
                                3315
                                3316
012.157  315 070 012            3317  SBR     CALL    FNCP            FETCH NEXT CHARACTER
                                3318
                                3319  *       INPUT CHARACTER MUST BE AN ASCII A,B,C,D,E,F,G OR H.  (A=110, B=150,
                                3320  *       C=300, D=600, E=1200, F=1800, G=2000, H=2400, I=3600, J=4800, K=7200,
                                3321  *       L=9600)
```

```
                        3322  *
012.162  376 101       3323  SBR1    CPI     'A'             SEE IF CHARACTER IS IN RANGE
012.164  330           3324          RC                      IF LESS THAN AN 'A'
                        3325
012.165  376 116       3326          CPI     'N'
012.167  320           3327          RNC                     IF GREATER THAN A 'P'
                        3328
012.170  346 017       3329          ANI     F1.BR           MASK FOR LOWER BITS
                        3330
                        3331  *       ALTERNATE ENTRY POINT FROM *ASBR*
                        3332  *
012.172  107           3333  SBR.    MOV     B,A             SAVE RESULT
012.173  072 312 100   3334          LDA     MODES           GET SERIAL I/O MODE
012.176  346 360       3335          ANI     377Q-MS.BR      TOSS OLD BAUD RATE
012.200  260           3336          ORA     B               REPLACE WITH NEW BAUD RATE
012.201  062 312 100   3337          STA     MODES           UPDATE *IMAGE*
                        3338  *       JMP     IACE            SET ACE TO NEW RATE
000.000                 3339          ERRNZ   *-IACE



                        3341  **      IACE - INITIALIZE ACE (UART)
                        3342  *
                        3343  *       *IACE* SETS UP THE DEFAULT I/O PARAMETERS ACCORDING TO THE SWITCH
                        3344  *       POSITIONS ON PORT MP.PUP1
                        3345  *
                        3346  *
                        3347  *       ENTRY   NONE
                        3348  *
                        3349  *       EXIT    NONE
                        3350  *
                        3351  *       USES    A,B,C,D,E,H,L,F
                        3352
                        3353
012.204  076 200       3354  IACE    MVI     A,AB.DLAB       SET DIVISOR LATCH ACCESS BIT
012.206  323 103       3355          OUT     AP.LCR
                        3356
                        3357  *       SET BAUD RATE DIVISORS TO DESIRED BAUD RATE
                        3358  *
012.210  072 312 100   3359          LDA     MODES           GET SERIAL I/O MODE (POWER-UP SWITCH #1)
012.213  117           3360          MOV     C,A             SAVE IMAGE
012.214  346 017       3361          ANI     MS.BR           MASK FOR BAUD RATE SWITCHES
012.216  050 001       3362          JR      Z,IACE0.5       IF SWITCHES ARE SET TO ZERO KEEP 110 BAUD
                        3363
012.220  075           3364          DCR     A               ELSE, SWITCHES = SWITCHES-1
012.221  107           3365  IACE0.5 MOV     B,A
012.222  007           3366          RLC                     SWITCHES*2 = TABLE VECTOR
012.223  041 301 012   3367          LXI     H,BRTAB         POINT TO BAUD RATE DIVISOR TABLE
012.226  205           3368          ADD     L               ADD VECTOR
012.227  157           3369          MOV     L,A
012.230  176           3370          MOV     A,M             GET DESIRED DIVISOR LSB
012.231  323 100       3371          OUT     AP.DLL          OUTPUT TO ACE
012.233  043           3372          INX     H
012.234  176           3373          MOV     A,M             GET DIVISOR MSB
012.235  323 101       3374          OUT     AP.DLM          OUTPUT TO ACE
```

```
                        3375
                        3376  *       SET WORD CONFIGURATION
                        3377  *
012.237  257           3378          XRA     A               CLEAR ACC
012.240  260           3379          ORA     B               SEE IF 110 BAUD
012.241  006 000       3380          MVI     B,0             SET ONE STOP BIT IF NOT 110 BAUD
012.243  040 002       3381          JR      NZ,ACE1         IF (B) NOT 110 BAUD
                        3382
012.245  006 004       3383          MVI     B,AB.2SB        ELSE, SET TWO STOP BITS IN B
012.247  171           3384  ACE1    MOV     A,C             GET PARITY CONFIGURATION
012.250  346 160       3385          ANI     P1.PEN+P1.EPS+P1.SPS
012.252  017           3386          RRC                     SHIFT INTO POSITION FOR UART
012.253  260           3387          ORA     B               ADD NUMBER OF STOP BITS
012.254  107           3388          MOV     B,A             SAVE RESULT
012.255  346 010       3389          ANI     P1.PEN/2        SEE IF PARITY WAS ON
012.257  076 002       3390          MVI     A,AB.7BW        SET SEVEN BIT WORD IF PARITY ON
012.261  040 002       3391          JR      NZ,ACE2         IF PARITY ON
                        3392
012.263  076 003       3393          MVI     A,AB.8BW        ELSE, SET AN 8 BIT WORD WITH NO PARITY
012.265  260           3394  ACE2    ORA     B               ADD TO STOP BITS AND PARITY SELECT/TYPE
012.266  323 103       3395          OUT     AP.LCR          OUTPUT WORD SIZE AND PARITY SELECTION TO ACE
                        3396
012.270  076 001       3397          MVI     A,AB.ERDA       ENABLE RECEIVED DATA AVAILABLE INTERRUPTS
012.272  323 101       3398          OUT     AP.IER
012.274  076 003       3399          MVI     A,AB.DTR+AB.RTS     SET DATA TERMINAL READY
012.276  323 104       3400          OUT     AP.MCR
012.300  311           3401          RET


                        3403  **      BRTAB - BAUD RATE DIVISOR TABLE
                        3404  *
                        3405  *       *BRTAB* CONTAINS THE ACE DIVISOR LSB FOLLOWED BY THE MSB
                        3406  *
                        3407  *       TABLE MUST RESIDE IN ONE PAGE
                        3408
                        3409
012.301                 3410  BRTAB   EQU     *
012.301  321 006       3411  BR110   DB      209,6           110 BAUD
012.303  000 005       3412  BR150   DB      0,5             150 BAUD
012.305  173 002       3413  BR300   DB      123,2           300 BAUD
012.307  100 001       3414  BR600   DB      64,1            600 BAUD
012.311  240 000       3415  BR1200  DB      160,0           1200 BAUD
012.313  153 000       3416  BR1800  DB      107,0           1800 BAUD
012.315  140 000       3417  BR2000  DB      96,0            2000 BAUD
012.317  120 000       3418  BR2400  DB      80,0            2400 BAUD
012.321  065 000       3419  BR3600  DB      53,0            3600 BAUD
012.323  050 000       3420  BR4800  DB      40,0            4800 BAUD
012.325  033 000       3421  BR7200  DB      27,0            7200 BAUD
012.327  024 000       3422  BR9600  DB      20,0            9600 BAUD
012.331  012 000       3423  BR19.2K DB      10,0            19,200 BAUD
000.012                 3424  .       SET     BRTAB/256
000.000                 3425          ERRNZ   */256-.
```

```
                                3427  **      ICRT - INITIALIZE CRT CONTROLLER
                                3428  *
                                3429  *       *ICRT* SETS THE CRT CONTROLLER FOR AN 80 COLUMN, 24 LINE
                                3430  *       DISPLAY WITH THE DISPLAY HOME ADDRESS AND THE CURSOR ADDRESS
                                3431  *       AT *VRAMS*
                                3432  *
                                3433  *
                                3434  *       ENTRY   NONE
                                3435  *
                                3436  *       EXIT    NONE
                                3437  *
                                3438  *       USES    A,B,C,H,L,F
                                3439
                                3440
012.333                         3441  ICRT    EQU     *
                                3442
012.333  333 040               3443          IN      MP.PUP2         GET CONFIGURATION SWITCH INFO
                                3444
012.335  346 200               3445          ANI     P2.50HZ
012.337  041 351 017           3446          LXI     H,VPARD50       (H,L) = 50 HERTZ VIDEO PARAMETERS
012.342  040 003               3447          JR      NZ,ICRT0.5      IF SET FOR 50 HZ
                                3448
012.344  041 331 017           3449          LXI     H,VPARD60       (H,L) = 60 HERTZ VIDEO PARAMETERS
                                3450
012.347  006 020               3451  ICRT0.5 MVI     B,16            16 REGISTERS TO INITIALIZE IN CRTC
012.351  016 000               3452          MVI     C,0             START WITH REGISTER ZERO
                                3453
012.353  171                   3454  ICRT1   MOV     A,C             GET REGISTER ADDRESS
012.354  323 140               3455          OUT     VP.AR           SET ADDRESS REGISTER IN CRTC
012.356  176                   3456          MOV     A,M             GET DATA FOR CRTC REGISTER
012.357  323 141               3457          OUT     VP.REGO         OUTPUT TO REGISTER
012.361  014                   3458          INR     C               POINT TO NEXT REGISTER
012.362  043                   3459          INX     H               POINT TO NEXT REGISTER'S DATA
012.363  020 366               3460          DJNZ    ICRT1           IF NOT DONE WITH ALL REGISTERS
                                3461
012.365  311                   3462          RET
```

```
                                3464  **      IDT - IDENTIFY TERMINAL
                                3465  *
                                3466  *       *IDT* IDENTIFYS THE TERMINAL AS A DEC VT52 SO THAT EXISTING DEC
                                3467  *       SOFTWARE WHICH INTERROGATES THE CONSOLE TYPE WILL OPERATE AS IT
                                3468  *       WOULD WITH A VT52
                                3469  *
                                3470  *
                                3471  *       ENTRY   NONE
                                3472  *
                                3473  *       EXIT    NONE
                                3474  *
                                3475  *       USES    A,B,C,H,L,F
                                3476
                                3477
012.366  315 100 015           3478  IDT     CALL    PSOF            REPLY WITH *ESC / K*
012.371  033 057 313           3479          DB      ESC,'/','K'+200Q
```

```
012.374  311          3480          RET


                      3482  **      ARAMP - ANSI RESET ALL MODES TO POWER UP CONFIGURATION
                      3483  *
                      3484  *       *ARAMP* RESETS ALL FLAGS ETC., TO THE CONFIGURATION OF THE
                      3485  *       POWER UP SWITCHES
                      3486  *
                      3487  *
                      3488  *       ENTRY   (B) = ZERO IF NO PN WAS INPUT
                      3489  *
                      3490  *       EXIT    NONE
                      3491  *
                      3492  *       USES    ALL
                      3493
                      3494
012.375  170          3495  ARAMP   MOV     A,B             SEE IF PN WAS INPUT
012.376  267          3496          ORA     A
012.377  300          3497          RNZ                     IF PN WAS INPUT, ILLEGAL, EXIT
                      3498
                      3499  *       JMP     RAMP            ELSE, CONTINUE LIKE IN HEATH MODE
000.000               3500          ERRNZ   *-RAMP


                      3502  **      RAMP - RESET ALL MODES TO POWER UP CONFIGURATION
                      3503  *
                      3504  *       RAMP PROVIDES THE SAME FUNCTION AS A HARDWARE RESET
                      3505  *
                      3506  *
                      3507  *       ENTRY   NONE
                      3508  *
                      3509  *       EXIT    NONE
                      3510  *
                      3511  *       USES    ALL
                      3512
                      3513
013.000  363          3514  RAMP    DI                      LOCK OUT THE WORLD
                      3515  *       JMP     INIT
000.000               3516          ERRNZ   *-INIT


                      3518  ***     INIT - INITIALIZE SYSTEM ON POWER UP
                      3519  *
                      3520  *       INIT INITIALIZES THE SYSTEM BY CLEARING THE I/O PORTS, INITIALIZING
                      3521  *       THE SCRATCHPAD, INITIALIZING THE CRT CONTROLLER, INITIALIZING THE
                      3522  *       ACE SERIAL PORT, AND JUMPING TO THE MAIN CONTROL LOOP
                      3523
                      3524
013.001               3525  INIT    EQU     *
                      3526  *       JMP     IRAM            INITIALIZE RAM
```

```
000.000                  3527          ERRNZ   *-IRAM


                         3529  **      IRAM - INITIALIZE RAM
                         3530  *
                         3531  *       *IRAM* SETS ALL RAM LOCATIONS TO ZERO AND THEN COPYS IN THE
                         3532  *       DEFAULT DATA FROM PRSTAB
                         3533  *
                         3534  *
                         3535  *       ENTRY   NONE
                         3536  *
                         3537  *       EXIT    NONE
                         3538  *
                         3539  *       USES    A,B,C,D,E,H,L,F
                         3540
                         3541
013.001  041 000 100    3542  IRAM    LXI     H,RAM           POINT TO BEGINNING OF SCRATCHPAD
013.004  021 001 100    3543          LXI     D,RAM+1         COPY TO NEXT LOCATION
013.007  001 377 000    3544          LXI     B,255           COPY 255 TIMES
013.012  066 000        3545          MVI     M,0             PLACE ZERO IN FOR COPY
                         3546  *       LDIR                    COPY ZERO THROUGH RAM
013.014  355 260        3547          DB      I.LDIRA,I.LDIRB
                         3548
013.016  041 244 100    3549          LXI     H,KBDF          SET KEYBOARD FIFO
013.021  042 264 100    3550          SHLD    KBDFF
013.024  076 030        3551          MVI     A,24            SET VIDEO INFORMATION FOR NMI
013.026  062 276 100    3552          STA     VI.VD           VIDEO DISPLAYED
013.031  046 110        3553          MVI     H,VB.CBE+8      FAST BLINKING CURSOR ON LINE 8
013.033  056 010        3554          MVI     L,8             END ON LINE 8 TOO
013.035  042 277 100    3555          SHLD    VI.CSE
                         3556
013.040  333 000        3557          IN      MP.PUP1         INPUT POWER-UP SWITCH #1
                         3558
013.042  062 312 100    3559          STA     MODES           SAVE AS SERIAL I/O MODE
                         3560
013.045  333 040        3561          IN      MP.PUP2         INPUT POWER-UP SWITCH #2
013.047  346 177        3562          ANI     01111111B       TOSS 50HZ BIT
                         3563
013.051  062 310 100    3564          STA     MODEB           SAVE AS SPECIAL SET UP MODE
                         3565
013.054  333 040        3566          IN      MP.PUP2         INPUT POWER-UP SWITCH #2
013.056  346 177        3567          ANI     377Q-P2.50HZ    GET ALL BUT LINE FREQUENCY SWITCHES
013.060  062 310 100    3568          STA     MODEB           SAVE AS MODEB FLAGS
013.063  257            3569          XRA     A               CLEAR ALL INTERNAL FLAGS
013.064  062 311 100    3570          STA     MODEI           SAVE AS INTERNAL MODE WITH ALL OTHER FLAGS = ZERO
                         3571
                         3572
013.067  061 000 101    3573          LXI     SP,RAM+256      SET STACK POINTER TO TOP OF RAM
                         3574
013.072  041 000 370    3575          LXI     H,VRAMS         SET HOME, CURRENT LINE, AND CURSOR ADDRESSES
013.075  042 266 100    3576          SHLD    SHOME
013.100  042 270 100    3577          SHLD    CLSA
013.103  042 274 100    3578          SHLD    CURAD
013.106  021 000 010    3579          LXI     D,2048          WRITE SPACES IN ALL OF VIDEO RAM
```

```
013.111  315 160 016  3580           CALL    WSV
                      3581

                      3583  **       CONTINUE INITIALIZATION OF SYSTEM
                      3584  *
                      3585
013.114  315 333 012  3586  INIT1    CALL    ICRT            INITIALIZE CRTC
013.117  315 204 012  3587           CALL    IACE            INITIALIZE ACE
013.122  315 371 010  3588           CALL    EC              SET PROPER CURSOR TYPE
                      3589  *        IM1                     SET INTERRUPT MODE ONE
013.125  355 126      3590           DB      I.IM1A,I.IM1B
013.127  076 000      3591           MVI     A,0             CAUSE FIRST NMI
013.131  323 144      3592           OUT     VP.AR+VB.NMI
013.133  373          3593           EI                      LET IT ALL BEGIN
013.134  303 004 000  3594           JMP     MAIN            GO TO MAIN LOOP


                      3596  **       MPY80 - MULTIPLY BY EIGHTY
                      3597  *
                      3598  *        MULTIPLY AN 8 BIT NUMBER BY EIGHTY
                      3599  *
                      3600  *
                      3601  *        ENTRY   (A) = MULTIPLICAND
                      3602  *
                      3603  *        EXIT    (H,L) = (A)*80
                      3604  *
                      3605  *        USES    D,E,H,L,F
                      3606
                      3607
013.137  157          3608  MPY80    MOV     L,A             VALUE TO MULTIPLY TO (H,L)
013.140  046 000      3609           MVI     H,0
013.142  051          3610           DAD     H               *16
013.143  051          3611           DAD     H
013.144  051          3612           DAD     H
013.145  051          3613           DAD     H
013.146  124          3614           MOV     D,H             *5
013.147  135          3615           MOV     E,L
013.150  031          3616           DAD     D
013.151  031          3617           DAD     D
013.152  031          3618           DAD     D
013.153  031          3619           DAD     D
013.154  311          3620           RET
```

```
                        3622  **      NKC - NO KEYBOARD CLICK
                        3623  *
                        3624  *       *NKC* SETS THE FLAG WHICH DISABLES THE KEYBOARD CLICK DURING *AKI*
                        3625  *
                        3626  *
                        3627  *       ENTRY   (B,C) = MODEB
                        3628  *
                        3629  *       EXIT    NONE
                        3630  *
                        3631  *       USES    A,F
                        3632
                        3633
013.155  012           3634  NKC     LDAX    B               GET CURRENT FLAGS
013.156  366 002       3635          ORI     MB.NOTK         SET NO TICK
013.160  002           3636          STAX    B
013.161  311           3637          RET


                        3639  **      PCA - PERFORM CURSOR ADDRESSING
                        3640  *
                        3641  *       *PCA* SETS THE CURSOR LINE AND COLUMN VALUES ACCORDING TO THE
                        3642  *       NEXT TWO BYTES FROM THE INPUT FIFO.  LINE NUMBER 40Q IS THE TOP
                        3643  *       LINE OF THE DISPLAY.  COLUMN 40Q IS THE LEFTMOST COLUMN.  AN ILLEGAL
                        3644  *       LINE NUMBER WILL CAUSE THE CURSOR TO REMAIN ON THE CURRENT LINE.
                        3645  *       AN ILLEGAL COLUMN NUMBER WILL CAUSE THE CURSOR TO BE PLACED AT
                        3646  *       THE END OF THE CURRENT LINE.
                        3647  *
                        3648  *
                        3649  *       ENTRY   NONE
                        3650  *
                        3651  *       EXIT    NONE
                        3652  *
                        3653  *       USES    A,B,C,D,E,H,L,F
                        3654
                        3655
013.162  315 070 012   3656  PCA     CALL    FNCP            FETCH NEXT INPUT FIFO CHARACTER
013.165  376 030       3657          CPI     CAN             SEE IF TO CANCEL THIS SEQUENCE
013.167  310           3658          RZ                      IF CANCEL
                        3659
                        3660  *       HAVE A LINE NUMBER, PROCESS IT
                        3661  *
013.170  376 040       3662  PCA1    CPI     40Q             SEE IF LINE NUMBER IS IN RANGE
013.172  070 024       3663          JR      C,PCA2          IF LESS THAN LINE ZERO, USE SAME LINE
                        3664
013.174  376 070       3665          CPI     40Q+24
013.176  070 013       3666          JR      C,PCA1.5        IF LINE 23 OR LESS
                        3667
013.200  040 016       3668          JR      NZ,PCA2         IF NOT 24, USE SAVE
                        3669
013.202  107           3670          MOV     B,A             ELSE, SAVE LINE NUMBER
013.203  072 311 100   3671          LDA     MODEI           GET MODE FLAGS
013.206  346 200       3672          ANI     MI.25L          SEE IF 25TH LINE IS ENABLED
013.210  050 006       3673          JR      Z,PCA2          IF LINE 24 REQUESTED AND NOT AVAILABLE
                        3674
```

```
013.212  170            3675           MOV     A,B             ELSE, GET LINE 24 VALUE BACK AND USE IT
                        3676
013.213  326 040        3677  PCA1.5   SUI     40Q             MASK FOR LINE VALUE
013.215  062 273 100    3678           STA     CURVP           SAVE NEW VERTICAL POSITION
                        3679
013.220  315 070 012    3680  PCA2     CALL    FNCP            FETCH NEXT INPUT FIFO CHARACTER
013.223  376 030        3681           CPI     CAN             SEE IF TO CANCEL NOW
013.225  050 017        3682           JR      Z,PCA6          IF TO CANCEL
                        3683
                        3684  *        HAVE A COLUMN NUMBER
                        3685  *
013.227  376 040        3686  PCA3     CPI     40Q             SEE IF IN RANGE
013.231  070 004        3687           JR      C,PCA4          IF LESS THAN COLUMN ZERO
                        3688
013.233  376 160        3689           CPI     40Q+80
013.235  070 002        3690           JR      C,PCA5          IF COLUMN ZERO THRU 79
                        3691
013.237  076 157        3692  PCA4     MVI     A,40Q+79        OUT OF RANGE, SET CURSOR TO END OF LINE
013.241  326 040        3693  PCA5     SUI     40Q             MASK FOR COLUMN VALUE
013.243  062 272 100    3694           STA     CURHP           UPDATE COLUMN COUNTER
                        3695
013.246  303 322 015    3696  PCA6     JMP     SNCP            SET CURSOR POSITION
```

```
                        3698  **       PCIF - PLACE CHARACTER IN FIFO
                        3699  *
                        3700  *        *PCIF* PLACES A SINGLE CHARACTER INTO THE INPUT FIFO
                        3701  *
                        3702  *
                        3703  *        ENTRY   (A) = CHARACTER
                        3704  *
                        3705  *        EXIT    (A) = CHARACTER
                        3706  *                'C' = SET IF NO ROOM IN FIFO
                        3707  *
                        3708  *        USES    A,B,C,H,L,F
                        3709
                        3710
013.251  117            3711  PCIF     MOV     C,A             SAVE CHARACTER FOR FIFO
013.252  072 241 100    3712           LDA     IFC             GET INPUT FIFO COUNTER
013.255  376 200        3713           CPI     IFMAX           CHECK FOR FIFO ALREADY FULL
013.257  067            3714           STC                     SET 'C' FOR FIFO FULL
013.260  312 223 010    3715           JZ      DING            DING BELL AND EXIT
                        3716
013.263  107            3717           MOV     B,A             ELSE, SAVE FIFO COUNTER
013.264  072 240 100    3718           LDA     IFP             GET INPUT FIFO POINTER
013.267  200            3719           ADD     B               ADD COUNT FOR VECTOR TO NEXT CHARACTER ADDRESS
013.270  346 177        3720           ANI     IFCMSK          KEEP VECTOR IN FIFO
013.272  041 000 100    3721           LXI     H,INF           POINT TO INPUT FIFO
013.275  205            3722           ADD     L               ADD VECTOR
013.276  157            3723           MOV     L,A
013.277  161            3724           MOV     M,C             PLACE CHARACTER IN FIFO
013.300  170            3725           MOV     A,B             INCREMENT INPUT FIFO COUNTER
013.301  074            3726           INR     A
013.302  062 241 100    3727           STA     IFC
```

```
013.305  067           3728          STC                     CLEAR 'C' TO SHOW THAT CHARACTER WAS PLACED
013.306  077           3729          CMC
013.307  171           3730          MOV     A,C             CHARACTER TO (A)
013.310  311           3731          RET


                       3733  **      PCOF - PLACE CHARACTER IN OUTPUT FIFO
                       3734  *
                       3735  *       *PCOF* PLACES A SINGLE CHARACTER INTO THE OUTPUT FIFO
                       3736  *
                       3737  *
                       3738  *       ENTRY   (A) = CHARACTER
                       3739  *
                       3740  *       EXIT    (A) = CHARACTER
                       3741  *               'C' = SET IF NO ROOM IN FIFO
                       3742  *
                       3743  *       USES    A,B,C,H,L,F
                       3744
                       3745
013.311  117           3746  PCOF    MOV     C,A             SAVE CHARACTER
013.312  072 311 100   3747          LDA     MODE1           GET MODE FLAGS
013.315  346 010       3748          ANI     MI.ONLN         IS TERMINAL ON-LINE?
013.317  040 005       3749          JR      NZ,PCOF1        IF TERMINAL IS ON-LINE
                       3750
                       3751  *       TERMINAL IS OFF-LINE, PLACE CHARACTER IN INPUT FIFO INSTEAD
                       3752  *
013.321  171           3753          MOV     A,C
013.322  315 251 013   3754          CALL    PCIF            PLACE CHARACTER IN INPUT FIFO
013.325  311           3755          RET
                       3756
013.326  072 312 100   3757  PCOF1   LDA     MODES           GET MODE FOR SERIAL I/O
013.331  346 200       3758          ANI     MS.FDX          IS FULL DUPLEX SELECTED?
013.333  171           3759          MOV     A,C             (A) = CHARACTER
013.334  314 251 013   3760          CZ      PCIF            IF HALF DUPLEX, PLACE CHARACTER IN BOTH FIFOS
                       3761
013.337  072 243 100   3762          LDA     OFC             GET OUTPUT FIFO COUNTER
013.342  376 040       3763          CPI     OFMAX           SEE IF FIFO IS FULL
013.344  067           3764          STC                     SET 'C' FOR FULL
013.345  050 025       3765          JR      Z,PCOF2         IF FIFO IS ALREADY FULL
                       3766
013.347  107           3767          MOV     B,A             SAVE CURRENT COUNT
013.350  072 242 100   3768          LDA     OFP             GET OUTPUT FIFO POINTER
013.353  200           3769          ADD     B               ADD TO COUNT FOR VECTOR TO ADDRESS FOR THIS CHAR
013.354  346 037       3770          ANI     OFCMSK          KEEP VECTOR IN FIFO
013.356  041 200 100   3771          LXI     H,OUTF          POINT TO OUTPUT FIFO
013.361  205           3772          ADD     L               ADD VECTOR
013.362  157           3773          MOV     L,A
013.363  161           3774          MOV     M,C             PUT CHARACTER IN FIFO
013.364  170           3775          MOV     A,B             INCREMENT OUTPUT FIFO COUNTER
013.365  074           3776          INR     A
013.366  062 243 100   3777          STA     OFC
013.371  067           3778          STC                     CLEAR 'C' TO SHOW CHARACTER PLACED
013.372  077           3779          CMC
013.373  171           3780          MOV     A,C             (A) = OUTPUT CHARACTER
```

```
013.374  311              3781  PCOF2   RET

                          3783  **      PCOFT - PLACE CHARACTER IN OUTPUT FIFO DURING TASK TIME
                          3784  *
                          3785  *
                          3786  *       ENTRY   (A) = CHARACTER
                          3787  *
                          3788  *       EXIT    'C' SET IF NO ROOM
                          3789  *
                          3790  *       USES    A,B,C,H,L,F
                          3791
                          3792
013.375  363              3793  PCOFT   DI                      LOCK OUT INTERRUPT CHARACTER REMOVALS
013.376  315 311 013      3794          CALL    PCOF            PLACE CHARACTER
014.001  373              3795          EI                      ALLOW INTERRUPTS NOW
014.002  311              3796          RET


                          3798  **      PCRLF - PERFORM CARRIAGE RETURN AND/OR LINE FEED
                          3799  *
                          3800  *       *PCRLF* PERFORMS A LINE FEED PRIOR TO PERFORMING A CARRIAGE
                          3801  *       RETURN IF THE AUTO LINE FEED FUNCTION IS SELECTED
                          3802  *
                          3803  *
                          3804  *       ENTRY   NONE
                          3805  *
                          3806  *       EXIT    NONE
                          3807  *
                          3808  *       USES    A,H,L,F
                          3809
                          3810
014.003  072 310 100      3811  PCRLF   LDA     MODEB           GET MODE FLAGS
014.006  346 010          3812          ANI     MB,ALF          SEE IF AUTO LINE FEED IS SELECTED
014.010  304 354 006      3813          CNZ     PLF             IF SELECTED
                          3814  *       JMP     PCR             PERFORM CARRIAGE RETURN
000.000                   3815          ERRNZ   *-PCR


                          3817  **      PCR - PERFORM CARRIAGE RETURN
                          3818  *
                          3819  *       PCR MOVES THE CURSOR TO THE BEGINNING OF THE CURRENT LINE
                          3820  *
                          3821  *
                          3822  *       ENTRY   NONE
                          3823  *
                          3824  *       EXIT    NONE
                          3825  *
                          3826  *       USES    A,H,L,F
                          3827
```

```
                        3828
014,013  076 000       3829  PCR     MVI     A,0             SET CURSOR HORIZONTAL POSITION TO ZERO
014.015  062 272 100   3830          STA     CURHP
014.020  052 270 100   3831          LHLD    CLSA            SET CURSOR ADDRESS TO BEGINNING OF LINE
014.023  042 274 100   3832          SHLD    CURAD
                        3833
                        3834
014.026  311           3835          RET                     EXIT



                        3837  **      PDC - PERFORM DELETE CHARACTER
                        3838  *
                        3839  *       *PDC* DELETES THE CHARACTER AT THE CURSOR POSITION BY MOVING
                        3840  *       THE REMAINING CHARACTERS ON THE LINE TO THE LEFT ONE COLUMN AND
                        3841  *       INSERTING A SPACE IN THE LAST COLUMN ON THE LINE
                        3842  *
                        3843  *
                        3844  *       ENTRY   NONE
                        3845  *
                        3846  *       EXIT    NONE
                        3847  *
                        3848  *       USES    A,B,C,D,E,H,L,F
                        3849
                        3850
014.027                 3851  PDC     EQU     *
                        3852
                        3853
014.027  072 272 100   3854          LDA     CURHP           GET CURRENT COLUMN POSITION
                        3855  *       NEG                     SUBTRACT FROM 79 FOR NUMBER OF MOVES
014.032  355 104       3856          DB      I.NEGA,I.NEGB
014.034  306 117       3857          ADI     79
014.036  117           3858          MOV     C,A             SAVE RESULT
014.037  006 000       3859          MVI     B,0
                        3860  *       LD      DE,(CURAD)      (D,E) = CURSOR ADDRESS
014.041  355 133       3861          DB      I.LDEDA,I.LDEDB
014.043  274 100       3862          DW      CURAD
014.045  050 022       3863          JR      Z,PDC2          IF ALREADY AT COLUMN 79
                        3864
014.047  052 274 100   3865          LHLD    CURAD           (H,L) = CURSOR ADDRESS + 1
014.052  043           3866          INX     H
                        3867
014.053  172           3868  PDC1    MOV     A,D             KEEP POINTERS IN VIDEO RAM
014.054  366 370       3869          ORI     VRAMS/256
014.056  127           3870          MOV     D,A
014.057  174           3871          MOV     A,H
014.060  366 370       3872          ORI     VRAMS/256
014.062  147           3873          MOV     H,A
                        3874  *       LDI                     COPY FROM (H,L) TO (D,E)
014.063  355 240       3875          DB      I.LDIA,I.LDIB
014.065  170           3876          MOV     A,B             SEE IF (B,C) = ZERO
014.066  261           3877          ORA     C
014.067  040 362       3878          JR      NZ,PDC1         IF NOT DONE WITH LAST CHARACTER
                        3879
014.071  076 040       3880  PDC2    MVI     A,' '           PUT SPACE IN LAST COLUMN
```

```
014.073  022           3881           STAX    D
014.074  311           3882           RET


                       3884  **       PDL - PERFORM DELETE LINE
                       3885  *
                       3886  *        *PDL* MOVES THE REMAINING LINES OF THE DISPLAY UP ONE LINE, WRITES
                       3887  *        SPACES INTO THE LAST LINE, AND MOVES THE CURSOR TO THE BEGINNING
                       3888  *        OF THE CURRENT LINE.
                       3889  *
                       3890  *
                       3891  *        ENTRY   NONE
                       3892  *
                       3893  *        EXIT    NONE
                       3894  *
                       3895  *        USES    A,B,C,D,E,H,L,F
                       3896
                       3897
014.075                3898  PDL      EQU     *
                       3899
                       3900
014.075  076 000      3901           MVI     A,0             SET COLUMN COUNTER TO ZERO
014.077  062 272 100  3902           STA     CURHP
014.102  052 270 100  3903           LHLD    CLSA            SET *CURAD* TO BEGINNING OF THIS LINE
014.105  042 274 100  3904           SHLD    CURAD
014.110  353          3905           XCHG                    (D,E) = BEGINNING OF THIS LINE
014.111  041 120 000  3906           LXI     H,80            SET (H,L) TO BEGINNING OF NEXT LINE
014.114  031          3907           DAD     D
014.115  072 273 100  3908           LDA     CURVP           GET CURRENT LINE NUMBER
                       3909  *        NEG                     SUBTRACT FROM 23 FOR NUMBER OF LINES TO MOVE
014.120  355 104      3910           DB      I.NEGA,I.NEGB
014.122  306 027      3911           ADI     23
014.124  050 031      3912           JR      Z,PDL2          IF ON LINE 23, JUST BLANK IT
                       3913
014.126  117          3914           MOV     C,A             LINES LEFT*5 = MOD 16 MOVE COUNT
014.127  201          3915           ADD     C
014.130  201          3916           ADD     C
014.131  201          3917           ADD     C
014.132  201          3918           ADD     C
014.133  117          3919           MOV     C,A
                       3920
014.134  305          3921  PDL1     PUSH    B               SAVE (B,C)
014.135  172          3922           MOV     A,D             KEEP POINTERS IN VIDEO RAM
014.136  366 370      3923           ORI     VRAMS/256
014.140  127          3924           MOV     D,A
014.141  174          3925           MOV     A,H
014.142  366 370      3926           ORI     VRAMS/256
014.144  147          3927           MOV     H,A
014.145  016 020      3928           MVI     C,16            COPY 16 BYTES
014.147  006 000      3929           MVI     B,0
                       3930  *        LDIR
014.151  355 260      3931           DB      I.LDIRA,I.LDIRB
014.153  301          3932           POP     B               RESTORE (B,C)
014.154  015          3933           DCR     C               DECREMENT MOD 16 COUNT
```

```
014.155  040 355      3934          JR      NZ,PDL1         IF NOT DONE, DO 16 MORE
                       3935
014.157  353          3936  PDL2    XCHG                    (H,L) = LINE 23
014.160  006 005      3937          MVI     B,5             ERASE 80 CHARACTERS (5*16)
014.162  303 212 016  3938          JMP     WSVA            WRITE 80 SPACES AND EXIT


                       3940  **      PIC - PERFORM INSERT CHARACTER
                       3941  *
                       3942  *       *PIC* MOVES ALL CHARACTERS THAT ARE AT AND TO THE RIGHT OF THE
                       3943  *       CURSOR ONE COLUMN TO THE RIGHT
                       3944  *
                       3945  *
                       3946  *       ENTRY   NONE
                       3947  *
                       3948  *       EXIT    NONE
                       3949  *
                       3950  *       USES    A,B,C,D,E,H,L,F
                       3951
                       3952
014.165                3953  PIC     EQU     *
                       3954
                       3955
014.165  072 272 100  3956          LDA     CURHP           GET CURRENT COLUMN POSITION
                       3957  *       NEG                     SUBTRACT FROM 79 FOR NUMBER OF CHAR. TO MOVE
014.170  355 104      3958          DB      I,NEGA,I,NEGB
014.172  306 117      3959          ADI     79
014.174  310          3960          RZ                      IF AT COLUMN 79, NO COPY NEEDED
                       3961
014.175  117          3962          MOV     C,A             (B,C) = RESULT
014.176  006 000      3963          MVI     B,0
014.200  052 270 100  3964          LHLD    CLSA            GET CURRENT LINE STARTING ADDRESS
014.203  021 117 000  3965          LXI     D,79            ADD 79 FOR END OF LINE ADDRESS
014.206  031          3966          DAD     D
014.207  124          3967          MOV     D,H             (D,E) = ADDRESS OF END OF LINE
014.210  135          3968          MOV     E,L
014.211  053          3969          DCX     H               (H,L) = ADDRESS OF END OF LINE -1
                       3970
014.212  172          3971  PIC1    MOV     A,D             STAY IN VIDEO RAM
014.213  366 370      3972          ORI     VRAMS/256
014.215  127          3973          MOV     D,A
014.216  174          3974          MOV     A,H
014.217  366 370      3975          ORI     VRAMS/256
014.221  147          3976          MOV     H,A
                       3977  *       LDD                     COPY CHARACTER
014.222  355 250      3978          DB      I,LDDA,I,LDDB
014.224  170          3979          MOV     A,B             SEE IF COUNT EXHAUSTED
014.225  261          3980          ORA     C
014.226  040 362      3981          JR      NZ,PIC1         IF NOT DONE YET
                       3982
014.230  052 274 100  3983          LHLD    CURAD           GET CURSOR ADDRESS TO PLACE CHARACTER
014.233  311          3984          RET
```

```
                                3986  **      PIL - PERFORM INSERT LINE
                                3987  *
                                3988  *       *PIL* INSERTS A BLANK LINE AT THE CURSOR POSITION AFTER MOVING
                                3989  *       THE REMAINING LINES DOWN ONE LINE.  DATA ON LINE 23 IS LOST.
                                3990  *
                                3991  *
                                3992  *       ENTRY   NONE
                                3993  *
                                3994  *       EXIT    NONE
                                3995  *
                                3996  *       USES    A,B,C,D,E,H,L,F
                                3997
                                3998
014.234                         3999  PIL     EQU     *
                                4000
                                4001
014.234  072 273 100           4002          LDA     CURVP           GET CURRENT LINE NUMBER
                                4003  *       NEG                     SUBTRACT FROM 23 FOR NUMBER OF LINES TO MOVE
014.237  355 104                4004          DB      I.NEGA,I.NEGB
014.241  306 027                4005          ADI     23
014.243  050 061                4006          JR      Z,PIL2          IF ON LAST LINE, SKIP COPY
                                4007
014.245  117                   4008          MOV     C,A             LINES*5 = MOD 16 CHARACTER COUNT
014.246  201                   4009          ADD     C
014.247  201                   4010          ADD     C
014.250  201                   4011          ADD     C
014.251  201                   4012          ADD     C
014.252  117                   4013          MOV     C,A             (C) = RESULT
                                4014  *       LD      DE,(SHOME)      (D,E) = CURRENT HOME POSITION
014.253  355 133                4015          DB      I.LDEDA,I.LDEDB
014.255  266 100                4016          DW      SHOME
014.257  041 177 007           4017          LXI     H,1919          ADD NUMBER OF CHARACTERS TO POINT TO LAST CHAR.
014.262  031                   4018          DAD     D               (H,L) = LAST ADDRESS IN DISPLAY
014.263  345                   4019          PUSH    H               SAVE ON STACK
014.264  041 057 007           4020          LXI     H,1839          ADD NUMBER OF CHARACTERS TO END OF LINE 22
014.267  031                   4021          DAD     D               (H,L) = END OF LINE 22
014.270  321                   4022          POP     D               (D,E) = END OF LINE 23
                                4023
014.271  305                   4024  PIL1    PUSH    B               SAVE (B,C)
014.272  172                   4025          MOV     A,D             KEEP POINTERS IN VIDEO RAM
014.273  366 370                4026          ORI     VRAMS/256
014.275  127                   4027          MOV     D,A
014.276  174                   4028          MOV     A,H
014.277  366 370                4029          ORI     VRAMS/256
014.301  147                   4030          MOV     H,A
014.302  016 020                4031          MVI     C,16            COPY 16 BYTES
014.304  006 000                4032          MVI     B,0
                                4033  *       LDDR
014.306  355 270                4034          DB      I.LDDRA,I.LDDRB
014.310  301                   4035          POP     B               RESTORE (B,C)
014.311  015                   4036          DCR     C               COPY COMPLETE?
014.312  040 355                4037          JR      NZ,PIL1         IF NOT DONE
                                4038
014.314  172                   4039          MOV     A,D             KEEP POINTERS IN VIDEO RAM
014.315  366 370                4040          ORI     VRAMS/256
014.317  127                   4041          MOV     D,A
```

```
014,320  174            4042          MOV     A,H
014,321  366 370        4043          ORI     VRAMS/256
014,323  147            4044          MOV     H,A
                        4045  *       LDD                     COPY FIRST CHARACTER ON THIS LINE
014,324  355 250        4046          DB      I.LDDA,I.LDDB
                        4047
014,326  076 000        4048  PIL2    MVI     A,0             SET CURSOR TO BENINNING OF LINE
014,330  062 272 100    4049          STA     CURHP
014,333  052 270 100    4050          LHLD    CLSA            SET CURAD TO BEGINNING OF LINE
014,336  042 274 100    4051          SHLD    CURAD
014,341  006 005        4052          MVI     B,5             WRITE 80 SPACES (5*16)
014,343  303 212 016    4053          JMP     WSVA



                        4055  **      PSD - PARAMETER STRING DECODER
                        4056  *
                        4057  *       *PSD* INPUTS A PARAMETER STRING OF DECIMAL NUMBERS SEPARATED BY
                        4058  *       A SEMICOLON (TO A MAX OF 15) UNTIL THE FINAL CHARACTER OF THE
                        4059  *       ESCAPE SEQUENCE (NON DECIMAL AND NOT A SEMICOLON) IS INPUT
                        4060  *
                        4061  *
                        4062  *       ENTRY   NONE
                        4063  *
                        4064  *       EXIT    (A) = FINAL CHARACTER
                        4065  *               (B) = ZERO IF NO PARAMETER STRING PRECEEDING FINAL CHARACTER
                        4066  *
                        4067  *       USES    A,B,C,D,H,L,F
                        4068
                        4069
014,346  041 314 100    4070  PSD     LXI     H,PSDW          POINT TO PSD WORK AREA
014,351  001 000 000    4071          LXI     B,0             NO PARAMETER STRING YET
014,354  160            4072          MOV     M,B             FIRST VALUE = ZERO
014,355  315 070 012    4073          CALL    FNCP            INPUT FIRST CHARACTER
014,360  376 073        4074          CPI     ';'             CAN'T HAVE SEMICOLON AS FIRST CHARACTER
014,362  310            4075          RZ                      IF SEMICOLON, END SEQUENCE
                        4076
014,363  376 060        4077  PSD1    CPI     '0'             SEE IF LESS THAN DECIMAL
014,365  070 046        4078          JR      C,PSD4          IF TOO LOW, EXIT
                        4079
014,367  376 072        4080          CPI     '9'+1           SEE IF GREATER THAN DECIMAL
014,371  060 017        4081          JR      NC,PSD2         IF NOT DECIMAL
                        4082
                        4083  *       INPUT CHARACTER IS DECIMAL
                        4084  *
014,373  346 017        4085          ANI     00001111B       MASK FOR BINARY
014,375  127            4086          MOV     D,A             SAVE RESULT
                        4087
                        4088  *       NEW DIGITS VALUE, MULTIPLY OLD VALUE BY TEN
                        4089  *
014,376  176            4090          MOV     A,M             GET OLD VALUE
014,377  007            4091          RLC                     *2
015,000  147            4092          MOV     M,A
015,001  007            4093          RLC                     *4
015,002  007            4094          RLC                     *8
```

```
015.003  206           4095           ADD     M               *2 + *8 = *10
015.004  202           4096           ADD     D               ADD NEW DIGITS
015.005  167           4097           MOV     M,A             SAVE RESULT
015.006  004           4098           INR     B               SHOW THAT A FN HAS BEEN INPUT
015.007  014           4099           INR     C
015.010  030 016       4100           JR      PSD3.5          INPUT NEXT CHARACTER
                       4101
015.012  376 073       4102  PSD2     CPI     ';'             SEE IF CHARACTER IS A SEMICOLON
015.014  040 017       4103           JR      NZ,PSD4         IF NOT, THEN CHARACTER IS FINAL
                       4104
015.016  175           4105           MOV     A,L             ELSE, SEE IF ROOM FOR THIS PARAMETER
015.017  376 332       4106           CPI     PSDWE-2         SAVE ROOM FOR LAST PLUS FINAL
015.021  050 001       4107           JR      Z,PSD3          IF NO MORE ROOM
                       4108
015.023  043           4109           INX     H               POINT TO NEXT CELL
015.024  016 000       4110  PSD3     MVI     C,0             RESET PARAMETER BUILD REGISTER
015.026  066 000       4111           MVI     M,0             TOSS OLD FN
015.030  315 070 012   4112  PSD3.5   CALL    FNCP            GET NEXT CHARACTER
015.033  030 326       4113           JR      PSD1            DECODE CHARACTER
                       4114
015.035  127           4115  PSD4     MOV     D,A             SAVE FINAL CHARACTER
015.036  257           4116           XRA     A               SEE IF FN IS IN MEMORY AT CURRENT ADDRESS
015.037  261           4117           ORA     C
015.040  050 001       4118           JR      Z,PSD5          IF NOTHING IN PROGRESS
                       4119
015.042  043           4120           INX     H               ELSE, PUT FINAL IN NEXT BYTE
015.043  162           4121  PSD5     MOV     M,D             PUT FINAL CHARACTER IN WORK AREA
015.044  172           4122           MOV     A,D             AND (A)
015.045  021 314 100   4123           LXI     D,PSDW          (D,E) = PSDW
015.050  311           4124           RET
```

```
                       4126  **       PSIF - PUT STRING IN INPUT FIFO
                       4127  *
                       4128  *        *PSIF* PLACES THE STRING IMMEDIATELY FOLLOWING THE CALL TO
                       4129  *        THIS ROUTINE INTO THE INPUT FIFO
                       4130  *
                       4131  *
                       4132  *        ENTRY   NONE
                       4133  *
                       4134  *        EXIT    NONE
                       4135  *
                       4136  *        USES    A,B,C,D,E,H,L,F
                       4137
                       4138
015.051  321           4139  PSIF     POP     D               GET ADDRESS OF CHARACTERS TO OUTPUT
015.052  032           4140  PSIF1    LDAX    D               GET CHARACTER
015.053  023           4141           INX     D               POINT TO NEXT CHARACTER
015.054  267           4142           ORA     A               SET CPU FLAGS
015.055  372 067 015   4143           JM      PSIF2           IF THIS IS LAST CHARACTER TO OUTPUT
                       4144
015.060  363           4145           DI                      LOCK OUT INTERRUPTS
015.061  315 251 013   4146           CALL    PCIF            PLACE CHARACTER
015.064  373           4147           EI
```

```
015.065  030 363        4148           JR      PSIF1           GET NEXT CHARACTER
                        4149
015.067  346 177        4150  PSIF2    ANI     01111111B       MASK OFF TERMINATOR
015.071  325            4151           PUSH    D               SET RETURN ADDRESS BACK ON STACK
015.072  363            4152           DI
015.073  315 251 013     4153           CALL    PCIF            PLACE LAST CHARACTER
015.076  373            4154           EI
015.077  311            4155           RET


                        4157  **       PSOF - PUT STRING IN OUTPUT FIFO
                        4158  *
                        4159  *        *PSOF* PLACES THE CHARACTER STRING IMMEDIATELY FOLLOWING THE
                        4160  *        CALL TO THIS ROUTINE INTO THE OUTPUT FIFO
                        4161  *
                        4162  *
                        4163  *        ENTRY   NONE
                        4164  *
                        4165  *        EXIT    NONE
                        4166  *
                        4167  *        USES    A,B,C,D,E,H,L,F
                        4168
                        4169
015.100  321            4170  PSOF     POP     D               GET ADDRESS OF CHARACTERS TO OUTPUT
015.101  032            4171  PSOF1    LDAX    D               GET CHARACTER
015.102  023            4172           INX     D               POINT TO NEXT CHARACTER
015.103  267            4173           ORA     A               SET CPU FLAGS
015.104  372 114 015     4174           JM      PSOF2           IF THIS IS LAST CHARACTER TO OUTPUT
                        4175
015.107  315 375 013     4176           CALL    PCOFT           PLACE CHARACTER IN OUTPUT FIFO
015.112  030 365        4177           JR      PSOF1           GET NEXT CHARACTER
                        4178
015.114  346 177        4179  PSOF2    ANI     01111111B       TOSS TERMINATOR BIT
015.116  325            4180           PUSH    D               SET RETURN ADDRESS
015.117  303 375 013     4181           JMP     PCOFT           OUTPUT LAST CHARACTER


                        4183  **       *RMS* RESETS THE MODE SPECIFIED BY THE LAST CHARACTER IN THE SEQUENCE
                        4184  *
                        4185  *
                        4186  *        ENTRY   (B,C) = MODEB
                        4187  *                (D,E) = MODEA
                        4188  *
                        4189  *        EXIT    NONE
                        4190  *
                        4191  *        USES    A,B,C,D,E,H,L,F
                        4192
                        4193
015.122  041 227 015     4194  RMS      LXI     H,RMST          (H,L) = RESET MODE SEQUENCE TABLE
015.125  030 015        4195           JR      SMSA            CONTINUE SAME AS SET MODE SEQUENCE
```

```
                        4197  **      ASCP - ANSI SAVE CURSOR POSITION
                        4198  *
                        4199  *       *ASCP* SAVES THE CURRENT LINE AND COLUMN NUMBERS OF THE CURSOR
                        4200  *       POSITION
                        4201  *
                        4202  *
                        4203  *       ENTRY   (B) = ZERO IF NO PN WAS INPUT
                        4204  *
                        4205  *       EXIT    NONE
                        4206  *
                        4207  *       USES    A,H,L,F
                        4208
                        4209
015,127  170           4210  ASCP    MOV     A,B             SEE IF PN WAS INPUT
015,130  267           4211          ORA     A
015,131  300           4212          RNZ                     IF INPUT, ILLEGAL, EXIT
                        4213
                        4214  *       JMP     SCP             ELSE, SAVE CURSOR POSITION
000,000                 4215          ERRNZ   *-SCP


                        4217  **      SCP - SAVE CURSOR POSITION
                        4218  *
                        4219  *       *SCP* SAVES THE CURRENT POSITION OF THE CURSOR WHICH MAY BE RESTORED
                        4220  *       WITH THE USE OF *USCP*
                        4221  *
                        4222  *
                        4223  *       ENTRY   NONE
                        4224  *
                        4225  *       EXIT    NONE
                        4226  *
                        4227  *       USES    H,L
                        4228
                        4229
015,132  052 272 100    4230  SCP     LHLD    CURHP           GET CURSOR POSITIONS
000,000                 4231          ERRNZ   CURVP-CURHP-1   MUST BE CONTIGUOUS
015,135  042 305 100    4232          SHLD    CSA             KEEP IN CURSOR SAVED ADDRESS
015,140  311            4233          RET


                        4235  **      SMS - SET MODE SEQUENCE
                        4236  *
                        4237  *       *SMS* SETS THE MODE SPECIFIED BY THE LAST CHARACTER IN THE SEQUENCE
                        4238  *
                        4239  *
                        4240  *       ENTRY   (B,C) = MODEB
                        4241  *               (D,E) = MODEA
                        4242  *
                        4243  *       EXIT    NONE
                        4244  *
                        4245  *       USES    A,B,C,D,E,H,L,F
                        4246
```

```
                                4247
015.141  041 205 015           4248  SMS     LXI     H,SMST          (H,L) = SET MODE SEQUENCE TABLE
                                4249
015.144  315 070 012           4250  SMSA    CALL    FNCP            GET FINAL SEQUENCE CHARACTER
015.147  376 030                 4251          CPI     CAN             SEE IF TO CANCEL
015.151  310                     4252          RZ                      IF CANCEL, EXIT NOW
                                4253
015.152  376 061                 4254          CPI     '1'             CAN'T BE LOWER THAN ONE
015.154  332 120 003             4255          JC      IFCP            IF LOWER, EXIT LIKE NEVER HERE
                                4256
015.157  376 072                 4257          CPI     '9'+1           <= 9
015.161  322 120 003             4258          JNC     IFCP            IF GREATER THAN 9
                                4259
015.164  346 017                 4260          ANI     00001111B       FORM HALF ASCII FOR TABLE OFFSET
015.166  075                     4261  SMSB    DCR     A               1 = FIRST ENTRY
015.167  007                     4262          RLC                     *2 FOR TABLE WIDTH
015.170  205                     4263          ADD     L               ADD TO STARTING ADDRESS OF TABLE
015.171  157                     4264          MOV     L,A
015.172  176                     4265          MOV     A,M             GET FUNCTION ADDRESS LSB
015.173  043                     4266          INX     H
015.174  146                     4267          MOV     H,M             MSB
015.175  157                     4268          MOV     L,A
015.176  001 310 100             4269          LXI     B,MODEB         (B,C) = MODEB
015.201  021 307 100             4270          LXI     D,MODEA         (D,E) = MODEA
015.204  351                     4271          PCHL                    GO TO FUNCTION ROUTINE


                                4273  **      SMST - SET MODE SEQUENCE TABLE
                                4274  *
                                4275  *       *SMST* CONTAINS THE ADDRESS OF THE ROUTINE WHICH WILL SET
                                4276  *       THE MODE REQUESTED
                                4277
                                4278
015.205                         4279  SMST    EQU     *
015.205  237 010                 4280          DW      E25L            ENABLE 25TH LINE
015.207  155 013                 4281          DW      NKC             NO KEYBOARD CLICK
015.211  076 011                 4282          DW      EHSM            ENTER HOLD SCREEN MODE
015.213  251 015                 4283          DW      SBC             SET *BLOCK* CURSOR
015.215  176 010                 4284          DW      DC              DISABLE CURSOR
015.217  136 011                 4285          DW      EKSM            ENTER KEYPAD SHIFTED MODE
015.221  113 011                 4286          DW      EKAM            ENTER KEYPAD ALTERNATE MODE
015.223  303 010                 4287          DW      EALF            ENABLE AUTO LINE FEED ON CARRIAGE RETURN
015.225  276 010                 4288          DW      EACR            ENABLE AUTO CARRIAGE RETURN ON LINE FEED
000.015                         4289  .       SET     */256
000.000                         4290          ERRNZ   SMST/256-.      TABLE MUST RESIDE IN ONE PAGE
```

```
                        4292  **      RMST - RESET MODE SEQUENCE TABLE
                        4293  *
                        4294  *       *RMST* CONTAINS THE ADDRESSES OF THE ROUTINES WHICH RESET THE
                        4295  *       MODE SPECIFIED
                        4296
                        4297
015.227                 4298  RMST    EQU     *
015.227  153 010       4299          DW      D25L            DISABLE 25TH LINE
015.231  120 011       4300          DW      EKC             ENABLE KEYBOARD CLICK
015.233  301 016       4301          DW      XHSM            EXIT HOLD SCREEN MODE
015.235  016 016       4302          DW      SUC             SET 'UNDERSCORE' CURSOR
015.237  371 010       4303          DW      EC              ENABLE CURSOR
015.241  316 016       4304          DW      XKSM            EXIT KEYPAD SHIFTED MODE
015.243  311 016       4305          DW      XKAM            EXIT KEYPAD ALTERNATE MODE
015.245  270 016       4306          DW      XALF            EXIT AUTO LINE FEED ON CARRIAGE RETURN
015.247  263 016       4307          DW      XACR            EXIT AUTO CARRIAGE RETURN ON LINE FEED
000.015                 4308  .       SET     */256
000.000                 4309          ERRNZ   RMST/256-.      MUST NOT CROSS PAGE BOUNDARY


                        4311  **      SBC - SET BLOCK CURSOR
                        4312  *
                        4313  *       *SBC* PROGRAMS THE CRTC TO DISPLAY A SLOW BLINKING BLOCK CURSOR
                        4314  *
                        4315  *
                        4316  *       ENTRY   (B,C) = MODEB
                        4317  *               (D,E) = MODEA
                        4318  *
                        4319  *       EXIT    NONE
                        4320  *
                        4321  *       USES    A,F
                        4322
                        4323
015.251  012           4324  SBC     LDAX    B               GET MODE FLAGS
015.252  366 001       4325          ORI     MB.CBLK         FLAG BLOCK CURSOR
015.254  002           4326          STAX    B
015.255  032           4327          LDAX    D
015.256  346 020       4328          ANI     MA.CD           SEE IF CURSOR IS DISABLED
015.260  300           4329          RNZ                     IF DISABLED, EXIT
                        4330
015.261  046 140       4331  SBC.    MVI     H,0+VB.CBE+VB.CBPS      START ON LINE ZERO, BLINK, SLOW
015.263  056 011       4332          MVI     L,9                     STOP ON LINE NINE
015.265  042 277 100   4333          SHLD    VI.CSE
015.270  311           4334          RET
```

```
                                4336  **      SCH - SET CURSOR HOME
                                4337  *
                                4338  *       *SCH* PLACES THE CURSOR AT LINE ZERO, COLUMN ZERO
                                4339  *
                                4340  *
                                4341  *       ENTRY   NONE
                                4342  *
                                4343  *       EXIT    (H,L) = CURSOR ADDRESS
                                4344  *
                                4345  *       USES    A,H,L,F
                                4346
                                4347
015.271  041 000 000           4348  SCH     LXI     H,0             (H,L) = ZERO
015.274  042 272 100           4349          SHLD    CURHP           (CURHP & CURVP) = ZERO
000.000                        4350          ERRNZ   CURVP-CURHP-1
015.277  052 266 100           4351          LHLD    SHOME           RESET CURSOR AND LINE ADDRESSES
015.302  042 274 100           4352          SHLD    CURAD
015.305  042 270 100           4353          SHLD    CLSA
                                4354
                                4355
015.310  311                   4356          RET                     EXIT WITH OR WITHOUT


                                4358  **      AUSCP - ANSI UNSAVE CURSOR POSITION
                                4359  *
                                4360  *       *AUSCP* RETURNS THE CURSOR TO THE POSTION PREVIOUSLY SAVED BY *ASCP*
                                4361  *
                                4362  *
                                4363  *       ENTRY   (B) = ZERO IF NO PN WAS INPUT
                                4364  *
                                4365  *       EXIT    NONE
                                4366  *
                                4367  *       USES    A,B,C,D,E,H,L,F
                                4368
                                4369
015.311  170                   4370  AUSCP   MOV     A,B             SEE IF PN WAS INPUT
015.312  267                   4371          ORA     A
015.313  300                   4372          RNZ                     IF PN INPUT, ILLEGAL, EXIT
                                4373
                                4374  *       JMP     USCP            ELSE, UNSAVE CURSOR POSITION
000.000                        4375          ERRNZ   *-USCP


                                4377  **      USCP - UNSAVE CURSOR POSITION
                                4378  *
                                4379  *       *USCP* RETURNS TO CURHP AND CURVP THE VALUES SAVED WITH THE
                                4380  *       USE OF *SCP*
                                4381  *
                                4382  *
                                4383  *       ENTRY   NONE
                                4384  *
                                4385  *       EXIT    NONE
```

```
                                4386  *
                                4387  *       USES    H,L
                                4388
                                4389
015.314  052 305 100           4390  USCP    LHLD    CSA             GET SAVED ADDRESS
015.317  042 272 100           4391          SHLD    CURHP           UPDATE CURRENT POINTERS
000.000                         4392          ERRNZ   CURVP-CURHP-1   POINTERS ARE CONTIGUOUS
                                4393
                                4394  *       JMP     SNCP            SET NEW CURSOR POSITION
000.000                         4395          ERRNZ   *-SNCP


                                4397  **      SNCP - SET NEW CURSOR POSITION
                                4398  *
                                4399  *       *SNCP* USES THE CURRENT VALUES OF THE VERTICAL AND HORIZONTAL
                                4400  *       POSITION CELLS TO SET THE CURSOR POSITION
                                4401  *
                                4402  *
                                4403  *       ENTRY   *CURVP* = VERTICAL POSITION TO BE SET
                                4404  *               *CURHP* = HORIZONTAL POSITION TO BE SET
                                4405  *
                                4406  *       EXIT    'C' = CLEARED
                                4407  *
                                4408  *       USES    A,D,E,H,L,F
                                4409
                                4410
015.322  072 273 100           4411  SNCP    LDA     CURVP           GET VERTICAL POSITION
015.325  315 137 013           4412          CALL    MPY80           *80
015.330  353                   4413          XCHG                    (D,E) = NUMBER OF CHARACTERS TO NEW LINE
015.331  052 266 100           4414          LHLD    SHOME           GET HOME ADDRESS
015.334  031                   4415          DAD     D               ADD OFFSET FOR THIS LINES STARTING ADDRESS
015.335  174                   4416          MOV     A,H             STAY IN VIDEO RAM
015.336  366 370               4417          ORI     VRAMS/256
015.340  147                   4418          MOV     H,A
015.341  042 270 100           4419          SHLD    CLSA            SET CURRENT LINE STARTING ADDRESS
                                4420
015.344  072 272 100           4421          LDA     CURHP           GET HORIZONTAL POSITION
015.347  137                   4422          MOV     E,A             ADD TO LINE STARTING ADDRESS
015.350  026 000               4423          MVI     D,0
015.352  031                   4424          DAD     D
015.353  174                   4425          MOV     A,H             STAY IN VIDEO RAM
015.354  366 370               4426          ORI     VRAMS/256
015.356  147                   4427          MOV     H,A
015.357  042 274 100           4428          SHLD    CURAD           SET CURSOR ADDRESS
015.362  311                   4429          RET
```

```
                        4432  **      STAB - SEARCH TABLE
                        4433  *
                        4434  *       *STAB* SEARCHES TABLES OF SPECIFIED LENGTH AND WIDTH FOR
                        4435  *       A MATCH WITH THE FIRST CHARACTER IN AN ENTRY
                        4436  *
                        4437  *
                        4438  *       ENTRY   (A) = CHARACTER TO MATCH
                        4439  *               (D) = TABLE LENGTH
                        4440  *               (E) = TABLE WIDTH
                        4441  *               (H,L) = TABLE ADDRESS
                        4442  *
                        4443  *       EXIT    (A) = FIRST CHARACTER IN TABLE AFTER MATCH OR
                        4444  *                     ORIGINAL CHARACTER IF NO MATCH
                        4445  *               (H,L) = FIRST ADDRESS AFTER CHARACTER MATCHED
                        4446  *               'C' = SET IF NO MATCH
                        4447  *
                        4448  *       USES    A,B,C,D,E,H,L,F
                        4449
                        4450
015.363  117           4451  STAB    MOV     C,A             SAVE CHARACTER TO MATCH
015.364  256           4452  STAB1   XRA     M               CHARACTER MATCH TABLE ENTRY?
015.365  050 013       4453          JR      Z,STAB3         IF CHARACTER MATCHED
                        4454
015.367  103           4455          MOV     B,E             ELSE, GET WIDTH AND ADVANCE TO NEXT ENTRY
015.370  171           4456          MOV     A,C             GET CHARACTER TO MATCH
015.371  043           4457  STAB2   INX     H               ADVANCE (E) BYTES
015.372  020 375       4458          DJNZ    STAB2
                        4459
015.374  025           4460          DCR     D               DECREMENT LENGTH COUNTER
015.375  040 365       4461          JR      NZ,STAB1        IF NOT LAST ENTRY
                        4462
015.377  171           4463          MOV     A,C             ELSE, RESTORE ORIGINAL CHARACTER AND EXIT
016.000  067           4464          STC                     SET 'C'
016.001  311           4465          RET
                        4466
016.002  043           4467  STAB3   INX     H               MATCH FOUND, GET NEXT BYTE
016.003  176           4468          MOV     A,M
016.004  311           4469          RET



                        4471  **      SPWE - SET PREVIOUS WAS AN ESCAPE FLAG
                        4472  *
                        4473  *
                        4474  *       ENTRY   NONE
                        4475  *
                        4476  *       EXIT    NONE
                        4477  *
                        4478  *       USES    A,F
                        4479
                        4480
016.005  072 311 100   4481  SPWE    LDA     MODEI           GET INTERNAL MODE FLAGS
016.010  366 001       4482          ORI     MI.PWE          SET PWE
016.012  062 311 100   4483          STA     MODEI           UPDATE FLAGS
016.015  311           4484          RET
```

```
                        4486  **      SUC - SET UNDERSCORE CURSOR
                        4487  *
                        4488  *       *SUC* PROGRAMS THE CRTC FOR A SINGLE SCAN LINE CURSOR ON SCAN
                        4489  *       LINE EIGHT OF THE CHARACTER ROW.
                        4490  *
                        4491  *
                        4492  *       ENTRY   (B,C) = MODEB
                        4493  *               (D,E) = MODEA
                        4494  *
                        4495  *       EXIT    NONE
                        4496  *
                        4497  *       USES    A,B,C
                        4498
                        4499
016.016  012            4500  SUC     LDAX    B               GET MODE FLAGS
016.017  346 376        4501          ANI     255-MB,CBLK     RESET CURSOR = BLOCK FLAG
016.021  002            4502          STAX    B
016.022  032            4503          LDAX    D
016.023  346 020        4504          ANI     MA.CD           SEE IF CURSOR IS DISABLED
016.025  300            4505          RNZ                     IF DISABLED, EXIT
                        4506
016.026  046 110        4507  SUC.    MVI     H,8+VB.CBE      START ON LINE 8, BLINK FAST
016.030  056 010        4508          MVI     L,8             ALSO END ON LINE 8
016.032  042 277 100    4509          SHLD    VI.CSE
016.035  311            4510          RET


                        4512  **      TAB - TAB CURSOR TO NEXT EIGTH COLUMN
                        4513  *
                        4514  *       *TAB* PLACES THE CURSOR AT THE NEXT MULTIPLE OF EIGHT COLUMNS
                        4515  *       FROM THE BEGINNING OF THE LINE UNLESS THE CURSOR IS WITHIN SEVEN
                        4516  *       COLUMNS OF THE END, IN WHICH CASE THE CURSOR IS MOVED ONLY ONE COLUMN.
                        4517  *       THE CURSOR IS NOT WRAPPED TO THE NEXT LINE
                        4518  *
                        4519  *
                        4520  *       ENTRY   NONE
                        4521  *
                        4522  *       EXIT    NONE
                        4523  *
                        4524  *       USES    A,D,E,H,L,F
                        4525
                        4526
016.036  072 272 100    4527  TAB     LDA     CURHP           GET CURRENT HORIZONTAL POSITION
016.041  306 010        4528          ADI     8               ADD EIGHT COLUMNS
016.043  346 370        4529          ANI     377Q-7          TOSS ANY LEFT OVERS LESS THAN EIGHT
016.045  376 120        4530          CPI     80              PAST END OF LINE?
016.047  040 016        4531          JR      NZ,TAB2         IF STILL ON SAME LINE
                        4532
016.051  072 272 100    4533          LDA     CURHP           GET CURRENT POSITION ON LINE
016.054  376 117        4534          CPI     79              AT LAST COLUMN YET?
                        4535
016.056  050 003        4536          JR      Z,TAB1,5        IF AT LAST COLUMN, STAY
                        4537
016.060  074            4538  TAB1    INR     A               ELSE, MOVE ONE COLUMN CLOSER
```

```
016.061  030 004      4539          JR      TAB2            UPDATE CURSOR TO NEW POSITION
                      4540
016.063  076 117      4541  TAB1.5  MVI     A,79            ELSE, STOP AT LAST COLUMN OF LAST LINE
016.065  030 000      4542          JR      TAB2            UPDATE CURSOR ADDRESS
                      4543
016.067  062 272 100  4544  TAB2    STA     CURHP           UPDATE COLUMN COUNTER
016.072  137          4545          MOV     E,A             ADD TO LINE ADDRESS
016.073  026 000      4546          MVI     D,0
016.075  052 270 100  4547          LHLD    CLSA            GET CURRENT LINE ADDRESS
016.100  031          4548          DAD     D
016.101  174          4549          MOV     A,H             STAY IN VIDEO RAM
016.102  366 370      4550          ORI     VRAMS/256
016.104  147          4551          MOV     H,A
016.105  042 274 100  4552          SHLD    CURAD           UPDATE CURSOR ADDRESS
                      4553
                      4554
016.110  311          4555          RET                     EXIT


                      4557  **      UCP - UPDATE CURSOR POSITION
                      4558  *
                      4559  *       *UCP* SENDS THE LOWER ELEVEN BITS OF THE CURSOR ADDRESS TO THE CRTC
                      4560  *
                      4561  *
                      4562  *       ENTRY   NONE
                      4563  *
                      4564  *       EXIT    NONE
                      4565  *
                      4566  *       USES    A,H,L,F
                      4567
                      4568
016.111  072 273 100  4569  UCP     LDA     CURVP           GET LINE COUNT
016.114  315 137 013  4570          CALL    MPY80           MULTIPLY BY 80 CHARACTERS PER LINE
016.117  072 272 100  4571          LDA     CURHP           GET COLUMN COUNT
016.122  137          4572          MOV     E,A             ADD TO LINE COUNT
016.123  026 000      4573          MVI     D,0
016.125  031          4574          DAD     D
016.126  353          4575          XCHG                    (D,E) = CHARACTER OFFSET FROM HOME POSITION
016.127  052 266 100  4576          LHLD    SHOME           GET HOME POSITION
016.132  174          4577          MOV     A,H             MASK FOR A 2K ADDRESS
016.133  346 007      4578          ANI     HOMAX/256
016.135  147          4579          MOV     H,A
016.136  031          4580          DAD     D               ADD OFFSET
016.137  174          4581          MOV     A,H             GET CURAD MSB
016.140  346 077      4582          ANI     CURMAX/256      TOSS UPPER 5 BITS
016.142  147          4583          MOV     H,A
016.143  042 303 100  4584          SHLD    VI.CA
                      4585
                      4586  *       HIT NMI TO MAKE SURE ITS RUNNING
                      4587  *
016.146  076 000      4588  UCP.    MVI     A,0             SET REGISTER ZERO AS DUMMY
016.150  323 144      4589          OUT     VP,AR+VB,NMI    OUT NMI REQUEST WITH PORT NUMBER
016.152  311          4590          RET
```

```
                        4592  **      WEOL - WRAP AROUND AT END OF LINE
                        4593  *
                        4594  *       *WEOL* SETS THE FLAG WHICH CAUSES THE TERMINAL TO PERFORM
                        4595  *       A CARRIAGE RETURN AND LINE FEED WHEN THE 81ST CHARACTER FOR A
                        4596  *       LINE IS RECEIVED
                        4597  *
                        4598  *
                        4599  *       ENTRY   (B,C) = MODEB
                        4600  *
                        4601  *       EXIT    NONE
                        4602  *
                        4603  *       USES    A,F
                        4604
                        4605
016.153  012            4606  WEOL    LDAX    B               GET MODE FLAGS
016.154  366 004        4607          ORI     MB.WRAP         SET WRAP AROUND FLAG
016.156  002            4608          STAX    B
016.157  311            4609          RET


                        4611  **      WSV - WRITE SPACES TO VIDEO
                        4612  *
                        4613  *       *WSV* WRITES N ASCII SPACE CODES INTO THE VIDEO RAM.  THE LAST SPACE
                        4614  *       WRITTEN MUST BE AT THE END OF A VIDEO LINE (X*80+VRAMS)
                        4615  *
                        4616  *
                        4617  *       ENTRY   (D,E) = NUMBER OF SPACES TO BE WRITTEN
                        4618  *               (H,L) = ADDRESS OF FIRST SPACE TO BE WRITTEN
                        4619  *
                        4620  *       EXIT    NONE
                        4621  *
                        4622  *       USES    A,B,C,D,E,H,L,F
                        4623
                        4624
016.160                 4625  WSV     EQU     *
                        4626
                        4627
016.160  175            4628  WSV0.5  MOV     A,L             GET LSB OF ADDRESS
016.161  346 017        4629          ANI     00001111B       MASK FOR ANY CARRY PAST MOD 16
016.163  050 007        4630          JR      Z,WSV2          IF NO EXTRAS
                        4631
016.165  016 040        4632          MVI     C,' '           SPACE = FILL CHARACTER
016.167  161            4633          MOV     M,C             PLACE SPACE IN MEMORY
016.170  043            4634          INX     H               POINT TO NEXT
016.171  033            4635          DCX     D               DECREMENT COUNT
016.172  030 364        4636          JR      WSV0.5
                        4637
016.174  006 004        4638  WSV2    MVI     B,4             ROTATE COUNT TO A MOD 16 COUNTER
016.176  172            4639  WSV3    MOV     A,D             ROTATE MSB
016.177  037            4640          RAR
016.200  127            4641          MOV     D,A
016.201  173            4642          MOV     A,E             ROTATE LSB
016.202  037            4643          RAR
016.203  137            4644          MOV     E,A
```

```
016.204  020 370      4645           DJNZ    WSV3            LOOP 4 TIMES
                       4646
016.206  173          4647           MOV     A,E             CHECK FOR ZERO COUNT
016.207  267          4648           ORA     A
016.210  310          4649           RZ                      IF NO SPACES LEFT TO BE WRITTEN
                       4650
016.211  103          4651           MOV     B,E             PLACE MOD 16 COUNT IN B
016.212                4652  WSVA     EQU     *               ALTERNATE ENTRY POINT FROM *PLF*
                       4653
                       4654
016.212  016 040      4655  WSVA.    MVI     C,' '           (C) = ASCII SPACE
016.214  174          4656  WSV4     MOV     A,H             STAY IN VIDEO RAM
016.215  366 370      4657           ORI     VRAMS/256
016.217  147          4658           MOV     H,A
016.220  161          4659           MOV     M,C             PLACE 16 SPACES IN MEMORY
016.221  043          4660           INX     H
016.222  161          4661           MOV     M,C
016.223  043          4662           INX     H
016.224  161          4663           MOV     M,C
016.225  043          4664           INX     H
016.226  161          4665           MOV     M,C
016.227  043          4666           INX     H
016.230  161          4667           MOV     M,C
016.231  043          4668           INX     H
016.232  161          4669           MOV     M,C
016.233  043          4670           INX     H
016.234  161          4671           MOV     M,C
016.235  043          4672           INX     H
016.236  161          4673           MOV     M,C
016.237  043          4674           INX     H
016.240  161          4675           MOV     M,C
016.241  043          4676           INX     H
016.242  161          4677           MOV     M,C
016.243  043          4678           INX     H
016.244  161          4679           MOV     M,C
016.245  043          4680           INX     H
016.246  161          4681           MOV     M,C
016.247  043          4682           INX     H
016.250  161          4683           MOV     M,C
016.251  043          4684           INX     H
016.252  161          4685           MOV     M,C
016.253  043          4686           INX     H
016.254  161          4687           MOV     M,C
016.255  043          4688           INX     H
016.256  161          4689           MOV     M,C
016.257  043          4690           INX     H
016.260  020 332      4691           DJNZ    WSV4            LOOP UNTIL MOD 16 COUNTER = ZERO
                       4692
016.262  311          4693           RET
                       4694
```

```
                        4696  **      XACR - EXIT AUTO CARRIAGE RETURN
                        4697  *
                        4698  *       *XACR* CLEARS THE AUTO CARRIAGE RETURN FLAG
                        4699  *
                        4700  *
                        4701  *       ENTRY   (B,C) = MODEB
                        4702  *
                        4703  *       EXIT    NONE
                        4704  *
                        4705  *       USES    A,F
                        4706
                        4707
016.263  012           4708  XACR    LDAX    B               GET MODE FLAGS
016.264  346 357       4709          ANI     255-MB.ACR      CLEAR AUTO CARRIAGE RETURN
016.266  002           4710          STAX    B
016.267  311           4711          RET


                        4713  **      XALF - EXIT AUTO LINE FEED MODE
                        4714  *
                        4715  *       *XALF* CLEARS THE AUTO LINE FEED MODE FLAG
                        4716  *
                        4717  *
                        4718  *       ENTRY   (B,C) = MODEB
                        4719  *
                        4720  *       EXIT    NONE
                        4721  *
                        4722  *       USES    A,F
                        4723
                        4724
016.270  012           4725  XALF    LDAX    B               GET MODE FLAGS
016.271  346 367       4726          ANI     255-MB.ALF
016.273  002           4727          STAX    B
016.274  311           4728          RET


                        4730  **      XGM - EXIT GRAPHICS MODE
                        4731  *
                        4732  *       *EGM* CLEARS THE GRAPHICS MODE FLAG
                        4733  *
                        4734  *       ENTRY   (D,E) = MODEA
                        4735  *
                        4736  *       EXIT    NONE
                        4737  *
                        4738  *       USES    A,F
                        4739
                        4740
016.275  353           4741  XGM     XCHG                    (H,L) = MODEA
                        4742  *       RES     IB.GRPH,(HL)    RESET GRAPHICS MODE FLAG
016.276  313 216       4743          DB      I,RESHA,I,RESHB!IB.GRPH
016.300  311           4744          RET
```

```
                        4746  **      XHSM - EXIT HOLD SCREEN MODE
                        4747  *
                        4748  *       *XHSM* CLEARS THE HOLD SCREEN MODE FLAG
                        4749  *
                        4750  *
                        4751  *       ENTRY   (D,E) = MODEA
                        4752  *
                        4753  *       EXIT    NONE
                        4754  *
                        4755  *       USES    A,F
                        4756
                        4757
016.301  353           4758  XHSM    XCHG                    (H,L) = MODEA
                        4759  *       RES     IB.HSM,(H,L)
016.302  313 206       4760          DB      I.RESHA,I.RESHB!IB.HSM
016.304  311           4761          RET


                        4763  **      XICM - EXIT INSERT CHARACTER MODE
                        4764  *
                        4765  *       *XICM* CLEARS THE INSERT CHARACTER MODE FLAG
                        4766  *
                        4767  *
                        4768  *       ENTRY   (D,E) = MODEA
                        4769  *
                        4770  *       EXIT    NONE
                        4771  *
                        4772  *       USES    A,F
                        4773
                        4774
016.305  353           4775  XICM    XCHG                    (H,L) = MODEA
                        4776  *       RES     IB.ICM,(HL)     RESET INSERT CHARACTER MODE FLAG
016.306  313 266       4777          DB      I.RESHA,I.RESHB!IB.ICM
016.310  311           4778          RET


                        4780  **      XKAM - EXIT KEYPAD ALTERNATE MODE
                        4781  *
                        4782  *       *XKAM* CLEARS THE KEYPAD ALTERNATE MODE FLAG
                        4783  *
                        4784  *
                        4785  *       ENTRY   (B,C) = MODEB
                        4786  *
                        4787  *       EXIT    NONE
                        4788  *
                        4789  *       USES    A,F
                        4790
                        4791
016.311  012           4792  XKAM    LDAX    B               GET MODEB FLAGS
016.312  346 177       4793          ANI     377Q-MB.KPDA
016.314  002           4794          STAX    B
016.315  311           4795          RET
```

```
                        4797  **      XKSM - EXIT KEYPAD SHIFTED MODE
                        4798  *
                        4799  *       *XKSM* CLEARS THE KEYPAD SHIFTED MODE FLAG
                        4800  *
                        4801  *
                        4802  *       ENTRY   (B,C) = MODEB
                        4803  *
                        4804  *       EXIT    NONE
                        4805  *
                        4806  *       USES    A,F
                        4807
                        4808
016.316  012           4809  XKSM    LDAX    B               GET MODEB FLAGS
016.317  346 277       4810          ANI     377Q-MB.KPDS
016.321  002           4811          STAX    B
016.322  311           4812          RET


                        4814  **      AXMT25 - ANSI TRANSMIT 25TH LINE
                        4815  *
                        4816  *       *AXMT25* IS THE ENTRY POINT FOR *XMT25* WHEN THE TERMINAL IS
                        4817  *       IN THE ANSI MODE
                        4818  *
                        4819  *
                        4820  *       ENTRY   (B) = ZERO IF NO PN WAS INPUT
                        4821  *
                        4822  *       EXIT    TO *XMT25*
                        4823  *
                        4824  *       USES    A,B,C,D,E,H,L,F
                        4825
                        4826
016.323  170           4827  AXMT25  MOV     A,B             SEE IF PN WAS INPUT
016.324  267           4828          ORA     A
016.325  300           4829          RNZ                     IF INPUT, ILLEGAL, EXIT
                        4830
                        4831  *       JMP     XMT25           ELSE, TRANSMIT PAGE
000.000                 4832          ERRNZ   *-XMT25


                        4834  **      XMT25 - TRANSMIT 25TH LINE
                        4835  *
                        4836  *       *XMT25* TRANSMITS THE LINE IN THE SAME MANNER AS *XMTP*, BUT ONLY IF
                        4837  *       THE 25TH LINE IS ENABLED.
                        4838  *
                        4839  *
                        4840  *       ENTRY   NONE
                        4841  *
                        4842  *       EXIT    NONE
                        4843  *
                        4844  *       USES    A,B,C,D,E,H,L,F
                        4845
                        4846
```

```
016.326  072 311 100   4847  XMT25   LDA     MODEI           GET MODE FLAGS
016.331  346 200       4848          ANI     MI.25L          IS 25TH LINE ON?
016.333  050 021       4849          JR      Z,XMT25.1       IF NOT ON, JUST SEND CR
                       4850
016.335  052 266 100   4851          LHLD    SHOME           GET CURRENT HOME POSITION
016.340  021 200 007   4852          LXI     D,1920          ADD 24 LINES*80 COLUMNS
016.343  031           4853          DAD     D
016.344  174           4854          MOV     A,H             STAY IN VIDEO RAM
016.345  366 370       4855          ORI     VRAMS/256
016.347  147           4856          MOV     H,A
016.350  353           4857          XCHG                    FIRST ADDRESS ON LINE TO (D,E)
016.351  006 000       4858          MVI     B,0             CURRENT TRANSMIT MODE = NONE
016.353  315 001 017   4859          CALL    XMTL            TRANSMIT LINE
016.356  076 015       4860  XMT25.1 MVI     A,CR            TERMINATE LINE WITH A CARRIAGE RETURN
016.360  315 366 016   4861          CALL    XMTC
016.363  323 340       4862          OUT     MP.BELL         SIGNAL DONE TO USER
016.365  311           4863          RET


                       4865  **      XMTC - TRANSMIT CHARACTER
                       4866  *
                       4867  *       *XMTC* PLACES THE GIVEN CHARACTER INTO THE OUTPUT FIFO AND
                       4868  *       ASSURES THAT THE TERMINAL IS IN THE ON-LINE STATE UNTIL THE CHARACTER
                       4869  *       IS SENT
                       4870  *
                       4871  *
                       4872  *               **********     CAUTION! THIS ROUTINE MAY       **********
                       4873  *               **********  ONLY BE USED BY  *XMT??* ROUTINES  **********
                       4874  *
                       4875  *
                       4876  *
                       4877  *       ENTRY   (A) = CHARACTER TO SEND
                       4878  *
                       4879  *       EXIT    NONE
                       4880  *
                       4881  *       USES    A,D,E,H,L,F
                       4882
                       4883
016.366  365           4884  XMTC    PUSH    PSW             SAVE CHARACTER TO SEND
                       4885
016.367  333 105       4886  XMTC1   IN      AP.LSR          GET LINE STATUS REGISTER OF UART
016.371  346 040       4887          ANI     AB.THRE         HOLDING REGISTER EMPTY?
016.373  050 372       4888          JR      Z,XMTC1         IF NOT EMPTY
                       4889
016.375  361           4890          POP     PSW             ELSE, OUTPUT CHARACTER
016.376  323 100       4891          OUT     AP.THR
017.000  311           4892          RET
                       4893
```

```
                                4895  **      XMTL - TRANSMIT LINE
                                4896  *
                                4897  *       *XMTL* TRANSMITS ONE LINE TO THE HOST INCLUDING THE APPROPRIATE
                                4898  *       ESCAPE SEQUENCES FOR ENTERING AND EXITING THE GRAPHICS AND
                                4899  *       REVERSE VIDEO MODES
                                4900  *
                                4901  *
                                4902  *       ENTRY   (B) = CURRENT TRANSMIT MODE (RV AND/OR GRAPHICS)
                                4903  *               (D,E) = ADDRESS OF BEGINNING OF LINE
                                4904  *
                                4905  *       EXIT    (B) = CURRENT TRANSMIT MODE (UPDATED)
                                4906  *               (D,E) = LAST ADDRESS OF LINE +1
                                4907  *
                                4908  *       USES    A,B,C,D,E,H,L,F
                                4909
                                4910
017.001  046 120                4911  XMTL    MVI     H,80            SET NUMBER OF CHARACTERS TO SEND
017.003  345                    4912  XMTL1.1 PUSH    H
017.004  032                    4913          LDAX    D               GET CHARACTER
017.005  117                    4914          MOV     C,A             SAVE FOR LATER USE
017.006  023                    4915          INX     D               INCREMENT MEMORY POINTER
017.007  172                    4916          MOV     A,D             STAY IN VIDEO RAM
017.010  366 370                4917          ORI     VRAMS/256
017.012  127                    4918          MOV     D,A
017.013  325                    4919          PUSH    D               SAVE ADDRESS
017.014  171                    4920          MOV     A,C             GET CHARACTER TO SEND
017.015  346 177                4921          ANI     01111111B       TOSS ANY REVERSE VIDEO BIT FOR NOW
017.017  376 040                4922          CPI     ' '             SEE IF LESS THAN A SPACE (GRAPHICS)
017.021  070 010                4923          JR      C,XMTL1         IF A GRAPHIC CHARACTER
                                4924
017.023  376 177                4925          CPI     177Q            177Q IS ALSO A GRAPHIC DISPLAY CHARACTER
017.025  040 055                4926          JR      NZ,XMTL4        IF NOT GRAPHIC
                                4927
                                4928  *       CHARACTER IS GRAPHIC
                                4929  *
017.027  076 136                4930          MVI     A,'^'           SET PRINTABLE EQUIVALENT OF 177Q
017.031  030 012                4931          JR      XMTL2
                                4932
017.033  376 037                4933  XMTL1   CPI     37Q             SEE IF GRAPHIC '_'
017.035  040 004                4934          JR      NZ,XMTL1.3      IF NOT 37Q
                                4935
017.037  076 137                4936          MVI     A,'_'           ELSE, SET PRINTABLE
017.041  030 002                4937          JR      XMTL2           CHECK GRAPHIC MODE
                                4938
017.043  366 140                4939  XMTL1.3 ORI     01100000B       MAKE GRAPHIC PRINTABLE
017.045  365                    4940  XMTL2   PUSH    PSW             SAVE CHARACTER TO OUTPUT
                                4941
                                4942  *       SEE IF LAST CHARACTER OUTPUT WAS ALSO GRAPHIC
                                4943  *
                                4944  *       BIT     1,B
017.046  313 110                4945          DB      I.BITA,I.BITB!IB.XMTG!IR.B
017.050  040 067                4946          JR      NZ,XMTL6        IF LAST CHARACTER WAS ALSO GRAPHIC
                                4947
                                4948  *       SET     1,B             ELSE, SET MODE TO INCLUDE GRAPHICS
017.052  313 310                4949          DB      I.SETA,I.SETB!IB.XMTG!IR.B
017.054  072 310 100            4950          LDA     MODEB           SEE IF IN ANSI MODE
```

```
017.057  346 040        4951           ANI     MB.ANSI
017.061  050 012        4952           JR      Z,XMTL3          IF IN HEATH MODE
                        4953
017.063  315 302 017    4954           CALL    XMTS             TRANSMIT ANSI 'ENTER GRAPHICS' SEQUENCE
017.066  033 133 061    4955           DB      ESC,'[','10','m'+200Q
                        4956
017.073  030 044        4957           JR      XMTL6            GO CHECK FOR REVERSE VIDEO
                        4958
017.075  315 302 017    4959  XMTL3    CALL    XMTS             TRANSMIT HEATH 'ENTER GRAPHICS' SEQUENCE
017.100  033 306        4960           DB      ESC,'F'+200Q
017.102  030 035        4961           JR      XMTL6            CHECK FOR REVERSE VIDEO
                        4962
                        4963  *        CHARACTER IS NOT GRAPHIC
                        4964  *
017.104  365            4965  XMTL4    PUSH    PSW              SAVE CHARACTER TO OUTPUT
                        4966  *        BIT     1,B              SEE IF LAST WAS GRAPHIC
017.105  313 110        4967           DB      I.BITA,I.BITB!IB.XMTG!IR.B
017.107  050 030        4968           JR      Z,XMTL6          IF LAST WAS NOT GRAPHIC EITHER
                        4969
                        4970  *        RES     1,B              ELSE, SET NO GRAPHIC
017.111  313 210        4971           DB      I.RESA,I.RESB!IB.XMTG!IR.B
                        4972
017.113  072 310 100    4973           LDA     MODEB            SEE IF IN ANSI MODE
017.116  346 040        4974           ANI     MB.ANSI
017.120  050 012        4975           JR      Z,XMTL5          IF IN HEATH MODE
                        4976
017.122  315 302 017    4977           CALL    XMTS             ELSE, TRANSMIT ANSI 'EXIT GRAPHICS' SEQUENCE
017.125  033 133 061    4978           DB      ESC,'[','11','m'+200Q
017.132  030 005        4979           JR      XMTL6            CHECK FOR REVERSE VIDEO
                        4980
017.134  315 302 017    4981  XMTL5    CALL    XMTS             TRANSMIT HEATH 'EXIT GRAPHICS' SEQUENCE
017.137  033 307        4982           DB      ESC,'G'+200Q
                        4983
                        4984  *        SEE IF CHARACTER IS IN REVERSE VIDEO
                        4985  *
017.141                 4986  XMTL6    EQU     *
                        4987
                        4988  *        BIT     7,C              SEE IF MSB IS SET FOR RV
017.141  313 171        4989           DB      I.BITA,I.BITB!IB.RV!IR.C
017.143  050 035        4990           JR      Z,XMTL7.5        IF NOT RV
                        4991
                        4992  *        CHARACTER IS REVERSE VIDEO
                        4993  *
                        4994  *        BIT     2,B              SEE IF LAST WAS RV
017.145  313 120        4995           DB      I.BITA,I.BITB!IB.XMTR!IR.B
017.147  040 063        4996           JR      NZ,XMTL9         IF LAST WAS ALSO RV
                        4997
                        4998  *        SET     2,B              ELSE, SET RV MODE AS CURRENT
017.151  313 320        4999           DB      I.SETA,I.SETB!IB.XMTR!IR.B
017.153  072 310 100    5000           LDA     MODEB            SEE IF IN ANSI MODE
017.156  346 040        5001           ANI     MB.ANSI
017.160  050 011        5002           JR      Z,XMTL7          IF IN HEATH MODE
                        5003
017.162  315 302 017    5004           CALL    XMTS             ELSE, TRANSMIT ANSI 'ENTER REVERSE VIDEO'
017.165  033 133 067    5005           DB      ESC,'[','7','m'+200Q
017.171  030 041        5006           JR      XMTL9            GO TRANSMIT CHARACTER
```

```
                                5007
017.173  315 302 017           5008  XMTL7   CALL    XMTS            TRANSMIT HEATH 'ENTER REVERSE VIDEO'
017.176  033 360               5009          DB      ESC,'p'+200Q
017.200  030 032               5010          JR      XMTL9           OUTPUT CHARACTER
                                5011
                                5012  *       CHARACTER IS NOT REVERSE VIDEO
                                5013  *
017.202                         5014  XMTL7.5 EQU     *
                                5015  *       BIT     2,B             SEE IF LAST CHARACTER WAS RV
017.202  313 120               5016          DB      I.BITA,I.BITB!IB.XMTR!IR.B
017.204  050 026               5017          JR      Z,XMTL9         IF LAST WASN'T EITHER
                                5018
                                5019  *       RES     2,B             ELSE, SET CURRENT MODE AS NOT REVERSE
017.206  313 220               5020          DB      I.RESA,I.RESB!IB.XMTR!IR.B
                                5021
017.210  072 310 100           5022          LDA     MODEB           SEE IF IN ANSI MODE
017.213  346 040               5023          ANI     MB.ANSI
017.215  050 010               5024          JR      Z,XMTL8         IF IN HEATH MODE
                                5025
017.217  315 302 017           5026          CALL    XMTS            ELSE, SEND ANSI 'EXIT RV'
017.222  033 133 355           5027          DB      ESC,'[','m'+200Q
017.225  030 005               5028          JR      XMTL9           SEND CHARACTER
                                5029
017.227  315 302 017           5030  XMTL8   CALL    XMTS            SEND HEATH 'EXIT RV' SEQUENCE
017.232  033 361               5031          DB      ESC,'a'+200Q
                                5032
017.234  361                   5033  XMTL9   POP     PSW             GET CHARACTER TO SEND
017.235  315 366 016           5034          CALL    XMTC            SEND IT
017.240  321                   5035          POP     D               GET LINE ADDRESS
017.241  341                   5036          POP     H               GET COLUMN COUNTER
017.242  045                   5037          DCR     H               SEE IF ANY LEFT TO SEND
017.243  302 003 017           5038          JNZ     XMTL1,1         IF NOT DONE YET
                                5039
017.246  311                   5040          RET
```

```
                                5042  **      AXMTP - ANSI TRANSMIT PAGE
                                5043  *
                                5044  *       *AXMTP* IS THE ENTRY POINT FOR *XMTP* WHEN THE TERMINAL IS IN
                                5045  *       THE ANSI MODE
                                5046  *
                                5047  *
                                5048  *       ENTRY   (B) = ZERO IF NO PN WAS INPUT
                                5049  *
                                5050  *       EXIT    TO *XMTP*
                                5051  *
                                5052  *       USES    A,B,C,D,E,H,L,F
                                5053
                                5054
017.247  170                   5055  AXMTP   MOV     A,B             SEE IF PN WAS INPUT
017.250  267                   5056          ORA     A
017.251  300                   5057          RNZ                     IF INPUT, ILLEGAL, EXIT
                                5058
                                5059  *       JMP     XMTP            ELSE, TRANSMIT PAGE
```

```
000.000                  5060          ERRNZ   *-XMTP


                         5062  **      XMTP - TRANSMIT PAGE
                         5063  *
                         5064  *       *XMTP* TRANSMITS THE ENTIRE CONTENTS OF THE PAGE (EXCLUDING 25TH LINE)
                         5065  *       IF THE TERMINAL IS OFF LINE, THE CONTENTS OF THE PAGE IS STILL
                         5066  *       TRANSMITTED TO THE HOST AND THE TERMINAL IS RETURNED TO THE CURRENT
                         5067  *       STATE OF THE OFF LINE SWITCH AFTER THE PAGE HAS BEEN TRANSMITTED.
                         5068  *       IF GRAPHIC CHARACTERS OR REVERSE VIDEO CHARACTERS ARE ENCOUNTERED,
                         5069  *       THE PROPER ESCAPE CODES FOR ENTERING AND EXITING THESE MODES ARE SENT
                         5070  *       TO THE HOST IN THE SAME MANNER AS THE HOST WOULD USE TO PLACE THESE
                         5071  *       CHARACTERS ON THE SCREEN
                         5072  *
                         5073  *
                         5074  *       ENTRY   NONE
                         5075  *
                         5076  *       EXIT    NONE
                         5077  *
                         5078  *       USES    A,B,C,D,E,H,L,F
                         5079
                         5080
017.252  052 266 100    5081  XMTP    LHLD    SHOME           GET FIRST ADDRESS ON SCREEN
017.255  353            5082          XCHG                    TO (D,E)
017.256  006 000        5083          MVI     B,0             CLEAR CURRENT TRANSMIT MODE
017.260  046 030        5084          MVI     H,24            SEND 24 LINES
017.262  345            5085  XMTP1   PUSH    H               SAVE COUNT
017.263  315 001 017    5086          CALL    XMTL            TRANSMIT ONE LINE
017.266  341            5087          POP     H               GET LINE COUNTER
017.267  045            5088          DCR     H               COUNT - 1
017.270  040 370        5089          JR      NZ,XMTP1        IF NOT DONE
                         5090
017.272  076 015        5091          MVI     A,CR            SEND A CARRIAGE RETURN TO TERMINATE PAGE
017.274  315 366 016    5092          CALL    XMTC            TRANSMIT CHARACTER
017.277  323 340        5093          OUT     MP.BELL         SIGNAL DONE TO USER
017.301  311            5094          RET


                         5096  **      XMTS - TRANSMIT STRING
                         5097  *
                         5098  *       *XMTS* PLACES A STRING OF CHARACTERS IN THE OUTPUT FIFO AND
                         5099  *       ASSURES THAT THEY ARE SENT REGARDLESS OF WHETHER THE TERMINAL
                         5100  *       IS ON LINE OR NOT
                         5101  *
                         5102  *
                         5103  *       ENTRY   STRING TO BE SENT MUST IMMEDIATELY FOLLOW THE CALL
                         5104  *                       TO THIS ROUTINE.  FINAL CHARACTER MUST HAVE
                         5105  *                       BIT SEVEN SET
                         5106  *
                         5107  *       EXIT    NONE
                         5108  *
                         5109  *       USES    A,D,E,H,L,F
```

```
                          5110
                          5111
017.302  321             5112  XMTS    POP     D               GET ADDRESS OF CHARACTERS TO BE SENT
017.303  032             5113  XMTS1   LDAX    D               GET CHARACTER TO SEND
017.304  023             5114          INX     D               POINT TO NEXT
017.305  267             5115          ORA     A               SET CPU FLAGS
017.306  372 316 017      5116          JM      XMTS2           IF LAST CHARACTER TO SEND
                          5117
017.311  315 366 016      5118          CALL    XMTC            SEND ONE CHARACTER
017.314  030 365          5119          JR      XMTS1           GET NEXT
                          5120
017.316  346 177          5121  XMTS2   ANI     01111111B       TOSS TERMINATOR BIT
017.320  315 366 016      5122          CALL    XMTC            OUTPUT LAST CHARACTER
017.323  325             5123          PUSH    D               SET RETURN ADDRESS
017.324  311             5124          RET


                          5126  **      XRVM - EXIT REVERSE VIDEO MODE
                          5127  *
                          5128  *       *XRVM* CLEARS THE REVERSE VIDEO MODE FLAG
                          5129  *
                          5130  *
                          5131  *       ENTRY   (D,E) = MODEA
                          5132  *
                          5133  *       EXIT    NONE
                          5134  *
                          5135  *       USES    A,F
                          5136
                          5137
017.325                   5138  XRVM    EQU     *
                          5139
                          5140
017.325  353             5141          XCHG                    (H,L) = MODEA
                          5142  *       RES     IB.RV,(HL)      RESET FLAG
017.326  313 276          5143          DB      I.RESHA,I.RESHB!IB.RV
017.330  311             5144          RET
```

```
                        5147  **      VPARD60 - VIDEO PARAMETER DATA
                        5148  *
                        5149  *       THE CONTENTS OF VPARD60 ARE COPIED TO THE CTRC BY *ICRT* IF
                        5150  *       THE POWER UP CONFIGURATION SWITCH IS SET TO THE 60HZ POSITION
                        5151
 017.331                5152  VPARD60 EQU     *
 017.331  140           5153          DB      96              HORIZONTAL TOTAL
 017.332  120           5154          DB      80              HORIZONTAL DISPLAYED
 017.333  124           5155          DB      84              HORIZONTAL SYNC POSITION
 017.334  010           5156          DB      8               HORIZONTAL SYNC WIDTH
 017.335  031           5157          DB      25              VERTICAL TOTAL
 017.336  004           5158          DB      4               VERTICAL TOTAL ADJUST
 017.337  030           5159          DB      24              VERTICAL DISPLAYED
 017.340  031           5160          DB      25              VERTICAL SYNC POSITION
 017.341  000           5161          DB      0               INTERLACE MODE
 017.342  011           5162          DB      9               MAXIMUM SCAN LINE
 017.343  110           5163          DB      01001000B       FAST BLINK CURSOR STARTING AT LINE 8
 017.344  010           5164          DB      8               CURSOR END AT LINE 8
 017.345  000           5165          DB      0               MEMORY STARTING ADDRESS (MSB)
 017.346  000           5166          DB      0               MEMORY STARTING ADDRESS (LSB)
 017.347  000           5167          DB      0               CURSOR STARTING ADDRESS (MSB)
 017.350  000           5168          DB      0               CURSOR STARTING ADDRESS (LSB)


                        5170  **      VPARD50 - VIDEO PARAMETER DATA FOR 50 HERTZ
                        5171  *
                        5172  *       THE CONTENTS OF VPARD50 ARE COPIED TO THE CRTC BY *ICRT* IF
                        5173  *       THE POWER UP CONFIGURATION SWITCH IS SET TO THE 50 HZ POSITION
                        5174
                        5175
 017.351                5176  VPARD50 EQU     *
 017.351  140           5177          DB      96              HORIZONTAL TOTAL
 017.352  120           5178          DB      80              HORIZONTAL DISPLAYED
 017.353  124           5179          DB      84              HORIZONTAL SYNC POSITION
 017.354  010           5180          DB      8               HORIZONTAL SYNC WIDTH
 017.355  036           5181          DB      30              VERTICAL TOTAL
 017.356  007           5182          DB      7               VERTICAL TOTAL ADJUST
 017.357  030           5183          DB      24              VERTICAL DISPLAYED
 017.360  033           5184          DB      27              VERTICAL SYNC POSITION
 017.361  000           5185          DB      0               INTERLACE MODE
 017.362  011           5186          DB      9               MAXIMUM SCAN LINE
 017.363  110           5187          DB      01001000B       FAST BLINK CURSOR STARTING AT LINE 8
 017.364  010           5188          DB      8               CURSOR END AT LINE EIGHT
 017.365  000           5189          DB      0               MEMORY STARTING ADDRESS (MSB)
 017.366  000           5190          DB      0               MEMORY STARTING ADDRESS (LSB)
 017.367  000           5191          DB      0               CURSOR STARTING ADDRESS (MSB)
 017.370  000           5192          DB      0               CURSOR STARTING ADDRESS (LSB)
                        5193
 017.371  314           5194          DB      48H+43H+41H     MORAL SUPPORT
 017.372  110 061 071   5195          DB      'H19'           FINAL CODE IDENTIFIER
 017.375  122 116 102   5196          DB      52H,4EH,42H     CODE SUPPORT CODE
                        5197
```

```
                        5200  ***     RAM ALLOCATIONS


                        5202          ORG     100000A
100.000                 5203  RAM     EQU     *               256 BYTE SCRATCHPAD RAM AREA
100.000                 5204  INF     DS      128             INPUT FIFO
000.000                 5205          ERRNZ   INF&1111111B    7 LSB MUST BE ZERO
000.000                 5206          ERRNZ   *-INF/256       FIFO MUST RESIDE IN ONE PAGE
000.200                 5207  IFMAX   EQU     *-INF
100.200                 5208  OUTF    DS      32              OUTPUT FIFO
000.000                 5209          ERRNZ   OUTF&11111B     5 LSB MUST BE ZERO
000.000                 5210          ERRNZ   *-OUTF/256      FIFO MUST RESIDE IN ONE PAGE
000.040                 5211  OFMAX   EQU     *-OUTF
                        5212
100.240                 5213  IFP     DS      1               INPUT FIFO POINTER
100.241                 5214  IFC     DS      1               INPUT FIFO COUNTER
000.177                 5215  IFCMSK  EQU     IFMAX-1
100.242                 5216  OFP     DS      1               OUTPUT FIFO POINTER
100.243                 5217  OFC     DS      1               OUTPUT FIFO COUNTER
000.037                 5218  OFCMSK  EQU     OFMAX-1
                        5219
100.244                 5220  KBDFMIN EQU     *               BEGINNING OF FIFO
100.244                 5221  KBDF    DS      16              KEYBOARD FIFO (8 ENTRYS)
100.264                 5222  KBDFMAX EQU     *               END OF FIFO
000.020                 5223  KBDFL   EQU     KBDFMAX-KBDFMIN         KEYBOARD FIFO LENGTH
100.264                 5224  KBDFP   DS      2               KEYBOARD FIFO POINTER


                        5226  *       VIDEO AND CURSOR POSITIONS
                        5227  *
100.266                 5228  SHOME   DS      2               SOFTWARE HOME POSITION
100.270                 5229  CLSA    DS      2               CURRENT LINE STARTING ADDRESS
100.272                 5230  CURHP   DS      1               CURSOR HORIZONTAL POSITION
100.273                 5231  CURVP   DS      1               CURSOR VERTICAL POSITION
                        5232
100.274                 5233  CURAD   DS      2               CURSOR ADDRESS (ACTUAL MEMORY ADDRESS)
077.377                 5234  CURMAX  EQU     77377A          MAXIMUM VALUE OF CURAD FOR CRTC (11 BITS)
                        5235
                        5236  *       VIDEO PARAMETERS TO BE SENT TO THE CRTC BY *NMI*
                        5237  *
                        5238
100.276                 5239  VI.VD   DS      1               VERTICAL DISPLAYED VALUE
100.277                 5240  VI.CSE  DS      2               CURSOR START/OFF AND END VALUES
100.301                 5241  VI.SA   DS      2               VIDEO START ADDRESS (LIMIT 2K)
100.303                 5242  VI.CA   DS      2               CURSOR ADDRESS
                        5243
100.305                 5244  CSA     DS      2               CURSOR SAVED ADDRESS
                        5245
                        5246  *       MODE AND STATUS DEFINITIONS
                        5247  *
100.307                 5248  MODEA   DS      1               MODE REGISTER A
000.200                 5249  MA.RV   EQU     10000000B       REVERSE VIDEO MODE
000.100                 5250  MA.ICM  EQU     01000000B       INSERT CHARACTER MODE
```

```
    000.040             5251  MA.BRK  EQU     00100000B       BREAK KEY FLAG
    000.020             5252  MA.CD   EQU     00010000B       CURSOR DISABLED
    000.010             5253  MA.RVP  EQU     00001000B       REVERSE VIDEO PRESENT
    000.004             5254  MA.     EQU     00000100B
    000.002             5255  MA.GRPH EQU     00000010B       IN GRAPHICS MODE
    000.001             5256  MA.HSM  EQU     00000001B       HOLD SCREEN MODE
                        5257
    100.310             5258  MODEB   DS      1               MODE REGISTER B
    000.001             5259  MB.CBLK EQU     00000001B       CURSOR = BLOCK
    000.002             5260  MB.NOTK EQU     00000010B       NO TICK ON KEYBOARD
    000.004             5261  MB.WRAP EQU     00000100B       WRAP AROUND AT END OF LINE
    000.010             5262  MB.ALF  EQU     00001000B       AUTO LINE FEED ON CARRIAGE RETURN
    000.020             5263  MB.ACR  EQU     00010000B       AUTO CARRIAGE RETURN ON LINE FEED
    000.040             5264  MB.ANSI EQU     00100000B       ANSI ESCAPE MODE
    000.100             5265  MB.KPDS EQU     01000000B       KEYPAD SHIFTED
    000.200             5266  MB.KPDA EQU     10000000B       KEYPAD ALTERNATE
                        5267
    100.311             5268  MODEI   DS      1               MODE REGISTER I (INTERNAL ONLY)
    000.001             5269  MI.PWE  EQU     00000001B       PREVIOUS WAS AN ESCAPE CHARACTER
    000.002             5270  MI.XMTM EQU     00000010B       TRANSMIT MODE (FORCE TERMINAL ON LINE)
    000.004             5271  MI.KID  EQU     00000100B       KEYBOARD INPUT DISABLED
    000.010             5272  MI.ONLN EQU     00001000B       TERMINAL ON LINE
    000.020             5273  MI.XOFF EQU     00010000B       XOFF SENT
    000.040             5274  MI.     EQU     00100000B
    000.100             5275  MI.PFP  EQU     01000000B       PROTECTED FIELDS PRESENT
    000.200             5276  MI.25L  EQU     10000000B       25TH LINE ENABLED
                        5277
    100.312             5278  MODES   DS      1               MODE REGISTER S (SERIAL I/O)
    000.017             5279  MS.BR   EQU     00001111B       BAUD RATE TABLE VECTOR
    000.020             5280  MS.PEN  EQU     00010000B       PARITY ENABLE
    000.040             5281  MS.EPS  EQU     00100000B       EVEN PARITY SELECT
    000.100             5282  MS.SPS  EQU     01000000B       STICK PARITY SELECT
    000.200             5283  MS.FDX  EQU     10000000B       FULL DUPLEX
                        5284
    100.313             5285  HSMLC   DS      1               HOLD SCREEN MODE LINE COUNTER
                        5286
    100.314             5287  PSDW    DS      16              PARAMETER STRING DECODING WORK AREA
    000.334             5288  PSDWE   EQU     *&255           LSB OF END ADDRESS +1 OF WORK AREA
                        5289


    370.000             5291          ORG     370000A
    370.000             5292  VRAMS   EQU     *               VIDEO RAM STARTING ADDRESS
    007.377             5293  HOMAX   EQU     377377A-VRAMS   MAXIMUM VIDEO ADDRESS MASK
    370.000             5294          END
ASSEMBLY COMPLETE
 5294 STATEMENTS
    0 ERRORS DETECTED
11184 BYTES FREE
```

.	000015	3424S	3425	4289S	4290	4308S	4309	
A1M	004376	1370L	3008					
A1RM	005047	1382	1433L					
A1RM1	005056	1439L	1448					
A1SM	005015	1379	1399L					
A1SM1	005024	1405L	1418					
A2M	005073	1466L	3005					
A2RM	005134	1475	1519L					
A2RM1	005137	1523L	1532					
A2SM	005112	1472	1491L					
A2SM1	005115	1495L	1504					
AB.2SB	000004	49E	3383					
AB.5BW	000000	45E						
AB.6BW	000001	46E						
AB.7BW	000002	47E	3390					
AB.8BW	000003	48E	3393					
AB.RI	000020	68E						
AB.CTS	000020	76E						
AB.DCTS	000001	73E						
AB.DDSR	000002	74E						
AB.DLAB	000200	54E	3354					
AB.DR	000001	64E						
AB.DRLS	000010	75E						
AB.DSR	000040	77E						
AB.DTR	000001	57E	3399					
AB.EMS	000010	36E						
AB.EPS	000020	51E						
AB.ERDA	000001	33E	3397					
AB.ERLS	000004	35E						
AB.ETRE	000002	34E	289	478				
AB.FE	000010	67E						
AB.IID	000006	42E						
AB.IIP	000001	41E						
AB.LOOP	000020	61E						
AB.OR	000002	65E						
AB.OUT1	000004	59E						
AB.OUT2	000010	60E						
AB.PE	000004	66E						
AB.PEN	000010	50E						
AB.RDAI	000004	39E	427					
AB.RLSD	000200	78E						
AB.RTS	000002	58E	3399					
AB.SBRK	000100	53E	416	423				
AB.SP	000040	52E						
AB.THRE	000040	69E	447	4887				
AB.TREI	000002	40E	462					
AB.TSRE	000100	70E						
ACDN	005156	1548L	3014					
ACDN1	005170	1556L	1560					
ACE1	012247	3381	3384L					
ACE2	012265	3391	3394L					
ACLFT	005201	1577L	3020					
ACLFT1	005214	1586L	1588					
ACPR	005223	1605L	3054					
ACPR1	005257	1620	1624L					
ACPR2	005311	1636	1640L					
ACRT	005324	1660L	3017					
ACRT1	005337	1669L	1671					

```
ACUP     005346    1688L    3011
ACUP1    005360    1696L    1700
AESCT    004340     991     1305E   1320
AESCTL   000004     992     1320E
AESCTW   000003     993     1319E   1320
AKI      000070     271      312L
AKI1     000072     316L
AKI1.3   000254     339      415L
AKI1.5   000264     329      335     420L
AKI1.7   000277     418      426L
AKI1.75  000330     446L     448
AKI1.8   000352     444      456L
AKI2     000362     428      462L
AKI3     001003     469      472     477L
AKI4     001013     463      482L
AKI4.3   001053     501      509L
AKI4.4   001057     505      514L
AKI4.8   001070     516      522L
AKI5     001111     458      525     534L
AKI6     001113     434      439     459     475     480     486     510     532     537L
AP.DLL   000100      80E     3371
AP.DLM   000101      82E     3374
AP.IER   000101      32E      284     290     477     479    3398
AP.IIR   000102      38E      426
AP.LCR   000103      44E      415     417     422     424    3355    3395
AP.LSR   000105      63E      446    4886
AP.MCR   000104      56E     3400
AP.MSR   000106      72E
AP.RBR   000100      28E      432
AP.THR   000100      30E      451     474    4891
APCA     005371    1721L    3023    3042
APCA1    006004    1728     1731L
APCA2    006023    1732     1742L
APCA3    006026    1734     1739    1744L
APCA4    006034    1747     1750L
APCA5    006042    1751     1754L
APDC     006050    1770L    3038
APDC1    006062    1778L    1782
APDL     006073    1799L    3035
APDL1    006105    1807L    1811
APIL     006116    1829L    3032
APIL1    006130    1837L    1841
ARAMP    012375    3072     3495L
ARM      006141    1858L    3048
ARM1     006150    1864L    1884
ASBR     006206    1899L    3063
ASBR1    006217    1901     1908L
ASCP     015127    3066     4210L
ASGM     006222    1923L    3051
ASGM1    006227    1929L    1933
ASGM1.5  006237    1925     1936L
ASGM2    006262    1940     1948L
ASGMT    006265    1936     1958E   1973
ASGMTL   000004    1937     1973E
ASGMTW   000003    1938     1972E   1973
ASM      006301    1988L    3045
ASM1     006310    1994L    2014
AUSCP    015311    3069     4370L
```

```
EDD2      011027    2767      2773L
EEL       011051    1250      2360      2805      2819L
EGM       011061    1194      1967      2835L
EHM       011065    1407      2852L
EHSM      011076    1231      2870L     4282
EICM      011107    1176      2002      2889L
EICSEQ    000100     785      1174E     1175
EIB       007377    2326L     3026
EII       011031    2789L     3029
EKAM      011113    1166      1308      2906L     4286
EKC       011120    2923L     4300
EKI       011125    1290      1867      2940L
EKSM      011136    1269      2957L     4285
ELB       011143    1317      2978L
ELBT      011172    2985      3002E     3075
ELBTL     000027    2983      3075E
ELBTW     000003    2984      3074E     3075
EOL       011277    1209      2791      2796      3090L     3115
ERM       011317    1206      2328      2333      3113L
ERVM      011361    1259      1964      3147E
ESC       000033     20E       603       607       628       741       745       764       768       776       780       802      1162
                    1346      1614      2178      3479      4955      4960      4978      4982      5005      5009      5027      5031
ESCF      000200     227E       587       610       626       721       727       748       785       788       880       881       882
                     883       884       885       886       887       888       898       898       899       899       899       900       900
                     900       901       901       901       902       902       902       903       903       903       904       904       904
                     905       905       905       906       906       906       907       907       907       908       908       909       909
ESCTAB    004134    996       1157E     1296
ESCTABL   000054    997       1296E
ESCTABW   000003    998       1295E     1296
FCIF      011367     261      3171L     3249
FCOF      012032     471      3209L
FNCP      012070    3245L     3317      3656      3680      4073      4112      4250
FNCP1     012073    3248L     3250
FVKF      012107     264      1034      3274L
FVKF1     012153    3281      3299L
HOMAX     007377    2093      2295      4578      5293E
HSMLC     100313     820      1026      1043      1062      2874      5285L
HT        000011      14E     1337
I.BITA    000313     155E       286       438       468       591       599       623       633       655       696       701       707
                     737       760       772       822       835       846       858       960       977      1057      4945      4967      4989
                    4995      5016
I.BITB    000100     156E       286       438       468       591       599       623       633       655       696       701       707
                     737       760       772       822       835       846       858       960       977      1057      4945      4967      4989
                    4995      5016
I.BITHA   000313     157E
I.BITHB   000106     158E
I.EXAF    000010     173E       312       537
I.EXX     000331     174E       314       539
I.IM1A    000355     159E      3590
I.IM1B    000126     160E      3590
I.JMP     000303     187E
I.LDDA    000355     177E      3978      4046
I.LDDB    000250     178E      3978      4046
I.LDDRA   000355     179E      4034
I.LDDRB   000270     180E      4034
I.LDEDA   000355     175E      3123      3861      4015
I.LDEDB   000133     176E      3123      3861      4015
I.LDIA    000355     181E      3875
```

```
I.LDIB  000240    182E    3875
I.LDIRA 000355    183E    3295    3547    3931
I.LDIRB 000260    184E    3295    3547    3931
I.NEGA  000355    161E    3093    3856    3910    3958    4004
I.NEGB  000104    162E    3093    3856    3910    3958    4004
I.OUT   000323    185E
I.RESA  000313    169E    4971    5020
I.RESB  000200    170E    4971    5020
I.RESHA 000313    171E    4743    4760    4777    5143
I.RESHB 000206    172E    4743    4760    4777    5143
I.RET   000311    186E
I.RETNA 000355    188E     413
I.RETNB 000105    189E     413
I.SETA  000313    163E    4949    4999
I.SETB  000300    164E    4949    4999
I.SETHA 000313    165E    2837    2872    2891
I.SETHB 000306    166E    2837    2872    2891
I.SHDA  000355    167E    3126
I.SHDB  000122    168E    3126
IACE    012204   3339    3354L   3587
IACE0.5 012221   3362    3365L
IB.BRK  000020    209E
IB.CPLK 000010    197E     846
IB.ESCF 000070    198E     623
IB.ETRE 000010    199E     286
IB.GRPH 000010    210E    2837    4743
IB.HSM  000000    203E    2872    4760
IB.ICM  000060    205E    2891    4777
IB.IFF  000070    200E
IB.KCB  000070    201E     599     633     655     737     760     772     858
IB.KPDA 000070    206E
IB.KPDS 000060    207E     701     707
IB.KSB  000000    202E     591     696     822     835    1057
IB.ONLN 000030    208E     438     468
IB.PWE  000000    211E     977
IB.RV   000070    204E    4989    5143
IB.XMTG 000010    213E    4945    4949    4967    4971
IB.XMTR 000020    214E    4995    4999    5016    5020
IB.XOFF 000040    212E     960
ICRT    012333   3441E    3586
ICRT0.5 012347   3447     3451L
ICRT1   012353   3454L    3460
IDT     012366   1228     3478L
IFC     100241    442     964     1038    3171    3177    3179    3712    3727    5214L
IFCMSK  000177   3187    3720    5215E
IFCP    003120    262     957L    4255    4258
IFCP0.5 003155    961     967     977L
IFCP0.7 003213    989     996L
IFCP0.9 003222    994    1000L
IFCP1   003230    979    1005L
IFCP1.1 003273   1034L   1040
IFCP1.3 003303   1038L   1053
IFCP1.5 003322   1035    1046L
IFCP1.7 003341   1049    1055E
IFCP1.8 003351   1059    1062L
IFCP1.9 003364   1006    1067L
IFCP2   003374   1068    1075L
IFCP3   004007   1001    1083L
```

```
IFCP4   004022   1071   1091L
IFCP5   004045   1099   1104L
IFCP6   004047   1094   1097   1102   1108L
IFCP7   004124   1127   1141L
IFF     000200    226E
IFMAX   000200    443    965   1039   3713   5207E  5215
IFP     100240   3180   3188   3718   5213L
INF     100000   3182   3721   5204L  5205   5206   5207
INIT    013001    237   3516   3525E
INIT1   013114   3586L
IR.A    000007    216E    286    623    701    707    960    977
IR.B    000000    217E   4945   4949   4967   4971   4995   4999   5016   5020
IR.C    000001    218E    438    468   4989
IR.D    000002    219E    599    633    655    737    760    772    858
IR.E    000003    220E    591    696    822    835    846   1057
IR.H    000004    221E
IR.L    000005    222E
IRAM    013001   3527   3542L
KAE1    002350    580    879E    890
KAE1L   000011    581    890E
KAE1W   000002    582    889E    890
KAE2    002372    687    897E    911
KAE2L   000014    688    911E
KAE2W   000005    689    910E    911
KAE3    003066    839    919E    934
KAE3L   000015    840    934E
KAE3W   000002    841    933E    934
KB.BRK  000004    125E    327
KB.CHAR 000177    117E
KB.CPLK 000002    124E
KB.CTL  000200    118E    499    614
KB.EX1  000013    119E    499
KB.EX2  000014    120E
KB.ONLN 000010    126E    319    324
KB.RPT  000100    127E
KB.SHFT 000001    123E
KB.STB  000200    128E    485
KBDF    100244   3286   3292   3549   5221L
KBDFL   000020   3293   5223E
KBDFMAX 100264    524   5222E  5223
KBDFMIN 100244   3279   5220E  5223
KBDFP   100264    522    531   3277   3285   3550   5224L
KCE     001117    265    568L   1051
KCE1    001133   579L
KCE1.2  001204    600    606L
KCE1.4  001214    596    610L
KCE1.5  001220    585    613L
KCE10   002015    713    717L
KCE10.1 002130    761    767L
KCE10.2 002140    757    772L
KCE10.3 002154    773    779L
KCE10.4 002164    753    783L
KCE10.5 002203    722    791L
KCE11   002214    725    801L
KCE11.5 002235    807    810L
KCE12   002246    684    817L
KCE12.5 002263    823    828L
KCE13   002266    818    835L
```

```
KCE13.5 002306   836   846L
KCE14   002324   847   850   853   858L
KCE2    001226   588   592   611   623L  749   786   789   795
KCE2.3  001271   647   650L
KCE2.7  001276   641   653L
KCE3    001277   574   616   624   655L  719
KCE4    001311   634   663L
KCE4.3  001335   672   675L
KCE4.7  001340   666   677L
KCE5    001341   656   678L
KCE6    001344   577   683L
KCE7    002002   698   707L
KCE8    002006   703   710L
KCE9    002007   702   708   712L
KE10.03 002062   738   744L
KE10.07 002071   728   734   748L
KP.1    000200   116E  490
KP.2    000240   122E  316   482   492
LF      000012    15E  1005  1340
MA.     000004  5254E
MA.BRK  000040   270   337   420  5251E
MA.CD   000020  2523  2743  4328  4504  5252E
MA.GRPH 000002  1092  5255E
MA.HSM  000001  1011  5256E
MA.ICM  000100   756   784  1115  5250E
MA.RV   000200  1110  1112  3152  5249E
MA.RVP  000010  3152  5253E
MAIN    000004   258   258L  3594
MAIN.N  000016   264L  3248
MAIN1   000037   273L
MAIN2   000066   282   287   291L
MAINA   000045   279L  1032  1052
MB.ACR  000020  2126  2626  4709  5263E
MB.ALF  000010  2505  2644  3812  4726  5262E
MB.ANSI 000040   595   640   665   732   752   805   987  2661  2853  4951  4974  5001
                5023  5264E
MB.CBLK 000001  2746  4325  4501  5259E
MB.KPDA 000200   712  2907  4793  5266E
MB.KPDS 000100  2958  4810  5265E
MB.NOTK 000002   515  2924  3635  5260E
MB.WRAP 000004  1130  2545  4607  5261E
MI.     000040  5274E
MI.25L  000200  1738  2070  2268  2486  2596  2599  3672  4848  5276E
MI.KID  000004   333   503   509  2576  2941  5271E
MI.ONLN 000010   274   322   324  3748  5272E
MI.PFP  000100  5275E
MI.PWE  000001   984  4482  5269E
MI.XMTM 000002  5270E
MI.XOFF 000020   453   969  5273E
MODEA   100307   269   328   338   421   755   783  1010  1088  1091  1109  1114  1864
                1945  1994  2522  2524  2742  2744  4270  5248L
MODEB   100310   514   594   639   664   697   731   751   804   986  1087  1129  1404
                1438  1863  1993  2125  2745  2852  2854  3564  3568  3811  4269  4950  4973
                5000  5022  5258L
MODEI   100311   273   321   325   332   452   454   500   504   958   970   983   985
                1737  2069  2267  2485  2487  2575  2577  2594  2601  2940  2942  3570  3671
                3747  4481  4483  4847  5268L
MODES   100312  3334  3337  3359  3559  3757  5278L
```

```
MP.BELL 000340   134E   534   2560   4862   5093
MP.PUP1 000000   136E   3557
MP.PUP2 000040   143E   3443   3561   3566
MP.TICK 000300   132E   518
MPY80   013137   3608L  4412   4570
MS.BR   000017   3335   3361   5279E
MS.EPS  000040   5281E
MS.FDX  000200   3758   5283E
MS.PEN  000020   5280E
MS.SPS  000100   5282E
NKC     013155   3634L  4281
NMI     000146   357L
NULL    000000   11E
OFC     100243   280    3209   3215   3762   3777   5217L
OFCMSK  000037   3223   3770   5218E
OFMAX   000040   3763   5211E  5218
OFP     100242   3216   3224   3768   5216L
OUTF    100200   3218   3771   5208L  5209   5210   5211
P1.BR   000017   137E   3329
P1.EPS  000040   139E   3385
P1.FDX  000200   141E
P1.PEN  000020   138E   3385   3389
P1.SPS  000100   140E   3385
P2.50HZ 000200   151E   3445   3567
P2.ALF  000010   147E
P2.CBLK 000001   144E
P2.KPDS 000100   150E
P2.NOSC 000020   148E
P2.NOTK 000002   145E
P2.VT52 000040   149E
P2.WRAP 000004   146E
PBS     007126   1335   2146E
PBS1    007143   2156L
PCA     013162   1225   3656L
PCA1    013170   3662L
PCA1.5  013213   3666   3677L
PCA2    013220   3663   3668   3673   3680L
PCA3    013227   3686L
PCA4    013237   3687   3692L
PCA5    013241   3690   3693L
PCA6    013246   3682   3696L
PCIF    013251   457    637    651    659    3711L  3754   3760   4146   4153
PCOF    013311   3746L  3794
PCOF1   013326   3749   3757L
PCOF2   013374   3765   3781L
PCOFT   013375   663    675    678    803    810    813    859    862    865    868    973    1031
                 1065   1623   1626   1639   1642   2181   2184   3793L  4176   4181
PCR     014013   1139   2127   3815   3829L
PCRLF   014003   1344   3811L
PDC     014027   1218   1772   1776   1779   3851E
PDC1    014053   3868L  3878
PDC2    014071   3863   3880L
PDL     014075   1215   1801   1805   1808   3898E
PDL1    014134   3921L  3934
PDL2    014157   3912   3936L
PIC     014165   1116   3953E
PIC1    014212   3971L  3981
PIL     014234   1212   1831   1835   1838   3999E
```

```
PIL1     014271    4024L     4037
PIL2     014326    4006      4048L
PLF      006354    1138      2035     2051L    2124     3813
PLF0.2   006362    2056L
PLF0.5   007034    2071      2083L
PLF1     007070    2065      2098L
PLF2     007074    2096      2101L
PLFCR    007112    1012      1024     1028     1044     1063     1341     2124L
PRLF     007234    1203      1314     2235     2251L
PRLF0.2  007242    2256L
PRLF0.5  007316    2269      2284L
PRLF1    007355    2265      2300L
PRLF2    007361    2298      2303L
PSD      014346    1374      1470     2978     4070L
PSD1     014363    4077L     4113
PSD2     015012    4081      4102L
PSD3     015024    4107      4110L
PSD3.5   015030    4100      4112L
PSD4     015035    4078      4103     4115L
PSD5     015043    4118      4121L
PSDW     100314    2993      4070     4123     5287L
PSDWE    000334    4106      5288E
PSIF     015051     602       740      763      775     4139L
PSIF1    015052    4140L     4148
PSIF2    015067    4143      4150L
PSOF     015100     606       744      767      779     1613     1628     1643     2177     3478     4170L
PSOF1    015101    4171L     4177
PSOF2    015114    4174      4179L
RAM      100000    3542      3543     3573     5203E
RAMP     013000    1287      3500     3514L
RMS      015122    1284      4194L
RMST     015227    1523      4194     4298E    4309
RUBOUT   000177      21E     1067
SBC      015251    4283      4324L
SBC,     015261    2747      4331L
SBR      012157    1265      3317L
SBR.     012172    1908      3333L
SBR1     012162    3323L
SCH      015271    1200      1723     2363     4348L
SCP      015132    1244      4215     4230L
SCRL     000037     228E     1048
SHOME    100266    2085      2091     2273     2287     2293     2602     2697     3117     3576     4016     4351     4414
                   4576      4851     5081     5228L
SMS      015141    1281      4248L
SMSA     015144    4195      4250L
SMSB     015166    1501      1529     4261L
SMST     015205    1495      4248     4279E    4290
SNCP     015322    1755      3696     4395     4411L
SPWE     016005    1163      1347     4481L
STAB     015363     583       690      842     1000     1078     1939     2986     4451L
STAB1    015364    4452L     4461
STAB2    015371    4457L     4458
STAB3    016002    4453      4467L
START    000000     237L
SUC      016016    4302      4500L
SUC.     016026    2749      4507L
TAB      016036    1338      4527L
TAB1     016060    4538L
```

```
TAB1.5   016063   4536   4541L
TAB2     016067   4531   4539   4542   4544L
UCP      016111    267   1037   4569L
UCP.     016146   2109   2311   4588L
USCP     015314   1247   4375   4390L
VA.CAL   000017    104E    400
VA.CAM   000016    103E    396
VA.CE    000013    100E    376
VA.CS    000012     96E    372
VA.HD    000001     87E
VA.HSP   000002     88E
VA.HSW   000003     89E
VA.IM    000010     94E
VA.LPL   000021    106E
VA.LPM   000020    105E
VA.MSLA  000011     95E
VA.SAL   000015    102E    388
VA.SAM   000014    101E    384
VA.VD    000006     92E    364
VA.VSP   000007     93E
VA.VT    000004     90E
VA.VTA   000005     91E
VB.CBE   000100     97E   3553   4331   4507
VB.CBPS  000040     98E   4331
VB.CND   000040     99E   2525
VB.NMI   000004    112E   3592   4589
VB.RBD   000010    111E
VI.CA    100303    395   4584   5242L
VI.CSE   100277    371   2527   3555   4333   4509   5240L
VI.SA    100301    383   2095   2297   5241L
VI.VD    100276    366   2489   2609   3552   5239L
VP.AR    000140     86E    365    373    377    385    389    397    401   3455   3592   4589
VP.REGI  000143    110E
VP.REGO  000141    108E    367    375    379    387    391    399    403   3457
VPARD50  017351   3446   5176E
VPARD60  017331   3449   5152E
VRAMS    370000   1123   2060   2089   2104   2157   2210   2260   2291   2306   2382   2385   3575
                  3869   3872   3923   3926   3972   3975   4026   4029   4040   4043   4417   4426   4550
                  4657   4855   4917   5292E  5293
WEOL     016153   1275   1411   4606L
WSV      016160   3097   3133   3580   4625E
WSV0.5   016160   4628L  4636
WSV2     016174   4630   4638L
WSV3     016176   4639L  4645
WSV4     016214   4656L  4691
WSVA     016212   2084   2286   2365   2698   2821   3938   4053   4652E
WSVA.    016212   2606   4655L
XACR     016263   1877   4307   4708L
XALF     016270   4306   4725L
XGM      016275   1197   1970   4741L
XHSM     016301   1234   4301   4758L
XICM     016305   1222   1872   4775L
XICSEQ   000117    788   1220E  1221
XKAM     016311   1169   1311   4305   4792L
XKSM     016316   1272   4304   4809L
XMT25    016326   1237   4832   4847L
XMT25.1  016356   4849   4860L
XMTC     016366   4861   4884L   5034   5092   5118   5122
```

XMTC1	016367	4886L	4888							
XMTL	017001	4859	4911L	5086						
XMTL1	017033	4923	4933L							
XMTL1.1	017003	4912L	5038							
XMTL1.3	017043	4934	4939L							
XMTL2	017045	4931	4937	4940L						
XMTL3	017075	4952	4959L							
XMTL4	017104	4926	4965L							
XMTL5	017134	4975	4981L							
XMTL6	017141	4946	4957	4961	4968	4979	4986E			
XMTL7	017173	5002	5008L							
XMTL7.5	017202	4990	5014E							
XMTL8	017227	5024	5030L							
XMTL9	017234	4996	5006	5010	5017	5028	5033L			
XMTP	017252	1172	5060	5081L						
XMTP1	017262	5085L	5089							
XMTS	017302	4954	4959	4977	4981	5004	5008	5026	5030	5112L
XMTS1	017303	5113L	5119							
XMTS2	017316	5116	5121L							
XOFF	000023	18E	450	1030						
XON	000021	17E	972	1064						
XRVM	017325	1262	1961	5138E						

14484 BYTES FREE