APPENDIX B. MONITOR LISTING

```
8080 MACRO ASSEMBLER, VER 2.3  ERRORS = 0 PAGE 1

                ;*****************************************************************
                ;
                ;               PROGRAM: 8080A BOARD MONITOR
                ;
                ;               COPYRIGHT (C) 1975
                ;               INTEL CORPORATION
                ;               3065 BOWERS AVENUE
                ;               SANTA CLARA, CALIFORNIA  95051
                ;
                ;*****************************************************************
                ;
                ;
                ; ABSTRACT
                ; ========
                ;
                ; THIS PROGRAM RUNS ON THE 8080A BOARD AND IS DESIGNED TO PROVIDE
                ; THE USER WITH A MINIMAL MONITOR.  BY USING THIS PROGRAM,
                ; THE USER CAN EXAMINE AND CHANGE MEMORY OR CPU REGISTERS, LOAD
                ; A PROGRAM (IN ABSOLUTE HEX) INTO RAM, AND EXECUTE INSTRUCTIONS
                ; ALREADY IN MEMORY.  THE MONITOR ALSO PROVIDES THE USER WITH
                ; ROUTINES FOR PERFORMING CONSOLE I/O.
                ;
                ;
                ; PROGRAM ORGANIZATION
                ; ======= ============
                ;
                ; THE LISTING IS ORGANIZED IN THE FOLLOWING WAY.  FIRST THE COMMAND
                ; RECOGNIZER, WHICH IS THE HIGHEST LEVEL ROUTINE IN THE PROGRAM.
                ; NEXT THE ROUTINES TO IMPLEMENT THE VARIOUS COMMANDS.  FINALLY,
                ; THE UTILITY ROUTINES WHICH ACTUALLY DO THE DIRTY WORK.  WITHIN
                ; EACH SECTION, THE ROUTINES ARE ORGANIZED IN ALPHABETICAL
                ; ORDER, BY ENTRY POINT OF THE ROUTINE.
                ;
                ; THIS PROGRAM EXPECTS TO RUN IN THE FIRST 1K OF ADDRESS SPACE.
                ; IF, FOR SOME REASON, THE PROGRAM IS RE-ORG'ED, CARE SHOULD
                ; BE TAKEN TO MAKE SURE THAT THE TRANSFER INSTRUCTIONS FOR RST 1
                ; AND RST 7 ARE ADJUSTED APPROPRIATELY.
                ;
                ; THE PROGRAM ALSO EXPECTS THAT RAM LOCATIONS 5K-1 TO 5K-256,
                ; INCLUSIVE, ARE RESERVED FOR THE PROGRAM'S OWN USE.  THESE
                ; LOCATIONS MAY BE ALTERED, HOWEVER, BY CHANGING THE EQU'ED
                ; SYMBOL "DATA" AS DESIRED.
                ;
                ; LIST OF FUNCTIONS
                ; ==== == =========
                ;
                ;       GETCM
                ;       -----
                ;
                ;       DCMD
                ;       GCMD
```

```
                ;       ICMD
                ;       MCMD
                ;       SCMD
                ;       XCMD
                ;       -----
                ;
                ;       BREAK
                ;       CI
                ;       CNVBN
                ;       CO
                ;       CROUT
                ;       ECHO
                ;       ERROR
                ;       FRET
                ;       GETCH
                ;       GETHX
                ;       GETNM
                ;       HILO
                ;       NMOUT
                ;       PRVAL
                ;       REGDS
                ;       RGADR
                ;       RSTTF
                ;       SRET
                ;       STHF0
                ;       STHLF
                ;       VALDG
                ;       VALDL
                ;       -----
                ;
  0000                  ORG     0H
                ;
                ;*****************************************************************
                ;
                ;
                ;                       MONITOR EQUATES
                ;
                ;
                ;*****************************************************************
                ;
                ;
  001B          BRCHR EQU     1BH     ; CODE FOR BREAK CHARACTER (ESCAPE)
  13FD          BRLOC EQU     13FDH   ; LOCATION OF USER BRANCH INSTRUCTION IN RAM
  03FA          BRTAB EQU     3FAH    ; LOCATION OF START OF BRANCH TABLE IN ROM
  0027          CMD   EQU     027H    ; COMMAND INSTRUCTION FOR USART INITIALIZATION
  00FB          CNCTL EQU     0FBH    ; CONSOLE (USART) CONTROL PORT
  00FA          CNIN  EQU     0FAH    ; CONSOLE INPUT PORT
  00FA          CNOUT EQU     0FAH    ; CONSOLE OUTPUT PORT
  00FB          CONST EQU     0FBH    ; CONSOLE STATUS INPUT PORT
  000D          CR    EQU     0DH     ; CODE FOR CARRIAGE RETURN
  1300          DATA  EQU     5*1024-256      ; START OF MONITOR RAM USAGE
```

```
001B            ESC    EQU     1BH     ; CODE FOR ESCAPE CHARACTER
000F            HCHAR  EQU     0FH     ; MASK TO SELECT LOWER HEX CHAR FROM BYTE
00FF            INVRT  EQU     0FFH    ; MASK TO INVERT HALF BYTE FLAG
000A            LF     EQU     0AH     ; CODE FOR LINE FEED
0000            LOWER  EQU     0       ; DENOTES LOWER HALF OF BYTE IN ICMD
                ;LSGNON        EQU     ---     ; LENGTH OF SIGNON MESSAGE - DEFINED LATE
00CF            MODE   EQU     0CFH    ; MODE SET FOR USART INITIALIZATION
                ;MSTAK         EQU     ---     ; START OF MONITOR STACK - DEFINED LATER
                ;NCMDS         EQU     ---     ; NUMBER OF VALID COMMANDS
000F            NEWLN  EQU     0FH     ; MASK FOR CHECKING MEMORY ADDR DISPLAY
007F            PRTY0  EQU     07FH    ; MASK TO CLEAR PARITY BIT FROM CONSOLE CHAR
13ED            REGS   EQU     DATA+255-18     ; START OF REGISTER SAVE AREA
0002            RBR    EQU     2       ; MASK TO TEST RECEIVER STATUS
0038            RSTU   EQU     38H     ; TRANSFER LOCATION FOR RST 7 INSTRUCTION
                ;RTABS         EQU     ---     ; SIZE OF ENTRY IN RTAB TABLE
001B            TERM   EQU     1BH     ; CODE FOR ICMD TERMINATING CHARACTER (ESCAPE)
0001            TRDY   EQU     1       ; MASK TO TEST TRANSMITTER STATUS
00FF            UPPER  EQU     0FFH    ; DENOTES UPPER HALF OF BYTE IN ICMD
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ;                       MONITOR MACROS
                ;
                ;
                ;*****************************************************************
                ;
                ;
      1         TRUE   MACRO   WHERE   ; BRANCH IF FUNCTION RETURNS TRUE (SUCCESS)
      1                JC      WHERE
                       ENDM
                ;
      1         FALSE  MACRO   WHERE   ; BRANCH IF FUNCTION RETURNS FALSE (FAILURE)
      1                JNC     WHERE
                       ENDM
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ;                       USART INITIALIZATION CODE
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ;   THE USART IS ASSUMED TO COME UP IN THE RESET POSITION (THIS
                ;   FUNCTION IS TAKEN CARE OF BY THE HARDWARE).  THE USART WILL
                ;   BE INITIALIZED IN THE SAME WAY FOR EITHER A TTY OR CRT
                ;   INTERFACE.  THE FOLLOWING PARAMETERS ARE USED:
```

```
                        ;
                        ;       MODE INSTRUCTION
                        ;       ==== ===========
                        ;
                        ;       2 STOP BITS
                        ;       PARITY DISABLED
                        ;       8 BIT CHARACTERS
                        ;       BAUD RATE FACTOR OF 64
                        ;
                        ;       COMMAND INSTRUCTION
                        ;       ======= ===========
                        ;
                        ;       NO HUNT MODE
                        ;       NOT(RTS)  FORCED TO 0
                        ;       RECEIVE ENABLED
                        ;       DATA TERMINAL READY
                        ;       TRANSMIT ENABLED
                        ;
0000    3ECF                    MVI     A,MODE
0002    D3FB                    OUT     CNCTL   ; OUTPUT MODE SET TO USART
0004    3E27                    MVI     A,CMD
0006    D3FB                    OUT     CNCTL   ; OUTPUT COMMAND WORD TO USART
                        ;
                        ;*****************************************************************
                        ;
                        ;
                        ;                       RESTART ENTRY POINT
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ;
0008                    GO:
0008    22F313                  SHLD    LSAVE   ; SAVE HL REGISTERSS
000B    E1                      POP     H       ; GET TOP OF STACK ENTRY
000C    22F513                  SHLD    PSAVE   ; ASSUME THIS IS LAST P COUNTER
000F    210000                  LXI     H,0     ; CLEAR HL
0012    39                      DAD     SP      ; GET STACK POINTER VALUE
0013    22F713                  SHLD    SSAVE   ; SAVE USER'S STACK POINTER
0016    21F313                  LXI     H,ASAVE+1       ; NEW VALUE FOR STACK POINTER
0019    F9                      SPHL            ; SET MONITOR STACK POINTER FOR REG SAVE
001A    F5                      PUSH    PSW     ; SAVE A AND FLAGS
001B    C5                      PUSH    B       ; SAVE B AND C
001C    D5                      PUSH    D       ; SAVE D AND E
                        ;
                        ;*************************************************************
                        ;
                        ;
                        ;                       PRINT SIGNON MESSAGE
                        ;
                        ;
```

```
                        ;*************************************************************
                        ;
                        ;
001D   219D03                LXI     H,SGNON ; GET ADDRESS OF SIGNON MESSAGE
0020   060E                  MVI     B,LSGNON        ; COUNTER FOR CHARACTERS IN MESSAGE
0022                    MSGL:
0022   4E                    MOV     C,M     ; FETCH NEXT CHAR TO C REG
0023   CDE301                CALL    CO      ; SEND IT TO THE CONSOLE
0026   23                    INX     H       ; POINT TO NEXT CHARACTER
0027   05                    DCR     B       ; DECREMENT BYTE COUNTER
0028   C22200                JNZ     MSGL    ; RETURN FOR NEXT CHARACTER
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ;
                        ;                       COMMAND RECOGNIZING ROUTINE
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ; FUNCTION: GETCM
                        ; INPUTS: NONE
                        ; OUTPUTS: NONE
                        ; CALLS: GETCH,ECHO,ERROR
                        ; DESTROYS: A,B,C,H,L,F/F'S
                        ; DESCRIPTION: GETCM RECEIVES AN INPUT CHARACTER FROM THE USER
                        ;               AND ATTEMPTS TO LOCATE THIS CHARACTER IN ITS COMMAND
                        ;               CHARACTER TABLE.  IF SUCCESSFUL, THE ROUTINE
                        ;               CORRESPONDING TO THIS CHARACTER IS SELECTED FROM
                        ;               A TABLE OF COMMAND ROUTINE ADDRESSES, AND CONTROL
                        ;               IS TRANSFERRED TO THIS ROUTINE.  IF THE CHARACTER
                        ;               DOES NOT MATCH ANY ENTRIES, CONTROL IS PASSED TO
                        ;               THE ERROR HANDLER.
                        ;
002B                    GETCM:
002B   21ED13                LXI     H,MSTAK ; ALWAYS WANT TO RESET STACK PTR TO MONITOR
002E   F9                    SPHL            ; /STARTING VALUE SO ROUTINES NEEDN'T CLEAN UP
002F   0E2E                  MVI     C,'.'   ; PROMPT CHARACTER TO C
0031   CDF401                CALL    ECHO    ; SEND PROMPT CHARACTER TO USER TERMINAL
0034   C33B00                JMP     GTC03   ; WANT TO LEAVE ROOM FOR RST BRANCH
                        ;
0038                          ORG     RSTU    ; ORG TO RST TRANSFER LOCATION
0038   C3FD13                JMP     USRBR   ; JUMP TO USER BRANCH LOCATION
                        ;
003B                    GTC03:
003B   CD1B02                CALL    GETCH   ; GET COMMAND CHARACTER TO A
003E   CDF401                CALL    ECHO    ; ECHO CHARACTER TO USER
0041   79                    MOV     A,C     ; PUT COMMAND CHARACTER INTO ACCUMULATOR
0042   010600                LXI     B,NCMDS ; C CONTAINS LOOP AND INDEX COUNT
0045   21B903                LXI     H,CTAB  ; HL POINTS INTO COMMAND TABLE
```

```
0048             GTC05:
0048   BE              CMP     M       ; COMPARE TABLE ENTRY AND CHARACTER
0049   CA5400          JZ      GTC10   ; BRANCH IF EQUAL - COMMAND RECOGNIZED
004C   23              INX     H       ; ELSE, INCREMENT TABLE POINTER
004D   0D              DCR     C       ; DECREMENT LOOP COUNT
004E   C24800          JNZ     GTC05   ; BRANCH IF NOT AT TABLE END
0051   C30D02          JMP     ERROR   ; ELSE, COMMAND CHARACTER IS ILLEGAL
0054             GTC10:
0054   21AB03          LXI     H,CADR  ; IF GOOD COMMAND, LOAD ADDRESS OF TABLE
                                       ; /OF COMMAND ROUTINE ADDRESSES
0057   09              DAD     B       ; ADD WHAT IS LEFT OF LOOP COUNT
0058   09              DAD     B       ; ADD AGAIN - EACH ENTRY IN CADR IS 2 BYTES LON
0059   7E              MOV     A,M     ; GET LSP OF ADDRESS OF TABLE ENTRY TO A
005A   23              INX     H       ; POINT TO NEXT BYTE IN TABLE
005B   66              MOV     H,M     ; GET MSP OF ADDRESS OF TABLE ENTRY TO H
005C   6F              MOV     L,A     ; PUT LSP OF ADDRESS OF TABLE ENTRY INTO L
005D   E9              PCHL            ; NEXT INSTRUCTION COMES FROM COMMAND ROUTINE
                 ;
                 ;
                 ;*****************************************************************
                 ;
                 ;
                 ;            COMMAND IMPLEMENTING ROUTINES
                 ;
                 ;
                 ;*****************************************************************
                 ;
                 ;
                 ; FUNCTION: DCMD
                 ; INPUTS: NONE
                 ; OUTPUTS: NONE
                 ; CALLS: ECHO,NMOUT,HILO,GETCM,CROUT,GETNM
                 ; DESTROYS: A,B,C,D,E,H,L,F/F'S
                 ; DESCRIPTION: DCMD IMPLEMENTS THE DISPLAY MEMORY (D) COMMAND
                 ;
005E             DCMD:
005E   0E02          MVI     C,2     ; GET 2 NUMBERS FROM INPUT STREAM
0060   CD5702        CALL    GETNM
0063   D1            POP     D       ; ENDING ADDRESS TO DE
0064   E1            POP     H       ; STARTING ADDRESS TO HL
0065             DCM05:
0065   CDEE01        CALL    CROUT   ; ECHO CARRIAGE RETURN/LINE FEED
0068   7C            MOV     A,H     ; DISPLAY ADDRESS OF FIRST LOCATION IN LINE
0069   CDC302        CALL    NMOUT
006C   7D            MOV     A,L     ; ADDRESS IS 2 BYTES LONG
006D   CDC302        CALL    NMOUT
0070             DCM10:
0070   0E20          MVI     C,' '
0072   CDF401        CALL    ECHO    ; USE BLANK AS SEPARATOR
0075   7E            MOV     A,M     ; GET CONTENTS OF NEXT MEMORY LOCATION
0076   CDC302        CALL    NMOUT   ; DISPLAY CONTENTS
```

```
0079   CDBD01          CALL    BREAK   ; SEE IF USER WANTS OUT
     1          +       TRUE    DCM12   ; IF SO, BRANCH
007C 1 DA8500    +       JC      DCM12
007F   CD9C02          CALL    HILO    ; SEE IF ADDRESS OF DISPLAYED LOCATION IS
                                       ; /GREATER THAN OR EQUAL TO ENDING ADDRESS
     1          +       FALSE   DCM15   ; IF NOT, MORE TO DISPLAY
0082 1 D28B00    +       JNC     DCM15
0085            DCM12:
0085   CDEE01          CALL    CROUT   ; CARRIAGE RETURN/LINE FEED TO END LINE
0088   C32B00          JMP     GETCM   ; ALL DONE
008B            DCM15:
008B   23              INX     H       ; IF MORE TO GO, POINT TO NEXT LOC TO DISPLAY
008C   7D              MOV     A,L     ; GET LOW ORDER BITS OF NEW ADDRESS
008D   E60F            ANI     NEWLN   ; SEE IF LAST HEX DIGIT OF ADDRESS DENOTES
                                       ; /START OF NEW LINE
008F   C27000          JNZ     DCM10   ; NO - NOT AT END OF LINE
0092   C36500          JMP     DCM05   ; YES - START NEW LINE WITH ADDRESS
                       ;
                       ;
                       ;*****************************************************************
                       ;
                       ;
                       ; FUNCTION: GCMD
                       ; INPUTS: NONE
                       ; OUTPUTS: NONE
                       ; CALLS: ERROR,GETHX,RSTTF
                       ; DESTROYS: A,B,C,D,E,H,L,F/F'S
                       ; DESCRIPTION: GCMD IMPLEMENTS THE BEGIN EXECUTION (G) COMMAND.
                       ;
0095                   GCMD:
0095   CD2202          CALL    GETHX   ; GET ADDRESS (IF PRESENT) FROM INPUT STREAM
     1          +       FALSE   GCM05   ; BRANCH IF NO NUMBER PRESENT
0098 1 D2AA00    +       JNC     GCM05
009B   7A              MOV     A,D     ; ELSE, GET TERMINATOR
009C   FE0D            CPI     CR      ; SEE IF CARRIAGE RETURN
009E   C20D02          JNZ     ERROR   ; ERROR IF NOT PROPERLY TERMINATED
00A1   21F513          LXI     H,PSAVE ; WANT NUMBER TO REPLACE SAVE PGM COUNTER
00A4   71              MOV     M,C
00A5   23              INX     H
00A6   70              MOV     M,B
00A7   C3B000          JMP     GCM10
00AA            GCM05:
00AA   7A              MOV     A,D     ; IF NO STARTING ADDRESS, MAKE SURE THAT
00AB   FE0D            CPI     CR      ; /CARRIAGE RETURN TERMINATED COMMAND
00AD   C20D02          JNZ     ERROR   ; ERROR IF NOT
00B0            GCM10:
00B0   C32E03          JMP     RSTTF   ; RESTORE REGISTERS AND BEGIN EXECUTION

                       ;
                       ;
                       ;*****************************************************************
```

```
                        ;
                        ;
                        ; FUNCTION: ICMD
                        ; INPUTS: NONE
                        ; OUTPUTS: NONE
                        ; CALLS: ERROR,ECHO,GETCH,VALDL,VALDG,CNVBN,STHLF,GETNM,CROUT
                        ; DESTROYS: A,B,C,D,E,H,L,F/F'S
                        ; DESCRIPTION: ICMD IMPLEMENTS THE INSERT CODE INTO MEMORY (I) COMMAND
                        ;
00B3                    ICMD:
00B3   0E01                     MVI     C,1
00B5   CD5702                   CALL    GETNM   ; GET SINGLE NUMBER FROM INPUT STREAM
00B8   3EFF                     MVI     A,UPPER
00BA   32F913                   STA     TEMP    ; TEMP WILL HOLD THE UPPER/LOWER HALF BYTE FLAG
00BD   D1                       POP     D       ; ADDRESS OF START TO DE
00BE                    ICM05:
00BE   CD1B02                   CALL    GETCH   ; GET A CHARACTER FROM INPUT STREAM
00C1   4F                       MOV     C,A
00C2   CDF401                   CALL    ECHO    ; ECHO IT
00C5   79                       MOV     A,C     ; PUT CHARACTER BACK INTO A
00C6   FE1B                     CPI     TERM    ; SEE IF CHARACTER IS A TERMINATING CHARACTER
00C8   CAF400                   JZ      ICM25   ; IF SO, ALL DONE ENTERING CHARACTERS
00CB   CD8A03                   CALL    VALDL   ; ELSE, SEE IF VALID DELIMITER
     1                  +       TRUE    ICM05   ; IF SO SIMPLY IGNORE THIS CHARACTER
00CE 1 DABE00           +       JC      ICM05
00D1   CD6F03                   CALL    VALDG   ; ELSE, CHECK TO SEE IF VALID HEX DIGIT
     1                  +       FALSE   ICM20   ; IF NOT, BRANCH TO HANDLE ERROR CONDITION
00D4 1 D2EE00           +       JNC     ICM20
00D7   CDDA01                   CALL    CNVBN   ; CONVERT DIGIT TO BINARY
00DA   4F                       MOV     C,A     ; MOVE RESULT TO C
00DB   CD5003                   CALL    STHLF   ; STORE IN APPROPRIATE HALF WORD
00DE   3AF913                   LDA     TEMP    ; GET HALF BYTE FLAG
00E1   B7                       ORA     A       ; SET F/F'S
00E2   C2E600                   JNZ     ICM10   ; BRANCH IF FLAG SET FOR UPPER
00E5   13                       INX     D       ; IF LOWER, INC ADDRESS OF BYTE TO STORE IN
00E6                    ICM10:
00E6   EEFF                     XRI     INVRT   ; TOGGLE STATE OF FLAG
00E8   32F913                   STA     TEMP    ; PUT NEW VALUE OF FLAG BACK
00EB   C3BE00                   JMP     ICM05   ; PROCESS NEXT DIGIT
00EE                    ICM20:
00EE   CD4503                   CALL    STHF0   ; ILLEGAL CHARACTER
00F1   C30D02                   JMP     ERROR   ; MAKE SURE ENTIRE BYTE FILLED THEN ERROR
00F4                    ICM25:
00F4   CD4503                   CALL    STHF0   ; HERE FOR ESCAPE CHARACTER - INPUT IS DONE
00F7   CDEE01                   CALL    CROUT   ; ADD CARRIAGE RETURN
00FA   C32B00                   JMP     GETCM
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ;
```

```
                        ; FUNCTION: MCMD
                        ; INPUTS: NONE
                        ; OUTPUTS: NONE
                        ; CALLS: GETCM,HILO,GETNM
                        ; DESTROYS: A,B,C,D,E,H,L,F/F'S
                        ; DESCRIPTION: MCMD IMPLEMENTS THE MOVE DATA IN MEMORY (M) COMMAND.
                        ;
00FD                    MCMD:
00FD  0E03                    MVI     C,3
00FF  CD5702                  CALL    GETNM   ; GET 3 NUMBERS FROM INPUT STREAM
0102  C1                      POP     B       ; DESTINATION ADDRESS TO BC
0103  E1                      POP     H       ; ENDING ADDRESS TO HL
0104  D1                      POP     D       ; STARTING ADDRESS TO DE
0105                    MCM05:
0105  E5                      PUSH    H       ; SAVE ENDING ADDRESS
0106  62                      MOV     H,D
0107  6B                      MOV     L,E     ; SOURCE ADDRESS TO HL
0108  7E                      MOV     A,M     ; GET SOURCE BYTE
0109  60                      MOV     H,B
010A  69                      MOV     L,C     ; DESTINATION ADDRESS TO HL
010B  77                      MOV     M,A     ; MOVE BYTE TO DESTINATION
010C  03                      INX     B       ; INCREMENT DESTINATION ADDRESS
010D  78                      MOV     A,B
010E  B1                      ORA     C       ; TEST FOR DESTINATION ADDRESS OVERFLOW
010F  CA2B00                  JZ      GETCM   ; IF SO, CAN TERMINATE COMMAND
0112  13                      INX     D       ; INCREMENT SOURCE ADDRESS
0113  E1                      POP     H       ; ELSE, GET BACK ENDING ADDRESS
0114  CD9C02                  CALL    HILO    ; SEE IF ENDING ADDR>=SOURCE ADDR
    1                   +     FALSE   GETCM   ; IF NOT, COMMAND IS DONE
0117 1 D22B00           +     JNC     GETCM
011A  C30501                  JMP     MCM05   ; MOVE ANOTHER BYTE
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ;
                        ; FUNCTION: SCMD
                        ; INPUTS: NONE
                        ; OUTPUTS: NONE
                        ; CALLS: GETHX,GETCM,NMOUT,ECHO
                        ; DESTROYS: A,B,C,D,E,H,L,F/F'S
                        ; DESCRIPTION: SCMD IMPLEMENTS THE SUBSTITUTE INTO MEMORY (S) COMMAND.
                        ;
011D                    SCMD:
011D  CD2202                  CALL    GETHX   ; GET A NUMBER, IF PRESENT, FROM INPUT
0120  C5                      PUSH    B
0121  E1                      POP     H       ; GET NUMBER TO HL - DENOTES MEMORY LOCATION
0122                    SCM05:
0122  7A                      MOV     A,D     ; GET TERMINATOR
0123  FE20                    CPI             ; SEE IF SPACE
0125  CA2D01                  JZ      SCM10   ; YES - CONTINUE PROCESSING
```

```
0128   FE2C            CPI     ','     ; ELSE, SEE IF COMMA
012A   C22B00          JNZ     GETCM   ; NO - TERMINATE COMMAND
012D                SCM10:
012D   7E              MOV     A,M     ; GET CONTENTS OF SPECIFIED LOCATION TO A
012E   CDC302          CALL    NMOUT   ; DISPLAY CONTENTS ON CONSOLE
0131   0E2D            MVI     C,'-'
0133   CDF401          CALL    ECHO    ; USE DASH FOR SEPARATOR
0136   CD2202          CALL    GETHX   ; GET NEW VALUE FOR MEMORY LOCATION, IF ANY
    1              +    FALSE   SCM15   ; IF NO VALUE PRESENT, BRANCH
0139 1 D23D01      +    JNC     SCM15
013C   71              MOV     M,C     ; ELSE, STORE LOWER 8 BITS OF NUMBER ENTERED
013D                SCM15:
013D   23              INX     H       ; INCREMENT ADDRESS OF MEMORY LOCATION TO VIEW
013E   C32201          JMP     SCM05
                    ;
                    ;
                    ;*****************************************************************
                    ;
                    ;
                    ; FUNCTION: XCMD
                    ; INPUTS: NONE
                    ; OUTPUTS: NONE
                    ; CALLS: GETCH,ECHO,REGDS,GETCM,ERROR,RGADR,NMOUT,CROUT,GETHX
                    ; DESTROYS: A,B,C,D,E,H,L,F/F'S
                    ; DESCRIPTION: XCMD IMPLEMENTS THE REGISTER EXAMINE AND CHANGE (X)
                    ;                COMMAND.
                    ;
0141                XCMD:
0141   CD1B02          CALL    GETCH   ; GET REGISTER IDENTIFIER
0144   4F              MOV     C,A
0145   CDF401          CALL    ECHO    ; ECHO IT
0148   79              MOV     A,C
0149   FE0D            CPI     CR
014B   C25401          JNZ     XCM05   ; BRANCH IF NOT CARRIAGE RETURN
014E   CDE602          CALL    REGDS   ; ELSE, DISPLAY REGISTER CONTENTS
0151   C32B00          JMP     GETCM   ; THEN TERMINATE COMMAND
0154                XCM05:
0154   4F              MOV     C,A     ; GET REGISTER IDENTIFIER TO C
0155   CD1703          CALL    RGADR   ; CONVERT IDENTIFIER INTO RTAB TABLE ADDR
0158   C5              PUSH    B
0159   E1              POP     H       ; PUT POINTER TO REGISTER ENTRY INTO HL
015A   0E20            MVI     C,' '
015C   CDF401          CALL    ECHO    ; ECHO SPACE TO USER
015F   79              MOV     A,C
0160   32F913          STA     TEMP    ; PUT SPACE INTO TEMP AS DELIMITER
0163                XCM10:
0163   3AF913          LDA     TEMP    ; GET TERMINATOR
0166   FE20            CPI     ' '     ; SEE IF A BLANK
0168   CA7001          JZ      XCM15   ; YES - GO CHECK POINTER INTO TABLE
016B   FE2C            CPI     ','     ; NO - SEE IF COMMA
016D   C22B00          JNZ     GETCM   ; NO - MUST BE CARRIAGE RETURN TO END COMMAND
```

```
0170                XCM15:
0170   7E                   MOV     A,M
0171   B7                   ORA     A         ; SET F/F'S
0172   C27B01               JNZ     XCM18     ; BRANCH IF NOT AT END OF TABLE
0175   CDEE01               CALL    CROUT     ; ELSE, OUTPUT CARRIAGE RETURN LINE FEED
0178   C32B00               JMP     GETCM     ; AND EXIT
017B                XCM18:
017B   E5                   PUSH    H         ; PUT POINTER ON STACK
017C   5E                   MOV     E,M
017D   1613                 MVI     D,DATA SHR 8    ; FETCH ADDRESS OF SAVE LOCATION FROM TABI
017F   23                   INX     H
0180   46                   MOV     B,M       ; FETCH LENGTH FLAG FROM TABLE
0181   D5                   PUSH    D         ; SAVE ADDRESS OF SAVE LOCATION
0182   D5                   PUSH    D
0183   E1                   POP     H         ; MOVE ADDRESS TO HL
0184   C5                   PUSH    B         ; SAVE LENGTH FLAG
0185   7E                   MOV     A,M       ; GET 8 BITS OF REGISTER FROM SAVE LOCATION
0186   CDC302               CALL    NMOUT     ; DISPLAY IT
0189   F1                   POP     PSW       ; GET BACK LENGTH FLAG
018A   F5                   PUSH    PSW       ; SAVE IT AGAIN
018B   B7                   ORA     A         ; SET F/F'S
018C   CA9401               JZ      XCM20     ; IF 8 BIT REGISTER, NOTHING MORE TO DISPLAY
018F   2B                   DCX     H         ; ELSE, FOR 16 BIT REGISTER, GET LOWER 8 BITS
0190   7E                   MOV     A,M
0191   CDC302               CALL    NMOUT     ; DISPLAY THEM
0194                XCM20:
0194   0E2D                 MVI     C,'-'
0196   CDF401               CALL    ECHO      ; USE DASH AS SEPARATOR
0199   CD2202               CALL    GETHX     ; SEE IF THERE IS A VALUE TO PUT INTO REGISTER
     1            +         FALSE   XCM30     ; NO - GO CHECK FOR NEXT REGISTER
019C 1 D2B401     +         JNC     XCM30
019F   7A                   MOV     A,D
01A0   32F913               STA     TEMP      ; ELSE, SAVE THE TERMINATOR FOR NOW
01A3   F1                   POP     PSW       ; GET BACK LENGTH FLAG
01A4   E1                   POP     H         ; PUT ADDRESS OF SAVE LOCATION INTO HL
01A5   B7                   ORA     A         ; SET F/F'S
01A6   CAAB01               JZ      XCM25     ; IF 8 BIT REGISTER, BRANCH
01A9   70                   MOV     M,B       ; SAVE UPPER 8 BITS
01AA   2B                   DCX     H         ; POINT TO SAVE LOCATION FOR LOWER 8 BITS
01AB                XCM25:
01AB   71                   MOV     M,C       ; STORE ALL OF 8 BIT OR LOWER 1/2 OF 16 BIT REG
01AC                XCM27:
01AC   110300               LXI     D,RTABS   ; SIZE OF ENTRY IN RTAB TABLE
01AF   E1                   POP     H         ; POINTER INTO REGISTER TABLE RTAB
01B0   19                   DAD     D         ; ADD ENTRY SIZE TO POINTER
01B1   C36301               JMP     XCM10     ; DO NEXT REGISTER
01B4                XCM30:
01B4   7A                   MOV     A,D       ; GET TERMINATOR
01B5   32F913               STA     TEMP      ; SAVE IN MEMORY
01B8   D1                   POP     D         ; CLEAR STACK OF LENGTH FLAG AND ADDRESS
01B9   D1                   POP     D         ; /OF SAVE LOCATION
```

```
  01BA   C3AC01          JMP     XCM27   ; GO INCREMENT REGISTER TABLE POINTER
                 ;
                 ;
                 ;*****************************************************************
                 ;
                 ;
                 ;                   UTILITY ROUTINES
                 ;
                 ;
                 ;*****************************************************************
                 ;
                 ;
                 ; FUNCTION: BREAK
                 ; INPUTS: NONE
CTER INPUT       ; OUTPUTS: CARRY - 1 IF ESCAPE CHARA
                 ;                - 0 IF ANY OTHER CHARACTER OR NO CHARACTER PENDING
                 ; CALLS: NOTHING
                 ; DESTROYS: A,F/F'S
                 ; DESCRIPTION: BREAK IS USED TO SENSE AN ESCAPE CHARACTER FROM
                 ;              THE USER.  IF NO CHARACTER IS PENDING, OR IF THE
                 ;              PENDING CHARACTER IS NOT THE ESCAPE, THEN A FAILURE
                 ;              RETURN (CARRY=0) IS TAKEN.  IN THIS CASE,
                                                                      THE
                 ;              PENDING CHARACTER (IF ANY) IS LOST.  IF THE PENDING
                 ;              CHARACTER IS AN ESCAPE CHARACTER, BREAK TAKES A SUCCESS
                 ;              RETURN (CARRY=1).
                 ;
  01BD           BREAK:
  01BD   DBFB          IN      CONST   ; GET CONSOLE STATUS
  01BF   E602          ANI     RBR     ; SEE IF CHARACTER PENDING
  01C1   CA1802        JZ      FRET    ; NO - TAKE FAILURE RETURN
  01C4   DBFA          IN      CNIN    ; YES - PICK UP CHARACTER
  01C6   E67F          ANI     PRTY0   ; STRIP OFF PARITY BIT
  01C8   FE1B          CPI     BRCHR   ; SEE IF BREAK CHARACTER
  01CA   CA4303        JZ      SRET    ; YES - SUCCESS RETURN
  01CD   C31802        JMP     FRET    ; NO - FAILURE RETURN - CHARACTER LOST
                 ;
                 ;
                 ;*****************************************************************
                 ;
                 ;
                 ; FUNCTION: CI
                 ; INPUTS: NONE
                 ; OUTPUTS: A - CHARACTER FROM CONSOLE
                 ; CALLS: NOTHING
                 ; DESTROYS: A,F/F'S
                 ; DESCRIPTION: CI WAITS UNTIL A CHARACTER HAS BEEN ENTERED AT THE
                 ;              CONSOLE AND THEN RETURNS THE CHARACTER, VIA THE A
                 ;              REGISSTER, TO THE CALLING ROUTINE.  THIS ROUTINE
                 ;              IS CALLED BY THE USER VIA A JUMP TABLE IN RAM.
                 ;
  01D0           CI:
```

```
01D0   DBFB          IN      CONST   ; GET STATUS OF CONSOLE
01D2   E602          ANI     RBR     ; CHECK FOR RECEIVER BUFFER READY
01D4   CAD001        JZ      CI      ; NOT YET - WAIT
01D7   DBFA          IN      CNIN    ; READY SO GET CHARACTER
01D9   C9            RET
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION: CNVBN
                ; INPUTS: C - ASCII CHARACTER '0'-'9' OR 'A'-'F'
                ; OUTPUTS: A - 0 TO F HEX
                ; CALLS: NOTHING
                ; DESTROYS: A,F/F'S
                ; DESCRIPTION: CNVBN CONVERTS THE ASCII REPRESENTATION OF A HEX
                ;              CNVBN INTO ITS CORRESPONDING BINARY VALUE.  CNVBN
                ;              DOES NOT CHECK THE VALIDITY OF ITS INPUT.
                ;
01DA            CNVBN:
01DA   79            MOV     A,C
01DB   D630          SUI     '0'     ; SUBTRACT CODE FOR '0' FROM ARGUMENT
01DD   FE0A          CPI     10      ; WANT TO TEST FOR RESULT OF 0 TO 9
01DF   F8            RM              ; IF SO, THEN ALL DONE
01E0   D607          SUI     7       ; ELSE, RESULT BETWEEN 17 AND 23 DECIMAL
01E2   C9            RET             ; SO RETURN AFTER SUBTRACTING BIAS OF 7
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION: CO
                ; INPUTS: C - CHARACTER TO OUTPUT TO CONSOLE
                ; OUTPUTS: C - CHARACTER OUTPUT TO CONSOLE
                ; CALLS: NOTHING
                ; DESTROYS: A,F/F'S
                ; DESCRIPTION: CO WAITS UNTIL THE CONSOLE IS READY TO ACCEPT A CHARACTER
                ;              AND THEN SENDS THE INPUT ARGUMENT TO THE CONSOLE.
                ;
01E3            CO:
01E3   DBFB          IN      CONST   ; GET STATUS OF CONSOLE
01E5   E601          ANI     TRDY    ; SEE IF TRANSMITTER READY
01E7   CAE301        JZ      CO      ; NO - WAIT
01EA   79            MOV     A,C     ; ELSE, MOVE CHARACTER TO A REGISTER FOR OUTPUT
01EB   D3FA          OUT     CNOUT   ; SEND TO CONSOLE
01ED   C9            RET
                ;
                ;
                ;*****************************************************************
                ;
                ;
```

```
                        ; FUNCTION CROUT
                        ; INPUTS: NONE
                        ; OUTPUTS: NONE
                        ; CALLS: ECHO
                        ; DESTROYS: A,B,C,F/F'S
                        ; DESCRIPTION: CROUT SENDS A CARRIAGE RETURN (AND HENCE A LINE
                        ;              FEED) TO THE CONSOLE.
                        ;
  01EE                  CROUT:
  01EE   0E0D                 MVI     C,CR
  01F0   CDF401               CALL    ECHO
  01F3   C9                   RET
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ;
                        ; FUNCTION: ECHO
                        ; INPUTS: C - CHARACTER TO ECHO TO TERMINAL
                        ; OUTPUTS: C - CHARACTER ECHOED TO TERMINAL
                        ; CALLS: CO
                        ; DESTROYS: A,B,F/F'S
                        ; DESCRIPTION: ECHO TAKES A SINGLE CHARACTER AS INPUT AND, VIA
                        ;              THE MONITOR, SENDS THAT CHARACTER TO THE USER
                        ;              TERMINAL.  A CARRIAGE RETURN IS ECHOED AS A CARRIAGE
                        ;              RETURN LINE FEED, AND AN ESCAPE CHARACTER IS ECHOED AS
                        ;
  01F4                  ECHO:
  01F4   41                   MOV     B,C     ; SAVE ARGUMENT
  01F5   3E1B                 MVI     A,ESC
  01F7   B8                   CMP     B       ; SEE IF ECHOING AN ESCAPE CHARACTER
  01F8   C2FD01               JNZ     ECH05   ; NO - BRANCH
  01FB   0E24                 MVI     C,'$'   ; YES - ECHO AS $
  01FD                  ECH05:
  01FD   CDE301               CALL    CO      ; DO OUTPUT THROUGH MONITOR
  0200   3E0D                 MVI     A,CR
  0202   B8                   CMP     B       ; SEE IF CHARACTER ECHOED WAS A CARRIAGE RETURN
  0203   C20B02               JNZ     ECH10   ; NO - NO NEED TO TAKE SPECIAL ACTION
  0206   0E0A                 MVI     C,LF    ; YES - WANT TO ECHO LINE FEED, TOO
  0208   CDE301               CALL    CO
  020B                  ECH10:
  020B   48                   MOV     C,B     ; RESTORE ARGUMENT
  020C   C9                   RET
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ;
                        ; FUNCTION: ERROR
                        ; INPUTS: NONE
                        ; OUTPUTS: NONE
```

```
                        ; CALLS: ECHO,CROUT,GETCM
                        ; DESTROYS: A,B,C,F/F'S
                        ; DESCRIPTION: ERROR PRINTS THE ERROR CHARACTER (CURRENTLY AN ASTERISK)
                        ;              ON THE CONSOLE, FOLLOWED BY A CARRIAGE RETURN-LINE FEED,
                        ;              AND THEN RETURNS CONTROL TO THE COMMAND RECOGNIZER.
                        ;
020D                    ERROR:
020D   0E2A                   MVI     C,'*'
020F   CDF401                 CALL    ECHO    ; SEND * TO CONSOLE
0212   CDEF01                 CALL    CROUT   ; SKIP TO BEGINNING OF NEXT LINE
0215   C32B00                 JMP     GETCM   ; TRY AGAIN FOR ANOTHER COMMAND
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ;
                        ; FUNCTION: FRET
                        ; INPUTS: NONE
                        ; OUTPUTS: CARRY - ALWAYS 0
                        ; CALLS: NOTHING
                        ; DESTROYS: CARRY
                        ; DESCRIPTION: FRET IS JUMPED TO BY ANY ROUTINE THAT WISHES TO
                        ;              INDICATE FAILURE ON RETURN.  FRET SETS THE CARRY
                        ;              FALSE, DENOTING FAILURE, AND THEN RETURNS TO THE
                        ;              CALLER OF THE ROUTINE INVOKING FRET.
                        ;
0218                    FRET:
0218   37                     STC             ; FIRST SET CARRY TRUE
0219   3F                     CMC             ; THEN COMPLEMENT IT TO MAKE IT FALSE
021A   C9                     RET             ; RETURN APPROPRIATELY
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ;
                        ; FUNCTION: GETCH
                        ; INPUTS: NONE
                        ; OUTPUTS: C - NEXT CHARACTER IN INPUT STREAM
                        ; CALLS: CI
                        ; DESTROYS: A,C,F/F'S
                        ; DESCRIPTION: GETCH RETURNS THE NEXT CHARACTER IN THE INPUT STREAM
                        ;              TO THE CALLING PROGRAM.
                        ;
021B                    GETCH:
021B   CDD001                 CALL    CI      ; GET CHARACTER FROM TERMINAL
021E   E67F                   ANI     PRTY0   ; TURN OFF PARITY BIT IN CASE SET BY CONSOLE
0220   4F                     MOV     C,A     ; PUT VALUE IN C REGISTER FOR RETURN
0221   C9                     RET
                        ;
                        ;
                        ;*****************************************************************
```

```
                        ;
                        ;
                        ; FUNCTION: GETHX
                        ; INPUTS: NONE
                        ; OUTPUTS: BC - 16 BIT INTEGER
                        ;          D - CHARACTER WHICH TERMINATED THE INTEGER
                        ;          CARRY - 1 IF FIRST CHARACTER NOT DELIMITER
                        ;                - 0 IF FIRST CHARACTER IS DELIMITER
                        ; CALLS: GETCH,ECHO,VALDL,VALDG,CNVBN,ERROR
                        ; DESTROYS: A,B,C,D,E,F/F'S
                        ; DESCRIPTION: GETHX ACCEPTS A STRING OF HEX DIGITS FROM THE INPUT
                        ;              STREAM AND RETURNS THEIR VALUE AS A 16 BIT BINARY
                        ;              INTEGER.  IF MORE THAN 4 HEX DIGITS ARE ENTERED,
                        ;              ONLY THE LAST 4 ARE USED.  THE NUMBER TERMINATES WHEN
                        ;              A VALID DELIMITER IS ENCOUNTERED.  THE DELIMITER IS
                        ;              ALSO RETURNED AS AN OUTPUT OF THE FUNCTION.  ILLEGAL
                        ;              CHARACTERS (NOT HEX DIGITS OR DELIMITERS) CAUSE AN
                        ;              ERROR INDICATION.  IF THE FIRST (VALID) CHARACTER
                        ;              ENCOUNTERED IN THE INPUT STREAM IS NOT A DELIMITER,
                        ;              GETHX WILL RETURN WITH THE CARRY BIT SET TO 1;
                        ;              OTHERWISE, THE CARRY BIT IS SET TO 0 AND THE CONTENTS
                        ;              OF BC ARE UNDEFINED.
                        ;
0222                    GETHX:
0222   E5               PUSH    H       ; SAVE HL
0223   210000           LXI     H,0     ; INITIALIZE RESULT
0226   1E00             MVI     E,0     ; INITIALIZE DIGIT FLAG TO FALSE
0228                    GHX05:
0228   CD1B02           CALL    GETCH   ; GET A CHARACTER
022B   4F               MOV     C,A
022C   CDF401           CALL    ECHO    ; ECHO THE CHARACTER
022F   CD8A03           CALL    VALDL   ; SEE IF DELIMITER
     1                  +       FALSE   GHX10   ; NO - BRANCH
0232 1 D24102           +       JNC     GHX10
0235   51               MOV     D,C     ; YES - ALL DONE, BUT WANT TO RETURN DELIMITER
0236   E5               PUSH    H
0237   C1               POP     B       ; MOVE RESULT TO BC
0238   E1               POP     H       ; RESTORE HL
0239   7B               MOV     A,E     ; GET FLAG
023A   B7               ORA     A       ; SET F/F'S
023B   C24303           JNZ     SRET    ; IF FLAG NON-0, A NUMBER HAS BEEN FOUND
023E   CA1802           JZ      FRET    ; ELSE, DELIMITER WAS FIRST CHARACTER
0241                    GHX10:
0241   CD6F03           CALL    VALDG   ; IF NOT DELIMITER, SEE IF DIGIT
     1                  +       FALSE   ERROR   ; ERROR IF NOT A VALID DIGIT, EITHER
0244 1 D20D02           +       JNC     ERROR
0247   CDDA01           CALL    CNVBN   ; CONVERT DIGIT TO ITS BINARY VALUE
024A   1EFF             MVI     E,0FFH  ; SET DIGIT FLAG NON-0
024C   29               DAD     H       ; *2
024D   29               DAD     H       ; *4
024E   29               DAD     H       ; *8
```

```
024F   29              DAD     H       ; *16
0250   0600            MVI     B,0     ; CLEAR UPPER 8 BITS OF BC PAIR
0252   4F              MOV     C,A     ; BINARY VALUE OF CHARACTER INTO C
0253   09              DAD     B       ; ADD THIS VALUE TO PARTIAL RESULT
0254   C32802          JMP     GHX05   ; GET NEXT CHARACTER
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION: GETNM
                ; INPUTS: C - COUNT OF NUMBERS TO FIND IN INPUT STREAM
                ; OUTPUTS: TOP OF STACK - NUMBERS FOUND IN REVERSE ORDER (LAST ON TOP
                ;                         OF STACK)
                ; CALLS: GETHX,HILO,ERROR
                ; DESTROYS: A,B,C,D,E,H,L,F/F'S
                ; DESCRIPTION: GETNM FINDS A SPECIFIED COUNT OF NUMBERS, BETWEEN 1
                ;              AND 3, INCLUSIVE,  IN THE INPUT
                ;              STREAM AND RETURNS THEIR VALUES ON THE STACK.  IF 2
                ;              OR MORE NUMBERS ARE REQUESTED, THEN THE FIRST MUST BE
                ;              LESS THAN OR EQUAL TO THE SECOND, OR THE FIRST AND
                ;              SECOND NUMBERS WILL BE SET EQUAL.  THE LAST NUMBER
                ;              REQUESTED MUST BE TERMINATED BY A CARRIAGE RETURN
                ;              OR AN ERROR INDICATION WILL RESULT.
                ;
0257            GETNM:
0257   2E03            MVI     L,3     ; PUT MAXIMUM ARGUMENT COUNT INTO L
0259   79              MOV     A,C     ; GET THE ACTUAL ARGUMENT COUNT
025A   E603            ANI     3       ; FORCE TO MAXIMUM OF 3
025C   C8              RZ              ; IF 0, DON'T BOTHER TO DO ANYTHIING
025D   67              MOV     H,A     ; ELSE, PUT ACTUAL COUNT INTO H
025E            GNM05:
025E   CD2202          CALL    GETHX   ; GET A NUMBER FROM INPUT STREAM
     1          +       FALSE   ERROR   ; ERROR IF NOT THERE - TOO FEW NUMBERS
0261 1 D20D02  +       JNC     ERROR
0264   C5              PUSH    B       ; ELSE, SAVE NUMBER ON STACK
0265   2D              DCR     L       ; DECREMENT MAXIMUM ARGUMENT COUNT
0266   25              DCR     H       ; DECREMENT ACTUAL ARGUMENT COUNT
0267   CA7302          JZ      GNM10   ; BRANCH IF NO MORE NUMBERS WANTED
026A   7A              MOV     A,D     ; ELSE, GET NUMBER TERMINATOR TO A
026B   FE0D            CPI     CR      ; SEE IF CARRIAGE RETURN
026D   CA0D02          JZ      ERROR   ; ERROR IF SO - TOO FEW NUMBERS
0270   C35E02          JMP     GNM05   ; ELSE, PROCESS NEXT NUMBER
0273            GNM10:
0273   7A              MOV     A,D     ; WHEN COUNT 0, CHECK LAST TERMINATOR
0274   FE0D            CPI     CR
0276   C20D02          JNZ     ERROR   ; ERROR IF NOT CARRIAGE RETURN
0279   01FFFF          LXI     B,0FFFFH        ; HL GETS LARGEST NUMBER
027C   7D              MOV     A,L     ; GET WHAT'S LEFT OF MAXIMUM ARG COUNT
027D   B7              ORA     A       ; CHECK FOR 0
027E   CA8602          JZ      GNM20   ; IF YES, 3 NUMBERS WERE INPUT
```

```
0281                GNM15:
0281   C5           	PUSH    B       ; IF NOT, FILL REMAINING ARGUMENTS WITH 0FFFFH
0282   2D           	DCR     L
0283   C28102       	JNZ     GNM15
0286                GNM20:
0286   C1           	POP     B       ; GET THE 3 ARGUMENTS OUT
0287   D1           	POP     D
0288   E1           	POP     H
0289   CD9C02       	CALL    HILO    ; SEE IF FIRST >= SECOND
     1             +    FALSE   GNM25   ; NO - BRANCH
028C 1 D29102       +    JNC     GNM25
028F   54           	MOV     D,H
0290   5D           	MOV     E,L     ; YES - MAKE SECOND EQUAL TO THE FIRST
0291                GNM25:
0291   E3           	XTHL            ; PUT FIRST ON STACK - GET RETURN ADDR
0292   D5           	PUSH    D       ; PUT SECOND ON STACK
0293   C5           	PUSH    B       ; PUT THIRD ON STACK
0294   E5           	PUSH    H       ; PUT RETURN ADDRESS ON STACK
0295                GNM30:
0295   3D           	DCR     A       ; DECREMENT RESIDUAL COUNT
0296   F8           	RM              ; IF NEGATIVE, PROPER RESULTS ON STACK
0297   E1           	POP     H       ; ELSE, GET RETURN ADDR
0298   E3           	XTHL            ; REPLACE TOP RESULT WITH RETURN ADDR
0299   C39502       	JMP     GNM30   ; TRY AGAIN
                    ;
                    ;
                    ;*****************************************************************
                    ;
                    ;
                    ; FUNCTION: HILO
                    ; INPUTS: DE - 16 BIT INTEGER
                    ;         HL - 16 BIT INTEGER
                    ; OUTPUTS: CARRY - 0 IF HL<DE
                    ;                - 1 IF HL>=DE
                    ; CALLS: NOTHING
                    ; DESTROYS: F/F'S
                    ; DESCRIPTION: HILO COMPARES THE 2 16 BIT INTEGERS IN HL AND DE.  THE
                    ;               INTEGERS ARE TREATED AS UNSIGNED NUMBERS.  THE CARRY
                    ;               BIT IS SET ACCORDING TO THE RESULT OF THE COMPARISON.
                    ;
029C                HILO:
029C   C5           	PUSH    B       ; SAVE BC
029D   47           	MOV     B,A     ; SAVE A IN B REGISTER
029E   E5           	PUSH    H       ; SAVE HL PAIR
029F   7A           	MOV     A,D     ; CHECK FOR DE = 0000H
02A0   B3           	ORA     E
02A1   CABD02       	JZ      HIL05   ; WE'RE AUTOMATICALLY DONE IF IT IS
02A4   23           	INX     H       ; INCREMENT HL BY 1
02A5   7C           	MOV     A,H     ; WANT TO TEST FOR 0 RESULT AFTER
02A6   B5           	ORA     L       ; /INCREMENTING
02A7   CABD02       	JZ      HIL05   ; IF SO, HL MUST HAVE CONTAINED 0FFFFH
```

```
02AA   E1            POP     H       ; IF NOT, RESTORE ORIGINAL HL
02AB   D5            PUSH    D       ; SAVE DE
02AC   3EFF          MVI     A,0FFH  ; WANT TO TAKE 2'S COMPLEMENT OF DE CONTENTS
02AE   AA            XRA     D
02AF   57            MOV     D,A
02B0   3EFF          MVI     A,0FFH
02B2   AB            XRA     E
02B3   5F            MOV     E,A
02B4   13            INX     D       ; 2'S COMPLEMENT OF DE TO DE
02B5   7D            MOV     A,L
02B6   83            ADD     E       ; ADD HL AND DE
02B7   7C            MOV     A,H
02B8   8A            ADC     D       ; THIS OPERATION SETS CARRY PROPERLY
02B9   D1            POP     D       ; RESTORE ORIGINAL DE CONTENTS
02BA   78            MOV     A,B     ; RESTORE ORIGINAL CONTENTS OF A
02BB   C1            POP     B       ; RESTORE ORIGINAL CONTENTS OF BC
02BC   C9            RET             ; RETURN WITH CARRY SET AS REQUIRED
02BD             HIL05:
02BD   E1            POP     H       ; IF HL CONTAINS 0FFFFH, THEN CARRY CAN
02BE   78            MOV     A,B     ;  /ONLY BE SET TO 1
02BF   C1            POP     B       ; RESTORE ORIGINAL CONTENTS OF REGISTERS
02C0   C34303        JMP     SRET    ; SET CARRY AND RETURN
                 ;
                 ;
                 ;*****************************************************************
                 ;
                 ;
                 ; FUNCTION: NMOUT
                 ; INPUTS: A - 8 BIT INTEGER
                 ; OUTPUTS: NONE
                 ; CALLS: ECHO,PRVAL
                 ; DESTROYS: A,B,C,F/F'S
                 ; DESCRIPTION: NNMOUT CONVERTS THE 8 BIT, UNSIGNED INTEGER IN THE
                 ;              A REGISTER INTO 2 ASCII CHARACTERS.  THE ASCII CHARACTE
                 ;              ARE THE ONES REPRESENTING THE 8 BITS.  THESE TWO
                 ;              CHARACTERS ARE SENT TO THE CONSOLE AT THE CURRENT PRINT
                 ;              POSITION OF THE CONSOLE.
                 ;
02C3             NMOUT:
02C3   E5            PUSH    H       ; SAVE HL - DESTROYED BY PRVAL
02C4   F5            PUSH    PSW     ; SAVE ARGUMENT
02C5   0F            RRC
02C6   0F            RRC
02C7   0F            RRC
02C8   0F            RRC             ; GET UPPER 4 BITS TO LOW 4 BIT POSITIONS
02C9   E60F          ANI     HCHAR   ; MASK OUT UPPER 4 BITS - WANT 1 HEX CHAR
02CB   4F            MOV     C,A
02CC   CDDE02        CALL    PRVAL   ; CONVERT LOWER 4 BITS TO ASCII
02CF   CDF401        CALL    ECHO    ; SEND TO TERMINAL
02D2   F1            POP     PSW     ; GET BACK ARGUMENT
02D3   E60F          ANI     HCHAR   ; MASK OUT UPPER 4 BITS - WANT 1 HEX CHAR
```

```
02D5   4F          MOV     C,A
02D6   CDDE02      CALL    PRVAL
02D9   CDF401      CALL    ECHO
02DC   E1          POP     H       ; RESTORE SAVED VALUE OF HL
02DD   C9          RET
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION;  PRVAL
                ; INPUTS: C - INTEGER, RANGE 0 TO F
                ; OUTPUTS: C - ASCII CHARACTER
                ; CALLS: NOTHING
                ; DESTROYS: B,C,H,L,F/F'S
                ; DESCRIPTION: PRVAL CONVERTS A NUMBER IN THE RANGE 0 TO F HEX TO
                ;              THE CORRESPONDING ASCII CHARACTER, 0-9,A-F.  PRVAL
                ;              DOES NOT CHECK THE VALIDITY OF ITS INPUT ARGUMENT.
                ;
02DE            PRVAL:
02DE   21BF03       LXI     H,DIGTB ; ADDRESS OF TABLE
02E1   0600         MVI     B,0     ; CLEAR HIGH ORDER BITS OF BC
02E3   09           DAD     B       ; ADD DIGIT VALUE TO HL ADDRESS
02E4   4E           MOV     C,M     ; FETCH CHARACTER FROM MEMORY
02E5   C9           RET
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION: REGDS
                ; INPUTS: NONE
                ; OUTPUTS: NONE
                ; CALLS: ECHO,NMOUT,ERROR,CROUT
                ; DESTROYS: A,B,C,D,E,H,L,F/F'S
                ; DESCRIPTION: REGDS DISPLAYS THE CONTENTS OF THE REGISTER SAVE
                ;              LOCATIONS, IN FORMATTED FORM, ON THE CONSOLE.  THE
                ;              DISPLAY IIS DRIVEN FROM A TABLE, RTAB, WHICH CONTAINS
                ;              THE REGISTER'S PRINT SYMBOL, SAVE LOCATION ADDRESS,
                ;              AND LENGTH (8 OR 16 BITS).
                ;
02E6            REGDS:
02E6   21CF03       LXI     H,RTAB  ; LOAD HL WITH ADDRESS OF START OF TABLE
02E9            REG05:
02E9   4E           MOV     C,M     ; GET PRINT SYMBOL OF REGISTER
02EA   79           MOV     A,C
02EB   B7           ORA     A       ; TEST FOR 0 - END OF TABLE
02EC   C2F302       JNZ     REG10   ; IF NOT END, BRANCH
02EF   CDEE01       CALL    CROUT   ; ELSE, CARRIAGE RETURN/LINE FEED TO END
02F2   C9           RET             ; /DISPLAY
02F3            REG10:
```

```
02F3   CDF401          CALL    ECHO    ; ECHO CHARACTER
02F6   0E3D            MVI     C,'='
02F8   CDF401          CALL    ECHO    ; OUTPUT EQUALS SIGN, I.E. A=
02FB   23              INX     H       ; POINT TO START OF SAVE LOCATION ADDRESS
02FC   5E              MOV     E,M     ; GET LSP OF SAVE LOCATION ADDRESS TO E
02FD   1613            MVI     D,DATA SHR 8    ; PUT MSP OF SAVE LOC ADDRESS INTO D
02FF   23              INX     H       ; POINT TO LENGTH FLAG
0300   1A              LDAX    D       ; GET CONTENTS OF SAVE ADDRESS
0301   CDC302          CALL    NMOUT   ; DISPLAY ON CONSOLE
0304   7E              MOV     A,M     ; GET LENGTH FLAG
0305   B7              ORA     A       ; SET SIGN F/F
0306   CA0E03          JZ      REG15   ; IF 0, REGISTER IS 8 BITS
0309   1B              DCX     D       ; ELSE, 16 BIT REGISTER SO MORE TO DISPLAY
030A   1A              LDAX    D       ; GET LOWER 8 BITS
030B   CDC302          CALL    NMOUT   ; DISPLAY THEM
030E                REG15:
030E   0E20            MVI     C,' '
0310   CDF401          CALL    ECHO
0313   23              INX     H       ; POINT TO START OF NEXT TABLE ENTRY
0314   C3E902          JMP     REG05   ; DO NEXT REGISTER
                    ;
                    ;
                    ;*****************************************************************
                    ;
                    ;
                    ; FUNCTION: RGADR
                    ; INPUTS: C - CHARACTER DENOTING REGISTER
                    ; OUTPUTS: BC - ADDRESS OF ENTRY IN RTAB CORRESPONDING TO REGISTER
                    ; CALLS: ERROR
                    ; DESTROYS: A,B,C,D,E,H,L,F/F'S
                    ; DESCRIPTION: RGADR TAKES A SINGLE CHARACTER AS INPUT.  THIS CHARACTE
                    ;               DENOTES A REGISTER.  RGADR SEARCHES THE TABLE RTAB
                    ;               FOR A MATCH ON THE INPUT ARGUMENT.  IF ONE OCCURS,
                    ;               RGADR RETURNS THE ADDRESS OF THE ADDRESS OF THE
                    ;               SAVE LOCATION CORRESPONDING TO THE REGISTER.  THIS
                    ;               ADDRESS POINTS INTO RTAB.  IF NO MATCH OCCURS, THEN
                    ;               THE REGISTER IDENTIFIER IS ILLEGAL AND CONTROL IS
                    ;               PASSED TO THE ERROR ROUTINE.
                    ;
0317                RGADR:
0317   21CF03          LXI     H,RTAB  ; HL GETS ADDRESS OF TABLE START
031A   110300          LXI     D,RTABS ; DE GET SIZE OF A TABLE ENTRY
031D                RGA05:
031D   7E              MOV     A,M     ; GET REGISTER IDENTIFIER
031E   B7              ORA     A       ; CHECK FOR TABLE END (IDENTIFIER IS 0)
031F   CA0D02          JZ      ERROR   ; IF AT END OF TABLE, ARGUMENT IS ILLEGAL
0322   B9              CMP     C       ; ELSE, COMPARE TABLE ENTRY AND ARGUMENT
0323   CA2A03          JZ      RGA10   ; IF EQUAL, WE'VE FOUND WHAT WE'RE LOOKING FOR
0326   19              DAD     D       ; ELSE, INCREMENT TABLE POINTER TO NEXT ENTRY
0327   C31D03          JMP     RGA05   ; TRY AGAIN
032A                RGA10:
```

```
032A   23            INX     H       ; IF A MATCH, INCREMENT TABLE POINTER TO
032B   44            MOV     B,H     ; /SAVE LOCATION ADDRESS
032C   4D            MOV     C,L     ; RETURN THIS VALUE
032D   C9            RET
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION: RSTTF
                ; INPUTS: NONE
                ; OUTPUTS: NONE
                ; CALLS: NOTHING
                ; DESTROYS: A,B,C,D,E,H,L,F/F'S
                ; DESCRIPTION: RSTTF RESTORES ALL CPU REGISTER, FLIP/FLOPS, STACK
                ;              POINTER AND PROGRAM COUNTER FROM THEIR RESPECTIVE
                ;              SAVE LOCATIONS IN MEMORY.  THE ROUTINE THEN TRANSFERS
                ;              CONTROL TO THE LOCATION SPECIFIED BY THE PROGRAM
                ;              COUNTER (I.E. THE RESTORED VALUE).  THE ROUTINE
                ;              EXITS WITH THE INTERRUPTS ENABLED.
                ;
032E            RSTTF:
032E   F3            DI              ; DISABLE INTERRUPTS WHILE RESTORING THINGS
032F   21ED13        LXI     H,MSTAK ; SET MONITOR STACK POINTER TO START OF STACK
0332   F9            SPHL
0333   D1            POP     D       ; START ALSO END OF REGISTER SAVE AREA
0334   C1            POP     B
0335   F1            POP     PSW
0336   2AF713        LHLD    SSAVE   ; RESTORE USER STACK POINTER
0339   F9            SPHL
033A   2AF513        LHLD    PSAVE
033D   E5            PUSH    H       ; PUT USER RETURN ADDRESS ON USER STACK
033E   2AF313        LHLD    LSAVE   ; RESTORE HL REGISTERS
0341   FB            EI              ; ENABLE INTERRUPTS NOW
0342   C9            RET             ; JUMP TO RESTORED PC LOCATION
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION: SRET
                ; INPUTS: NONE
                ; OUTPUTS: CARRY = 1
                ; CALLS: NOTHING
                ; DESTROYS: CARRY
                ; DESCRIPTION: SRET IS JUMPED TO BY ROUTINES WISHING TO RETURN SUCCESS.
                ;              SRET SETS THE CARRY TRUE AND THEN RETURNS TO THE
                ;              CALLER OF THE ROUTINE INVOKING SRET.
                ;
0343            SRET:
0343   37            STC             ; SET CARRY TRUE
```

```
0344   C9              RET             ; RETURN APPROPRIATELY
                       ;
                       ;
                       ;*****************************************************************
                       ;
                       ;
                       ; FUNCTION: STHF0
                       ; INPUTS: DE - 16 BIT ADDRESS OF BYTE TO BE STORED INTO
                       ; OUTPUTS: NONE
                       ; CALLS: STHLF
                       ; DESTROYS: A,B,C,H,L,F/F'S
                       ; DESCRIPTION: STHF0 CHECKS THE HALF BYTE FLAG IN TEMP TO SEE IF
                       ;              IT IS SET TO LOWER.  IF SO, STHF0 STORES A 0 TO
                       ;              PAD OUT THE LOWER HALF OF THE ADDRESSED BYTE;
                       ;              OTHERWISE, THE ROUTINE TAKES NO ACTION.
                       ;
0345                   STHF0:
0345   3AF913          LDA     TEMP    ; GET HALF BYTE FLAG
0348   B7              ORA     A       ; SET F/F'S
0349   C0              RNZ             ; IF SET TO UPPER, DON'T DO ANYTHING
034A   0E00            MVI     C,0     ; ELSE, WANT TO STORE THE VALUE 0
034C   CD5003          CALL    STHLF   ; DO IT
034F   C9              RET
                       ;
                       ;
                       ;*****************************************************************
                       ;
                       ;
                       ; FUNCTION: STHLF
                       ; INPUTS: C - 4 BIT VALUE TO BE STORED IN HALF BYTE
                       ;         DE - 16 BIT ADDRESS OF BYTE TO BE STORED INTO
                       ; OUTPUTS: NONE
                       ; CALLS: NOTHING
                       ; DESTROYS: A,B,C,H,L,F/F'S
                       ; DESCRIPTION: STHLF TAKES THE 4 BIT VALUE IN C AND STORES IT IN
                       ;              HALF OF THE BYTE ADDRESSED BY REGISTERS DE.  THE
                       ;              HALF BYTE USED (EITHER UPPER OR LOWER) IS DENOTED
                       ;              BY THE VALUE OF THE FLAG IN TEMP.  STHLF ASSUMES
                       ;              THAT THIS FLAG HAS BEEN PREVIOUSLY SET
                       ;              (NOMINALLY BY ICMD).
                       ;
0350                   STHLF:
0350   D5              PUSH    D
0351   E1              POP     H       ; MOVE ADDRESS OF BYTE INTO HL
0352   79              MOV     A,C     ; GET VALUE
0353   E60F            ANI     0FH     ; FORCE TO 4 BIT LENGTH
0355   4F              MOV     C,A     ; PUT VALUE BACK
0356   3AF913          LDA     TEMP    ; GET HALF BYTE FLAG
0359   B7              ORA     A       ; CHECK FOR LOWER HALF
035A   C26303          JNZ     STH05   ; BRANCH IF NOT
035D   7E              MOV     A,M     ; ELSE, GET BYTE
```

```
035E   E6F0          ANI     0F0H    ; CLEAR LOWER 4 BITS
0360   B1            ORA     C       ; OR IN VALUE
0361   77            MOV     M,A     ; PUT BYTE BACK
0362   C9            RET
0363             STH05:
0363   7E            MOV     A,M     ; IF UPPER HALF, GET BYTE
0364   E60F          ANI     0FH     ; CLEAR UPPER 4 BITS
0366   47            MOV     B,A     ; SAVE BYTE IN B
0367   79            MOV     A,C     ; GET VALUE
0368   0F            RRC
0369   0F            RRC
036A   0F            RRC
036B   0F            RRC             ; ALIGN TO UPPER 4 BITS
036C   B0            ORA     B       ; OR IN ORIGINAL LOWER 4 BITS
036D   77            MOV     M,A     ; PUT NEW CONFIGURATION BACK
036E   C9            RET
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION: VALDG
                ; INPUTS: C - ASCII CHARACTER
                ; OUTPUTS: CARRY - 1 IF CHARACTER REPRESENTS VALID HEX DIGIT
                ;                - 0 OTHERWISE
                ; CALLS: NOTHING
                ; DESTROYS: A,F/F'S
                ; DESCRIPTION: VALDG RETURNS SUCCESS IF ITS INPUT ARGUMENT IS
                ;              AN ASCII CHARACTER REPRESENTING A VALID HEX DIGIT
                ;                   `-F), AND FAILURE OTHERWISE.
                ;
036F             VALDG:
036F   79            MOV     A,C
0370   FE30          CPI     '0'     ; TEST CHARACTER AGAINST '0'
0372   FA1802        JM      FRET    ; IF ASCII CODE LESS, CANNOT BE VALID DIGIT
0375   FE39          CPI     '9'     ; ELSE, SEE IF IN RANGE '0'-'9'
0377   FA4303        JM      SRET    ; CODE BETWEEN '0' AND '9'
037A   CA4303        JZ      SRET    ; CODE EQUAL '9'
037D   FE41          CPI     'A'     ; NOT A DIGIT - TRY FOR A LETTER
037F   FA1802        JM      FRET    ; NO - CODE BETWEEN '9' AND 'A'
0382   FE47          CPI     'G'
0384   F21802        JP      FRET    ; NO - CODE GREATER THAN 'F'
0387   C34303        JMP     SRET    ; OKAY - CODE IS 'A' TO 'F', INCLUSIVE
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION: VALDL
                ; INPUTS: C - CHARACTER
                ; OUTPUTS: CARRY - 1 IF INPUT ARGUMENT VALID DELIMTER
```

```
                        ;                   - 0 OTHERWISE
                        ; CALLS: NOTHING
                        ; DESTROYS: A,F/F'S
                        ; DESCRIPTION: VALDL RETURNS SUCCESS IF ITS INPUT ARGUMENT IS A VALID
                        ;              DELIMITER CHARACTER (SPACE, COMMA, CARRIAGE RETURN) AND
                        ;              FAILURE OTHERWISE.
                        ;
038A                    VALDL:
038A   79                       MOV     A,C
038B   FE2C                     CPI     ','     ; CHECK FOR COMMA
038D   CA4303                   JZ      SRET
0390   FE0D                     CPI     CR      ; CHECK FOR CARRIAGE RETURN
0392   CA4303                   JZ      SRET
0395   FE20                     CPI     ' '     ; CHECK FOR SPACE
0397   CA4303                   JZ      SRET
039A   C31802                   JMP     FRET    ; ERROR IF NONE OF THE ABOVE
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ;
                        ;                       MONITOR TABLES
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ;
039D                    SGNON:                          ; SIGNON MESSAGE
039D   0D0A4D43                 DB      CR,LF,'MCS-80 KIT',CR,LF
03A1   532D3830
03A5   204B4954
03A9   0D0A
000E                    LSGNON          EQU     $-SGNON ; LENGTH OF SIGNON MESSAGE
                        ;
03AB                    CADR:                           ; TABLE OF ADDRESSES OF COMMAND ROUTINES
03AB   0000                     DW      0               ; DUMMY
03AD   4101                     DW      XCMD
03AF   1D01                     DW      SCMD
03B1   FD00                     DW      MCMD
03B3   B300                     DW      ICMD
03B5   9500                     DW      GCMD
03B7   5E00                     DW      DCMD
                        ;
03B9                    CTAB:                           ; TABLE OF VALID COMMAND CHARACTERS
03B9   44                       DB      'D'
03BA   47                       DB      'G'
03BB   49                       DB      'I'
03BC   4D                       DB      'M'
03BD   53                       DB      'S'
03BE   58                       DB      'X'
0006                    NCMDS   EQU     $-CTAB  ; NUMBER OF VALID COMMANDS
```

```
                        ;
03BF                    DIGTB:                          ; TABLE OF PRINT VALUES OF HEX DIGITS
03BF    30                      DB      '0'
03C0    31                      DB      '1'
03C1    32                      DB      '2'
03C2    33                      DB      '3'
03C3    34                      DB      '4'
03C4    35                      DB      '5'
03C5    36                      DB      '6'
03C6    37                      DB      '7'
03C7    38                      DB      '8'
03C8    39                      DB      '9'
03C9    41                      DB      'A'
03CA    42                      DB      'B'
03CB    43                      DB      'C'
03CC    44                      DB      'D'
03CD    45                      DB      'E'
03CE    46                      DB      'F'
                        ;
03CF                    RTAB:                   ; TABLE OF REGISTER INFORMATION
03CF    41                      DB      'A'     ; REGISTER IDENTIFIER
03D0    F2                      DB      ASAVE AND 0FFH  ; ADDRESS OF REGISTER SAVE LOCATION
03D1    00                      DB      0       ; LENGTH FLAG - 0=8 BITS, 1=16 BITS
0003                    RTABS   EQU     $-RTAB  ; SIZE OF AN ENTRY IN THIS TABLE
03D2    42                      DB      'B'
03D3    F0                      DB      BSAVE AND 0FFH
03D4    00                      DB      0
03D5    43                      DB      'C'
03D6    EF                      DB      CSAVE AND 0FFH
03D7    00                      DB      0
03D8    44                      DB      'D'
03D9    EE                      DB      DSAVE AND 0FFH
03DA    00                      DB      0
03DB    45                      DB      'E'
03DC    ED                      DB      ESAVE AND 0FFH
03DD    00                      DB      0
03DE    46                      DB      'F'
03DF    F1                      DB      FSAVE AND 0FFH
03E0    00                      DB      0
03E1    48                      DB      'H'
03E2    F4                      DB      HSAVE AND 0FFH
03E3    00                      DB      0
03E4    4C                      DB      'L'
03E5    F3                      DB      LSAVE AND 0FFH
03E6    00                      DB      0
03E7    4D                      DB      'M'
03E8    F4                      DB      HSAVE AND 0FFH
03E9    01                      DB      1
03EA    50                      DB      'P'
03EB    F6                      DB      PSAVE+1 AND 0FFH
03EC    01                      DB      1
```

```
8080 MACRO ASSEMBLER, VER 2.3  ERRORS = 0 PAGE 27

  03ED   53              DB      'S'
  03EE   F8              DB      SSAVE+1 AND 0FFH
  03EF   01              DB      1
  03F0   00              DB      0       ; END OF TABLE MARKERS
  03F1   00              DB      0
                 ;
  03FA                   ORG     BRTAB
                 ;
  03FA   C3E301          JMP     CO      ; BRANCH TABLE FOR USER ACCESSIBLE ROUTINES
  03FD   C3D001          JMP     CI
                 ;
                 ;
                 ;*****************************************************************
                 ;
                 ;
  1300                   ORG     DATA
  13ED                   ORG     REGS    ; ORG TO REGISTER SAVE - STACK GOES IN HERE
                 ;
  13ED           MSTAK:  EQU     $       ; START OF MONITOR STACK
  13ED   00      ESAVE:  DB      0       ; E REGISTER SAVE LOCATION
  13EE   00      DSAVE:  DB      0       ; D REGISTER SAVE LOCATION
  13EF   00      CSAVE:  DB      0       ; C REGISTER SAVE LOCATION
  13F0   00      BSAVE:  DB      0       ; B REGISTER SAVE LOCATION
  13F1   00      FSAVE:  DB      0       ; FLAGS SAVE LOCATION
  13F2   00      ASAVE:  DB      0       ; A REGISTER SAVE LOCATION
  13F3   00      LSAVE:  DB      0       ; L REGISTER SAVE LOCATION
  13F4   00      HSAVE:  DB      0       ; H REGISTER SAVE LOCATION
  13F5   0000    PSAVE:  DW      0       ; PGM COUNTER SAVE LOCATION
  13F7   0000    SSAVE:  DW      0       ; USER STACK POINTER SAVE LOCATION
  13F9   00      TEMP:   DB      0       ; TEMPORARY MONITOR CELL
                 ;
  13FD                   ORG     BRLOC           ; ORG TO USER BRANCH LOCATION
                 ;
  0003           USRBR:  DS      3       ; BRANCH GOES IN HERE
                 ;
                 ;
                         END
NO PROGRAM ERRORS
```

SYMBOL TABLE

* 01

A	0007	ASAVE	13F2	B	0000	BRCHR	001B
BREAK	01BD	BRLOC	13FD	BRTAB	03FA	BSAVE	13F0
C	0001	CADR	03AB	CI	01D0	CMD	0027
CNCTL	00FB	CNIN	00FA	CNOUT	00FA	CNVBN	01DA
CO	01E3	CONST	00FB	CR	000D	CROUT	01EE
CSAVE	13EF	CTAB	03B9	D	0002	DATA	1300
DCM05	0065	DCM10	0070	DCM12	0085	DCM15	008B
DCMD	005E	DIGTB	03BF	DSAVE	13EE	E	0003
ECH05	01FD	ECH10	020B	ECHO	01F4	ERROR	020D
ESAVE	13ED	ESC	001B	FALSE	0F9C	FRET	0218
FSAVE	13F1	GCM05	00AA	GCM10	00B0	GCMD	0095
GETCH	021B	GETCM	002B	GETHX	0222	GETNM	0257
GHX05	0228	GHX10	0241	GNM05	025E	GNM10	0273
GNM15	0281	GNM20	0286	GNM25	0291	GNM30	0295
GO	0008 *	GTC03	003B	GTC05	0048	GTC10	0054
H	0004	HCHAR	000F	HIL05	02BD	HILO	029C
HSAVE	13F4	ICM05	00BE	ICM10	00E6	ICM20	00EE
ICM25	00F4	ICMD	00B3	INVRT	00FF	L	0005
LF	000A	LOWER	0000 *	LSAVE	13F3	LSGNO	000E
M	0006	MCM05	0105	MCMD	00FD	MODE	00CF
MSGL	0022	MSTAK	13ED	NCMDS	0006	NEWLN	000F
NMOUT	02C3	PRTY0	007F	PRVAL	02DE	PSAVE	13F5
PSW	0006	RBR	0002	REG05	02E9	REG10	02F3
REG15	030E	REGDS	02E6	REGS	13ED	RGA05	031D
RGA10	032A	RGADR	0317	RSTTF	032E	RSTU	0038
RTAB	03CF	RTABS	0003	SCM05	0122	SCM10	012D
SCM15	013D	SCMD	011D	SGNON	039D	SP	0006
SRET	0343	SSAVE	13F7	STH05	0363	STHF0	0345
STHLF	0350	TEMP	13F9	TERM	001B	TRDY	0001
TRUE	0F9F	UPPER	00FF	USRBR	13FD	VALDG	036F
VALDL	038A	XCM05	0154	XCM10	0163	XCM15	0170
XCM18	017B	XCM20	0194	XCM25	01AB	XCM27	01AC
XCM30	01B4	XCMD	0141				

* 02

* 03

* 04

* 05

* 06

* 07

* 08

* 09

* 10

* 11

* 12

* 13

..L