```
CP/M MACRO ASSEM 2.0    #001    MOSS 2.2 MONITOR

                  ;
                           TITLE   'MOSS 2.2 MONITOR'
                           PAGE    68
                           MACLIB  Z80
                  ;
                  ; MOSS MONITOR (VERSION 2.2)
                  ;
                  ; 20 JUNE 1980
                  ; ALL RIGHTS RESERVED BY ROBERT B. MASON
                  ;
F000              MOSS:    ORG     0F000H
F000 =            ROM:     EQU     0F000H   ;ROM START ADDRESS
0000 =            WSVEC:   EQU     0        ;VECTOR FOR WARM RESTART
0002 =            NBKPTS:  EQU     2        ;NUMBER OF BREAKPOINTS
0013 =            CTRLS:   EQU     13H      ;ASCII DC3
000D =            CR:      EQU     0DH      ;ASCII CARRIAGE RETURN
000A =            LF:      EQU     0AH      ;ASCII LINE FEED
000C =            FMFD:    EQU     0CH      ;ASCII FORM FEED
0007 =            BELL:    EQU     7        ;ASCII CNTRL CHAR TO RING THE BELL
0003 =            IOBYTE:  EQU     3        ;ADDRESS OF I/O CONTROL BYTE
0020 =            SDATA:   EQU     20H      ;SERIAL DATA PORT BASE ADDRESS
0021 =            SINTEN:  EQU     SDATA+1  ;SERIAL INTERRUPT ENABLE REGISTER
0022 =            SIDENT:  EQU     SDATA+2  ;SERIAL INTERRUPT IDENTIFICATION REGIS
0023 =            SLCTRL:  EQU     SDATA+3  ;SERIAL LINE CONTROL REGISTER
0024 =            SMDMCT:  EQU     SDATA+4  ;SERIAL MODEM CONTROL REGISTER
0025 =            SLSTAT:  EQU     SDATA+5  ;SERIAL LINE STATUS REGISTER
0026 =            SMDMST:  EQU     SDATA+6  ;SERIAL MODEM STATUS REGISTER
                  ;
                  ;
0006 =            SPSV:    EQU     6        ;STACK POINTER SAVE LOCATION
                  ;
                  ;
                  ; REGISTER STORAGE DISPLACEMENTS FROM
                  ; NORMAL SYSTEM STACK LOCATION.
                  ;
0015 =            ALOC:    EQU     15H
0013 =            BLOC:    EQU     13H
0012 =            CLOC:    EQU     12H
0011 =            DLOC:    EQU     11H
0010 =            ELOC:    EQU     10H
0014 =            FLOC:    EQU     14H
0031 =            HLOC:    EQU     31H
0030 =            LLOC:    EQU     30H
0034 =            PLOC:    EQU     34H
0017 =            SLOC:    EQU     17H
0035 =            TLOC:    EQU     35H
0025 =            TLOCX:   EQU     25H
0020 =            LLOCX:   EQU     20H
                  ;
0009 =            APLOC:   EQU     9
000B =            BPLOC:   EQU     11
000A =            CPLOC:   EQU     10
000D =            DPLOC:   EQU     13
000C =            EPLOC:   EQU     12
0008 =            FPLOC:   EQU     8
000F =            HPLOC:   EQU     15
000E =            LPLOC:   EQU     14
0007 =            XLOC:    EQU     7
0005 =            YLOC:    EQU     5
0002 =            RLOC:    EQU     2
0003 =            ILOC:    EQU     3
                  ;
                  ;
                  ; JUMP TARGETS FOR BASIC INPUT/OUTPUT
                  ;
F000 C35BF0       CBOOT:   JMP     INIT     ;COLD START
F003 C346F6       CONIN:   JMP     CI       ;CONSOLE INPUT
F006 C356F6       READER:  JMP     RI       ;READER INPUT
```

```
CP/M MACRO ASSEM 2.0     #002     MOSS 2.2 MONITOR

 F009 C300F6     CONOUT: JMP     CO       ;CONSOLE OUTPUT
 F00C C37CF6     PUNCH:  JMP     PO       ;PUNCH OUTPUT
 F00F C310F6     LIST:   JMP     LO       ;LIST OUTPUT
 F012 C323F6     CONST:  JMP     CSTS     ;CONSOLE STATUS
 F015 C36AF1             JMP     IOCHK    ;PUT IOBYTE INTO (A)
 F018 C365F1             JMP     IOSET    ;(C) HAS A NEW IOBYTE
 F01B C38AF0             JMP     MEMCK    ;MEMORY LIMIT CHECK
 F01E C394F6             JMP     RTS      ;IODEF- DEFINE USER I/O ENTRY POINTS
 F021 C394F6             JMP     RTS      ;SPCL- I/O CONTROL
 F024 C3CFF3             JMP     REST     ;BREAKPOINT ENTRY POINT
                 ;
                 ; TBL CONTAINS THE ADDRESSES OF THE ACTION ROUTINES
                 ;   THE EXECUTIVE USES IT TO LOOK UP THE DESIRED ADDRESS.
                 ;
 F027 F8F0       TBL:    DW      ASGN
 F029 09F1               DW      QPRT
 F02B 09F1               DW      QPRT
 F02D ACF1               DW      DISP
 F02F F6F4               DW      EOF
 F031 3CF1               DW      FILL
 F033 FDF1               DW      GOTO
 F035 D0F5               DW      HEXN
 F037 4DF2               DW      INPT
 F039 09F1               DW      QPRT
 F03B 09F1               DW      QPRT
 F03D 0EF5               DW      LEADER
 F03F 5DF2               DW      MOVE
 F041 09F1               DW      QPRT
 F043 55F2               DW      OUPT
 F045 09F1               DW      QPRT
 F047 21F5               DW      QUERY
 F049 4CF5               DW      READ
 F04B 67F2               DW      SUBS
 F04D 8FF2               DW      MTEST
 F04F 09F1               DW      QPRT
 F051 91F1               DW      COMP
 F053 8DF5               DW      WRITE
 F055 ECF2               DW      XMNE
 F057 9FF4               DW      I8250
 F059 82F1               DW      BYE
                 ;
                 ; THE COLD INITIALIZATION CODE
                 ;
 F05B F3         INIT:   DI               ;DISABLE INTERRUPTS
 F05C 313F00             LXI     SP,3FH   ;USE STACK TO INITIALIZE RESTARTS
 F05F 2100C3             LXI     H,JMP*256        ;  WITH RESTART ERROR VECTORS
 F062 11B2F6             LXI     D,RSTER
 F065 0610               MVI     B,16     ;16 TIMES (64 BYTES)
 F067 D5         INIT1:  PUSH    D
 F068 E5                 PUSH    H
                         DJNZ    INIT1
 F069+10FC
 F06B 3195F0             LXI     SP,FAKE-2        ;SET UP TEMPORARY STACK
 F06E 3E00               MVI     A,0      ; SKIP THE NEXT INST
 F06F                    ORG     $-1      ;SAVE A BYTE HERE
                 ;
                 ; MEMSIZ CALCULATES THE TOP OF CONTIGUOUS RAM.  IT SEARCHES
                 ;       FROM THE BOTTOM UP UNTIL A NON-RAM LOCATION IS
                 ;       FOUND.  IT THEN TAKES OFF FOR MONITOR WORK SPACE
                 ;       NEEDS AND RETURNS THE VALUE IN (H,L).
                 ;
 F06F C5         MEMSIZ: PUSH    B        ;MONITOR START LOCATION
 F070 0100F0             LXI     B,ROM
 F073 21FFFF             LXI     H,-1     ;START OF MEMORY ADDRESS SPACE
 F076 24         MEMSZ1: INR     H
 F077 7E                 MOV     A,M
 F078 2F                 CMA
 F079 77                 MOV     M,A
```

```
CP/M MACRO ASSEM 2.0     #003     MOSS 2.2 MONITOR

 F07A BE                 CMP      M
 F07B 2F                 CMA
 F07C 77                 MOV      M,A
                         JRNZ     MEMSZ2
 F07D+2004
 F07F 7C                 MOV      A,H      ;SEE IF ON MONITOR BORDER
 F080 B8                 CMP      B
                         JRNZ     MEMSZ1
 F081+20F3
 F083 25        MEMSZ2:  DCR      H        ;TAKE OFF WORKSPACE
 F084 01DEFF             LXI      B,EXIT-ENDX-3*NBKPTS+1
 F087 09                 DAD      B
 F088 C1                 POP      B        ;(B,C) IS UNPREDICTABLE DURING INIT
 F089 C9                 RET
                ;
                ; ROUTINE MEMCHK FINDS THE CURRENT TOP OF CONTIGUOUS MEMORY
                ;        (LESS THE MONITOR WORKSPACE) AND RETURNS THE VALUE.
                ;
 F08A E5        MEMCK:   PUSH     H        ;SAVE (H,L)
 F08B CD6FF0             CALL     MEMSIZ   ;GET THE RAM SIZE
 F08E 7D                 MOV      A,L
 F08F D63C               SUI      60       ;TAKE OFF WORK SPACE
                         JRNC     MEMCK0
 F091+3001
 F093 25                 DCR      H
 F094 44        MEMCK0:  MOV      B,H
 F095 E1                 POP      H
 F096 C9                 RET
                ;
 F097 99F0      FAKE:    DW       FAKE+2
 F099 F9                 SPHL
 F09A 1145F4             LXI      D,EXIT
 F09D EB                 XCHG
 F09E 011D00             LXI      B,ENDX-EXIT
                         LDIR
 F0A1+EDB0
 F0A3 010600             LXI      B,3*NBKPTS
 F0A6 D5                 PUSH     D
 F0A7 E1                 POP      H
 F0A8 2B                 DCX      H
                         LDIR
 F0A9+EDB0
 F0AB 21E8FF             LXI      H,-24
 F0AE 39                 DAD      SP
 F0AF E5                 PUSH     H
 F0B0 23                 INX      H        ;ADJUST USER STACK LOCATION
 F0B1 23                 INX      H
 F0B2 220600             SHLD     SPSV     ;SAVE THE STACK INITIAL VALUE
 F0B5 160A               MVI      D,10     ;INITIALIZE REGISTER STORAGE AREA
 F0B7 C5        INIT2:   PUSH     B
 F0B8 15                 DCR      D        ;LOOP CONTROL
                         JRNZ     INIT2
 F0B9+20FC
                ; INSERT I/O INIT CODE HERE
 F0BB CD94F6             CALL     RTS
 F0BE CD9FF4             CALL     I8250    ;INITIALIZE THE 8250
 F0C1 CD94F6             CALL     RTS
 F0C4 2190F4             LXI      H,LOGMSG ;LOG ONTO THE SYSTEM
 F0C7 CD95F6             CALL     PRTWD
                         JMPR     WINIT    ;GO TO MONITOR EXECUTIVE
 F0CA+1843
                ;
                ;  ROUTINE EXF READS ONE PARAMETER.  IT EXPECTS THE FIRST
                ;        CHARACTER OF THE PARAMETER TO BE IN THE A REGISTER
                ;        ON ENTRY.
                ;
 F0CC 0601      EXF:     MVI      B,1      ;SET UP FOR ONE PARAMETER
 F0CE 210000             LXI      H,0
```

```
CP/M MACRO ASSEM 2.0    #004    MOSS 2.2 MONITOR
                                JMPR    EX1      ;FIRST CHARACTER IN A ALREADY
 F0D1+180C
                        ;
                        ;  ROUTINE EXPR READS PARAMETERS FROM THE CONSOLE
                        ;       AND DEVELOPS A 16 BIT HEXADECIMAL FOR EACH ONE.
                        ;       THE NUMBER OF PARAMETERS WANTED IS IN THE B REG
                        ;       ON ENTRY.  A CARRIAGE RETURN WILL TERMINATE THE
                        ;       ENTRY SEQUENCE; A BLANK OR A COMMA WILL END THE
                        ;       CURRENT PARAMETER ENTRY.  EACH PARAMETER ONLY
                        ;       TAKES THE LAST 4 DIGITS TYPED IN; ANY EXCESS IS
                        ;       DISCARDED.  A NON-HEX DIGIT WILL TERMINATE THE
                        ;       ENTRY SEQUENCE AND CAUSE A WARM BOOT OF THE MON.
                        ;
                        AS3:    DJNZ    AS2      ;PART OF THE ASSIGN CODE
 F0D3+1079
                        EX3:    JRNZ    QPRT     ;NON-ZERO IS ERROR
 F0D5+2032
 F0D7 05                EXPR1:  DCR     B        ;MORE PARAMETERS?
 F0D8 C8                        RZ               ;NO, RETURN
 F0D9 210000            EXPR:   LXI     H,0      ;INITIALIZE PARAMETER
 F0DC CD7BF3            EX0:    CALL    ECHO     ;GET NEXT NUMBER
 F0DF 4F                EX1:    MOV     C,A      ;SAVE CHAR FOR LATER USE
 F0E0 CDB0F3                    CALL    NIBBLE
                                JRC     EX2      ;NOT A NUMBER, JUMP
 F0E3+3808
 F0E5 29                        DAD     H        ;MULTIPLY BY 16
 F0E6 29                        DAD     H
 F0E7 29                        DAD     H
 F0E8 29                        DAD     H
 F0E9 B5                        ORA     L        ;ADD ON NEW DIGIT
 F0EA 6F                        MOV     L,A
                                JMPR    EX0      ;GO GET NEXT DIGIT
 F0EB+18EF
 F0ED E3                EX2:    XTHL             ;PUT UNDER RETURN ADDRESS ON STACK
 F0EE E5                        PUSH    H        ;RESTORE RETURN ADDRESS
 F0EF 79                        MOV     A,C      ;REGET THE LAST CHARACTER
 F0F0 CDC3F3                    CALL    P2C      ;TEST FOR DELIMITER
                                JRNC    EX3      ;JUMP IF NOT CARRIAGE RETURN
 F0F3+30E0
                                DJNZ    QPRT     ;CARRET WITH MORE PARAM MEANS ERROR
 F0F5+1012
 F0F7 C9                        RET
                        ;
                        ; MAIN ACTION ROUTINES
                        ;
                        ;
                        ; LOGICAL ASSIGNMENT OF PERIPHERALS
                        ;
                        ;THIS ROUTINE CONTROLS THE ASSIGNMENT OF PHYSICAL
                        ;       PERIPHERALS TO THE FOUR LOGICAL DEVICE TYPES. IT
                        ;       ALTERS IOBYTE (MEMORY LOCATION 0003) TO MATCH THE
                        ;       CURRENT ASSIGNMENT.  THE FOUR LOGICAL DEVICES ARE
                        ;       CONSOLE, READER, LIST, AND PUNCH.  IN ALL CASES,
                        ;       THE TTY DEVICE IS SET UP AS THE DEFAULT DEVICE.
                        ;
 F0F8 CD7BF3            ASGN:   CALL    ECHO     ;GET THE LOGICAL DEVICE DESIRED
 F0FB 216EF1                    LXI     H,ALT    ;START OF CONVERSION TABLE
 F0FE 110500                    LXI     D,APT-ALT        ;DISTANCE BETWEEN LOGICAL CHOI
 F101 0604                      MVI     B,4      ;NUMBER OF LOGICAL CHOICES
 F103 BE                AS0:    CMP     M        ;IS THIS ONE IT?
                                JRZ     AS1      ;YES, JUMP
 F104+2842
 F106 19                        DAD     D        ;NO, GO TO NEXT LOGICAL ENTRY
                                DJNZ    AS0
 F107+10FA
 F109 218CF4            QPRT:   LXI     H,QMSG   ;GET ADDRESS OF QUESTION MARK MSG
 F10C CD98F6                    CALL    PRTWA    ;PRINT IT
                        ;
```

```
CP/M MACRO ASSEM 2.0     #005     MOSS 2.2 MONITOR

                  ; THE WARM START CODE
                  ;
F10F 2A0600       WINIT:  LHLD    SPSV      ;RESET THE STACK
F112 F9                   SPHL
F113 210FF1       WINITA: LXI     H,WINIT   ;RESET RETURN AND WARM START VECTOR
F116 E5                   PUSH    H
F117 220100               SHLD    WSVEC+1
F11A 3EC3                 MVI     A,0C3H
F11C 320000               STA     WSVEC
F11F CDA9F6               CALL    CRLF      ;START A NEW LINE
F122 CD78F3               CALL    DECHO     ;GET THE COMMAND
F125 D641                 SUI     'A'       ;GET RID OF ASCII ZONE
                          JRC     QPRT      ;BAD COMMAND
F127+38E0
F129 FE1A                 CPI     'Z'-'A'+1       ;CHECK UPPER LIMIT
                          JRNC    QPRT      ;BAD COMMAND
F12B+30DC
F12D 87                   ADD     A         ;DOUBLE IT FOR TABLE OFFSET
F12E 5F                   MOV     E,A       ;SET UP FOR DOUBLE ADD
F12F 1600                 MVI     D,0
F131 0602                 MVI     B,2       ;SET UP FOR TWO PARAMETERS
F133 2127F0               LXI     H,TBL     ;GET ACTION ROUTINE ADDRESS
F136 19                   DAD     D
F137 7E                   MOV     A,M       ;LOAD H,L INDIRECT
F138 23                   INX     H
F139 66                   MOV     H,M
F13A 6F                   MOV     L,A
F13B E9                   PCHL              ;GO TO ACTION ROUTINE
                  ;
                  ; FILL ACTION ROUTINE
                  ;
                  ;       THIS ROUTINE FILLS A BLOCK OF MEMORY WITH A USER-
                  ;       DETERMINED CONSTANT.  IT EXPECTS THREE PARAMETERS
                  ;       TO BE ENTERED IN THE FOLLOWING ORDER:
                  ;
                  ;       START ADDRESS
                  ;       FINISH ADDRESS
                  ;       FILL VALUE
                  ;
F13C CD86F3       FILL:   CALL    EXPR3     ;GET THREE PARAMETERS
F13F 71           FIO:    MOV     M,C       ;PUT DOWN THE FILL VALUE
F140 CD8FF3               CALL    HILO      ;INCREMENT AND CHECK THE POINTER
                          JRNC    FIO       ;NOT DONE YET, JUMP
F143+30FA
F145 D1                   POP     D         ;RESTORE STACK POINTER IN CASE
                          JMPR    WINIT     ; STACK WAS OVERWRITTEN
F146+18C7
                  ;
F148 50           AS1:    MOV     D,B       ;SAVE THE COUNTER RESIDUE
F149 0604                 MVI     B,4       ;LOOP CONTROL
F14B CD78F3               CALL    DECHO     ;GET THE NEW ASSIGNMENT
F14E 23           AS2:    INX     H         ;INCREMENT POINTER
F14F BE                   CMP     M         ;SEE IF THIS IS IT
                          JRNZ    AS3
F150+2081
F152 68                   MOV     L,B       ;SAVE THE RESIDUE TO FORM ASGT
F153 2D                   DCR     L         ;ADJUST VALUE
F154 42                   MOV     B,D       ;REGET THE LOGICAL RESIDUE
F155 2603                 MVI     H,3       ;SET UP THE IOBYTE MASK
F157 05                   DCR     B         ;ADJUST THIS ONE ALSO
                          JRZ     AS5       ;NO SHIFT NEEDED
F158+2804
F15A 29           AS4:    DAD     H         ;SHIFT THE MASKS INTO POSITION
F15B 29                   DAD     H
                          DJNZ    AS4       ;NOT DONE YET, JUMP
F15C+10FC
F15E 3A0300       AS5:    LDA     IOBYTE
F161 B4                   ORA     H         ;MASK THE DESIRED ASSIGNMENT IN
```

```
CP/M MACRO ASSEM 2.0    #006    MOSS 2.2 MONITOR

 F162 AC                XRA     H       ;LOGICAL ASGT BITS NOW OFF
 F163 B5                ORA     L       ;PUT IN NEW VALUE
 F164 4F                MOV     C,A
 F165 79        IOSET:  MOV     A,C
 F166 320300            STA     IOBYTE  ;SAVE NEW ASSIGNMENTS
 F169 C9                RET
 F16A 3A0300    IOCHK:  LDA     IOBYTE
 F16D C9                RET
                ;
 F16E 4C        ALT:    DB      'L'     ;LOGICAL LIST DEVICE TABLE
 F16F 32                DB      '2'     ;USER DEVICE #2
 F170 31                DB      '1'     ;USER DEVICE #1
 F171 4C                DB      'L'     ;LIST TO HIGH SPEED PRINTER
 F172 54                DB      'T'     ;LIST TO TTY
 F173 50        APT:    DB      'P'     ;LOGICAL PUNCH DEVICE TABLE
 F174 32                DB      '2'     ;USER DEVICE #2
 F175 31                DB      '1'     ;USER DEVICE #1
 F176 50                DB      'P'     ;PUNCH TO HIGH SPEED PUNCH
 F177 54                DB      'T'     ;PUNCH TO TTY
 F178 52        ART:    DB      'R'     ;LOGICAL READER DEVICE TABLE
 F179 32                DB      '2'     ;USER DEVICE #2
 F17A 31                DB      '1'     ;USER DEVICE #1
 F17B 50                DB      'P'     ;READER TO HIGH SPEED READER
 F17C 54                DB      'T'     ;READER TO TTY
 F17D 43        ACT:    DB      'C'     ;LOGICAL CONSOLE DEVICE TABLE
 F17E 31                DB      '1'     ;USER DEVICE #1
 F17F 42                DB      'B'     ;CONSOLE TO BATCH (PRINTER OR PTR)
 F180 43                DB      'C'     ;CONSOLE TO CRT
 F181 54                DB      'T'     ;CONSOLE TO TTY
                ;
                ; THE BYE ROUTINE IS USED TO PREVENT UNAUTHORIZED USAGE
                ;       OF THE SYSTEM.  THE SYSTEM LOCKS UP AND WILL NOT
                ;       RESPOND TO ANYTHING OTHER THAN TWO ASCII BELL
                ;       CHARACTERS.  WHEN IT SEES THEM CONSECUTIVELY,
                ;       CONTROL IS RETURNED TO THE MONITOR WITHOUT ALTERING
                ;       ANYTHING.
                ;
 F182 0602      BYE:    MVI     B,2     ;SET UP FOR TWO CHARACTERS
 F184 CD8FF6    BYE1:   CALL    CONI    ;GO READ THE CONSOLE
 F187 FE07              CPI     BELL    ;SEE IF AN ASCII BELL
                        JRNZ    BYE     ;NO, START OVER AGAIN
 F189+20F7
 F18B CD7EF3            CALL    ECH1    ;ECHO THE BELL
                        DJNZ    BYE1    ;NOT YET, GET NEXT ONE
 F18E+10F4
 F190 C9                RET             ;RETURN TO MONITOR
                ;
                ;  COMPARE ROUTINE
                ;
                ;THIS ROUTINE COMPARES TWO BLOCKS OF MEMORY AGAINST EACH
                ;       OTHER.  IF A DIFFERENCE IN THE RELATIVE ADDRESS
                ;       CONTENTS IS DETECTED, THE ADDRESS OF THE FIRST
                ;       BLOCK IS DISPLAYED, ALONG WITH ITS CONTENTS AND
                ;       THE CONTENTS OF THE OTHER BLOCK'S SAME RELATIVE
                ;       ADDRESS.
                ;
 F191 CD86F3    COMP:   CALL    EXPR3   ;GO GET THREE PARAMETERS
 F194 0A        CMPA:   LDAX    B       ;GET SOURCE 2 DATA
 F195 C5                PUSH    B       ;SAVE SOURCE 2 POINTER
 F196 46                MOV     B,M     ;READ SOURCE 1 DATA
 F197 B8                CMP     B       ;COMPARE DATA
                        JRZ     CMPB    ;JUMP IF OK
 F198+280C
 F19A F5                PUSH    PSW     ;SAVE SOURCE 2 DATA
 F19B CDFBF5            CALL    LADRB   ;WRITE THE ADDRESS
 F19E 78                MOV     A,B     ;GET SOURCE 1 DATA
 F19F CDF4F5            CALL    DASH1   ;FORMAT
 F1A2 F1                POP     PSW     ;REGET SOURCE 2 DATA
```

```
CP/M MACRO ASSEM 2.0    #007    MOSS 2.2 MONITOR

 F1A3 CDE6F5            CALL    HEX1    ;OUTPUT IT
 F1A6 C1        CMPB:   POP     B
 F1A7 CD9BF3            CALL    HILOXB  ;INCREMENT SOURCE 1 POINTER AND SEE IF
                        JMPR    CMPA    ;JUMP IF NOT DONE YET
 F1AA+18E8
                ;
                ; DISPLAY ACTION ROUTINE
                ;
                ;       THIS ROUTINE DISPLAYS A BLOCK OF MEMORY ON THE
                ;       CURRENT CONSOLE DEVICE (CONSOLE DUMP).  THE USER
                ;       MUST SPECIFY THE START AND FINISH ADDRESSES.
                ;       THE DISPLAY IS ORGANIZED TO DISPLAY UP TO 16 BYTES
                ;       PER DISPLAY LINE, WITH ALL COLUMNS ALIGNED SO
                ;       EACH COLUMN HAS THE SAME LAST HEX DIGIT IN ITS ADDRESS
                ;
 F1AC CDA4F6    DISP:   CALL    EXLF    ;GO GET BLOCK LIMITS
 F1AF CDFBF5    DIS1:   CALL    LADRB   ;DISPLAY THE START ADDRESS
 F1B2 7D                MOV     A,L     ;SEE IF ON 16 BYTE BOUNDARY
 F1B3 CDF0F1            CALL    TRPLSP  ;SKIP OVER TO RIGHT COLUMN
 F1B6 E5                PUSH    H       ;SAVE (H,L)
 F1B7 7E        DIS2:   MOV     A,M     ;GET THE CONTENTS
 F1B8 CDE6F5            CALL    HEX1    ;OUTPUT IT
 F1BB CD8FF3            CALL    HILO    ;INCREMENT, CHECK POINTER
                        JRC     DIS7    ;DONE IF CARRY SET
 F1BE+382A
 F1C0 CDFEF5            CALL    BLK     ;MAKE COLUMNS
 F1C3 7D                MOV     A,L     ;READY FOR NEW LINE?
 F1C4 E60F              ANI     0FH
                        JRNZ    DIS2
 F1C6+20EF
 F1C8 E1        DIS3:   POP     H       ;REGET LINE START ADDRESS
 F1C9 7D                MOV     A,L     ;SKIP OVER TO RIGHT SPACE
 F1CA E60F              ANI     0FH
 F1CC CDF5F1            CALL    TRPL2
 F1CF 7E        DIS4:   MOV     A,M     ;GET MEMORY VALUE
 F1D0 E67F              ANI     7FH     ;STRIP OFF PARITY BIT
 F1D2 4F                MOV     C,A     ;SET UP FOR OUTPUT
 F1D3 FE20              CPI     ' '     ;SEE IF PRINTABLE IN ASCII
                        JRC     DIS5    ;JUMP IF SO
 F1D5+3804
 F1D7 FE7E              CPI     7EH
                        JRC     DIS6
 F1D9+3802
 F1DB 0E2E      DIS5:   MVI     C,'.'   ;ELSE, PRINT A DOT
 F1DD CD09F0    DIS6:   CALL    CONOUT
 F1E0 CD9CF3            CALL    HILOX   ;INCREMENT (H,L) AND SEE IF DONE
 F1E3 7D                MOV     A,L     ;NOT DONE, READY FOR NEW LINE?
 F1E4 E60F              ANI     0FH
                        JRNZ    DIS4    ;JUMP IF NOT
 F1E6+20E7
                        JMPR    DIS1    ;DO THE NEXT LINE
 F1E8+18C5
 F1EA 93        DIS7:   SUB     E       ;SKIP OVER TO START ASCII PRINTOUT
 F1EB CDF0F1            CALL    TRPLSP
                        JMPR    DIS3    ;GO PRINT THE ASCII
 F1EE+18D8
                ;
 F1F0 E60F      TRPLSP: ANI     0FH     ;ISOLATE THE LOW FOUR BITS
 F1F2 47                MOV     B,A     ;PREPARE TO SPACE OVER TO RIGHT COLUMN
 F1F3 87                ADD     A       ;TRIPLE THE COUNT
 F1F4 80                ADD     B
 F1F5 47        TRPL2:  MOV     B,A     ;PUT BACK INTO B
 F1F6 04                INR     B       ;ADJUST COUNTER
 F1F7 CDFEF5    TRPL1:  CALL    BLK     ;DO THE SPACING
                        DJNZ    TRPL1   ;NO, DO ANOTHER COLUMN
 F1FA+10FB
 F1FC C9                RET
                ;
```

```
CP/M MACRO ASSEM 2.0    #008    MOSS 2.2 MONITOR
                    ; GO TO ACTION ROUTINE
                    ;
                    ;
                    ; GOTO COMMAND TRANSFERS CONTROL TO A SPECIFIED ADDRESS.
                    ;   IT ALLOWS THE SELECTIVE SETTING OF UP TO TWO BREAKPOINTS
                    ;   AS WELL AS ALLOWING ANY CONSOLE INPUT TO BREAKPOINT
                    ;   THE RUN, AS LONG AS INTERRUPT 1 IS ACTIVE.
                    ;
 F1FD CDC0F3        GOTO:   CALL    PCHK     ;SEE IF OLD ADDRESS WANTED
                            JRC     GO3      ;  YES, JUMP
 F200+3837
                            JRZ     GO0      ;  YES, BUT SET SOME BREAKPOINTS
 F202+2810
 F204 CDCCF0                CALL    EXF      ;GET NEW GOTO ADDRESS
 F207 D1                    POP     D
 F208 213400                LXI     H,PLOC   ;PUT ADDRESS IN PC LOCATION
 F20B 39                    DAD     SP
 F20C 72                    MOV     M,D      ;LOW BYTE
 F20D 2B                    DCX     H
 F20E 73                    MOV     M,E      ;HIGH BYTE
 F20F 79                    MOV     A,C
 F210 FE0D                  CPI     CR       ;SEE IF A CR WAS LAST ENTERED
                            JRZ     GO3
 F212+2825
 F214 0602          GO0:    MVI     B,NBKPTS
 F216 213500                LXI     H,TLOC   ;POINT TO TRAP STORAGE
 F219 39                    DAD     SP
 F21A C5            GO1:    PUSH    B        ;SAVE NUMBER OF BREAKPOINTS
 F21B E5                    PUSH    H        ;SAVE STORAGE POINTER
 F21C 0602                  MVI     B,2      ;SET UP TO GET A TRAP ADDRESS
 F21E CDD7F0                CALL    EXPR1    ;GET A TRAP ADDRESS
 F221 D1                    POP     D        ;GET THE TRAP ADDRESS INTO (D,E)
 F222 E1                    POP     H        ;REGET THE STORAGE ADDRESS
 F223 7A                    MOV     A,D      ;INSURE THE TRAP ADDRESS ISN'T ZERO
 F224 B3                    ORA     E
                            JRZ     GO2      ;JUMP IF SO
 F225+280A
 F227 73                    MOV     M,E      ;SAVE THE BREAKPOINT ADDRESS
 F228 23                    INX     H
 F229 72                    MOV     M,D
 F22A 23                    INX     H
 F22B 1A                    LDAX    D        ;SAVE THE INSTRUCTION FROM THE BP ADDR
 F22C 77                    MOV     M,A
 F22D 23                    INX     H
 F22E 3ECF                  MVI     A,RST OR 8       ;INSERT THE BREAKPOINT
 F230 12                    STAX    D
 F231 79            GO2:    MOV     A,C      ;REGET THE DELIMITER TO SEE
 F232 FE0D                  CPI     CR       ;  IF WE ARE DONE SETTING BREAKPOINTS
 F234 C1                    POP     B        ;  UNLOAD THE STACK FIRST
                            JRZ     GO3      ;YES, JUMP
 F235+2802
                            DJNZ    GO1      ;JUMP IF NOT AT BP LIMIT
 F237+10E1
 F239 CDA9F6        GO3:    CALL    CRLF
 F23C E1                    POP     H        ;GET RID OF STACK JUNK
 F23D 2143F4                LXI     H,RS9
 F240 E5                    PUSH    H
 F241 21CFF3                LXI     H,REST
 F244 220900                SHLD    9        ;SET BREAKPOINT JUMP VECTOR ADDRESS
 F247 211800                LXI     H,24     ;FIND REGISTER SET ROUTINE ADDRESS
 F24A 39                    DAD     SP
 F24B D1                    POP     D        ;ADJUST THE STACK
 F24C E9                    PCHL             ;GO TO THE DESIRED PLACE
                    ;
                    ; GENERAL PURPOSE INPUT/OUTPUT ROUTINES
                    ;
                    ;THESE ROUTINES ALLOW BYTE-BY-BYTE INPUT OR OUTPUT FROM
                    ;       THE CURRENT CONSOLE DEVICE.  THEY ARE INVOKED BY
```

```
CP/M MACRO ASSEM 2.0    #009    MOSS 2.2 MONITOR
                    ;       THE MONITOR "I" OR "O" COMMAND.
                    ;
 F24D CDD7F0        INPT:   CALL    EXPR1   ;GET INPUT PORT NUMBER
 F250 C1                    POP     B       ;GET PORT # INTO C REGISTER
                            INP     E       ;READ VALUE INTO E REGISTER
 F251+ED58
                            JMPR    BITS2   ;GO DO A BINARY PRINT OF THE VALUE
 F253+1851
                    ;
 F255 CDD9F0        OUPT:   CALL    EXPR    ;GET THE ADDRESS AND DATA FOR OUTPUT
 F258 D1                    POP     D       ;DATA VALUE INTO E
 F259 C1                    POP     B       ;PORT INTO C
                            OUTP    E       ;DO THE OUTPUT
 F25A+ED59
 F25C C9                    RET
                    ;
                    ;  MOVE ROUTINE
                    ;
                    ;       THIS ROUTINE EXPECTS THREE PARAMETERS, ENTERED IN THE
                    ;       SOURCE FIRST BYTE ADDRESS
                    ;       SOURCE LAST BYTE ADDRESS
                    ;       DESTINATION FIRST BYTE ADDRESS
                    ;
 F25D CD86F3        MOVE:   CALL    EXPR3   ;GET THREE PARAMETERS
 F260 7E            MOV1:   MOV     A,M     ;GET NEXT BYTE
 F261 02                    STAX    B       ;MOVE IT
 F262 CD9BF3                CALL    HILOXB  ;GO INCREMENT, CHECK SOURCE POINTER
                            JMPR    MOV1    ;NOT THERE YET, GO DO IT AGAIN
 F265+18F9
                    ;
                    ; SUBSTITUTE ACTION ROUTINE
                    ;
                    ;THIS ROUTINE ALLOWS THE USER TO INSPECT ANY MEMORY LOCATION
                    ;       AND ALTER THE CONTENTS, IF DESIRED AND IF THE ADDRESS
                    ;       IS IN RAM.  THE CONTENTS MAY BE LEFT UNALTERED
                    ;       BY ENTERING A SPACE, COMMA, OR A CARRIAGE RETURN.  IF
                    ;       A CARRIAGE RETURN IS ENTERED, THE ROUTINE IS TERMINATE
                    ;       IF A SPACE OR COMMA IS ENTERED, THE ROUTINE
                    ;       PROCEEDS TO THE NEXT LOCATION AND PRESENTS THE USER
                    ;       WITH AN OPPORTUNITY TO ALTER IT.
                    ;
 F267 CDD7F0        SUBS:   CALL    EXPR1   ;GO GET ONE PARAMETER
 F26A E1                    POP     H       ;GET THE START ADDRESS
 F26B 7E            SUB1:   MOV     A,M     ;GET THE CONTENTS OF THE ADDRESS
 F26C CDF4F5                CALL    DASH1   ;DISPLAY IT ON CONSOLE AND A DASH
 F26F CDC0F3                CALL    PCHK    ;GET, CHECK CHARACTER
 F272 D8                    RC              ;DONE IF CARRIAGE RETURN
                            JRZ     SUB2    ;NO CHANGE IF BLANK OR ,
 F273+280F
 F275 FE0A                  CPI     LF      ;SEE IF PREVIOUS BYTE WANTED
                            JRZ     SUB3    ;YES, DO IT
 F277+280D
 F279 E5                    PUSH    H       ;SAVE MEMORY POINTER
 F27A CDCCF0                CALL    EXF     ;GO GET REST OF NEW VALUE
 F27D D1                    POP     D       ;NEW VALUE TO E REGISTER
 F27E E1                    POP     H       ;RESTORE MEMORY POINTER
 F27F 73                    MOV     M,E     ;PUT DOWN NEW VALUE
 F280 79                    MOV     A,C     ;GET THE DELIMITER
 F281 FE0D                  CPI     CR      ;SEE IF DONE (CARRIAGE RETURN)
 F283 C8                    RZ              ;YES, RETURN TO MONITOR
 F284 23            SUB2:   INX     H       ;NO, INCREMENT MEMORY POINTER
 F285 23                    INX     H       ;ALLOW A FALL-THROUGH ON THE NEXT INST
 F286 2B            SUB3:   DCX     H       ;ADJUST (H,L) AS APPROPRIATE
 F287 7D                    MOV     A,L     ;GET LO ADDRESS BYTE
 F288 E607                  ANI     7       ;SEE IF ON A BOUNDARY
 F28A CCFBF5                CZ      LADRB   ;CALL IF ON THE BOUNDARY
                            JMPR    SUB1    ;GO DO THE NEXT LOCATION
 F28D+18DC
```

```
CP/M MACRO ASSEM 2.0    #010    MOSS 2.2 MONITOR

                   ;
                   ; MTEST ROUTINE TESTS A SPECIFIED BLOCK OF MEMORY TO
                   ;    SEE IF ANY HARD DATA BIT FAILURES EXIST.  IT IS
                   ;    NOT AN EXHAUSTIVE TEST, BUT JUST A QUICK INDICATION
                   ;    OF THE MEMORY'S OPERATIVENESS.
                   ;
F28F CDA4F6        MTEST:  CALL    EXLF
F292 7E            MTEST1: MOV     A,M     ;READ A BYTE
F293 F5                    PUSH    PSW     ;SAVE IT
F294 2F                    CMA             ;COMPLEMENT IT
F295 77                    MOV     M,A     ;WRITE IT
F296 AE                    XRA     M       ;RESULT SHOULD BE ZERO
F297 C4A1F2                CNZ     BITS    ;LOG ERROR IF NOT
F29A F1            MTEST2: POP     PSW     ;RESTORE ORIGINAL BYTE
F29B 77                    MOV     M,A
F29C CD9CF3                CALL    HILOX   ;POINT TO NEXT AND SEE IF DONE
                           JMPR    MTEST1  ;NO, CONTINUE
F29F+18F1
                   ;
F2A1 D5            BITS:   PUSH    D       ;SAVE (D,E)
F2A2 5F                    MOV     E,A     ;SAVE ERROR PATTERN IN E
F2A3 CDFBF5                CALL    LADRB   ;FIRST PRINT THE ADDRESS
F2A6 0608          BITS2:  MVI     B,8     ;LOOP CONTROL FOR 8 BITS
F2A8 7B            BITS1:  MOV     A,E     ;GET NEXT BIT
F2A9 07                    RLC             ;  INTO CARRY
F2AA 5F                    MOV     E,A     ;SAVE REST
F2AB 3E18                  MVI     A,'0'/2 ;BUILD ASCII 1 OR 0
F2AD 17                    RAL             ;  CARRY DETERMINES WHICH
F2AE 4F                    MOV     C,A     ;NOW, OUTPUT IT
F2AF CD09F0                CALL    CONOUT
                           DJNZ    BITS1   ;DO IT AGAIN
F2B2+10F4
F2B4 D1                    POP     D
F2B5 C9                    RET
                   ;
                   ; EXAMINE REGISTERS COMMAND INSPECTS THE VALUES OF THE
                   ;    THE REGISTERS STORED BY THE LAST ENCOUNTERED BREAKPOINT.
                   ;    THE VALUES MAY BE MODIFIED IF DESIRED.
                   ;
F2B6 23            XAA:    INX     H       ;SKIP OVER TO NEXT ENTRY
F2B7 23                    INX     H
F2B8 34            XA:     INR     M       ;SEE IF AT END OF TABLE
F2B9 C8                    RZ              ;COULDN'T FIND MATCH, QUIT
F2BA F2C1F2                JP      XAB     ;SORT OUT BIT 7 OF TABLE
F2BD F680                  ORI     80H     ;SET IT ON TEST VALUE
                           JMPR    XAC
F2BF+1802
F2C1 E67F          XAB:    ANI     7FH     ;RESET BIT 7
F2C3 35            XAC:    DCR     M       ;TO BE PULLED OUT IN ROM
F2C4 BE                    CMP     M       ;SEE IF THIS IS IT
                           JRNZ    XAA     ;NO, GO TRY AGAIN
F2C5+20EF
F2C7 CDFEF5                CALL    BLK     ;YES, PREPARE TO SHOW CURRENT VALUE
F2CA CD15F3                CALL    PRTVAL  ;GO PRINT THE VALUE
F2CD CDF7F5                CALL    DASH    ;PROMPT A NEW VALUE
F2D0 CDC0F3                CALL    PCHK    ;GET THE INPUT
F2D3 D8                    RC              ;DONE IF CARRIAGE RETURN
                           JRZ     XF      ;JUMP IF NO CHANGE DESIRED
F2D4+2812
F2D6 E5                    PUSH    H       ;TO BE CHANGED, SAVE POINTER
F2D7 CDCCF0                CALL    EXF     ;GET THE NEW VALUE
F2DA E1                    POP     H       ;  INTO (H,L)
F2DB 7D                    MOV     A,L     ;GET THE NEW LOW BYTE
F2DC 13                    INX     D       ;ADJUST POINTER
F2DD 12                    STAX    D       ;PUT IT DOWN
F2DE E3                    XTHL            ;RECOVER THE TABLE POINTER
F2DF 7E                    MOV     A,M     ;GET THE ATTRIBUTES
F2E0 E3                    XTHL            ;SET THE STACK STRAIGHT
```

```
CP/M MACRO ASSEM 2.0     #011     MOSS 2.2 MONITOR

 F2E1 07                   RLC              ;SEE IF 8 BIT REGISTER
                           JRNC     XE      ;JUMP IF SO
 F2E2+3003
 F2E4 13                   INX      D       ;REGISTER PAIR, DO OTHER 8 BITS
 F2E5 7C                   MOV      A,H
 F2E6 12                   STAX     D
 F2E7 E1         XE:       POP      H       ;RESTORE THE TABLE POINTER
 F2E8 79         XF:       MOV      A,C     ;SEE IF IT WAS A CR
 F2E9 FE0D                 CPI      CR
 F2EB C8                   RZ               ;DONE IF SO
 F2EC 213DF3     XMNE:     LXI      H,ACTBL ;GET ADDRESS OF REGISTER LOOK-UP TABLE
 F2EF CDC0F3     XMNE1:    CALL     PCHK    ;FIND OUT WHAT ACTION IS WANTED
                           JRC      XG      ;SHOW ALL IF CARRIAGE RETURN
 F2F2+380B
                           JRZ      XMNE1   ;IGNORE BLANKS OR COMMAS
 F2F4+28F9
 F2F6 FE27                 CPI      ''''    ;SEE IF PRIMES WANTED
                           JRNZ     XA      ;NO, MUST BE SINGLE REGISTER
 F2F8+20BE
 F2FA 2155F3               LXI      H,PRMTB ;YES, SET TABLE ADDRESS
                           JMPR     XMNE1   ;  AND FIND OUT WHICH ONE
 F2FD+18F0
                 ;
 F2FF 7E         XG:       MOV      A,M
 F300 4F                   MOV      C,A
 F301 3C                   INR      A       ;SEE IF AT END OF TABLE
 F302 C8                   RZ               ;DONE IF SO
 F303 FCA9F6               CM       CRLF    ;START A NEW LINE IF BIT 7 IS SET
 F306 CD09F0               CALL     CONOUT
 F309 CDF7F5               CALL     DASH    ;PROMPT FOR A NEW VALUE
 F30C CD15F3               CALL     PRTVAL  ;GO PRINT THE VALUE
 F30F CDFEF5               CALL     BLK     ;FORMATTER
 F312 23                   INX      H       ;POINT TO NEXT ENTRY
                           JMPR     XG      ;DO THE NEXT VALUE
 F313+18EA
                 ;
 F315 23         PRTVAL:   INX      H       ;POINT TO NEXT ENTRY
 F316 7E                   MOV      A,M     ;GET OFFSET AND ATTRIBUTES BYTE
 F317 E63F                 ANI      3FH     ;ISOLATE THE OFFSET
 F319 C602                 ADI      2       ;ALLOW FOR RETURN ADDRESS
 F31B EB                   XCHG             ;SWAP POINTERS
 F31C 6F                   MOV      L,A     ;BUILD THE ADDRESS OF THE REG CONTENTS
 F31D 2600                 MVI      H,0
 F31F 39                   DAD      SP
 F320 EB                   XCHG             ;RE-SWAP THE POINTERS
 F321 7E                   MOV      A,M     ;NOW FIND OUT ATTRIBUTES
 F322 0601                 MVI      B,1     ;SET UP FOR SINGLE REG VALUE
 F324 07                   RLC
                           JRNC     PV1     ;JUMP IF SINGLE REGISTER VALUE WANTED
 F325+300E
 F327 04                   INR      B       ;SET UP FOR REGISTER PAIR
 F328 07                   RLC
                           JRNC     PV1     ;JUMP IF REGISTER PAIR IS NEXT
 F329+300A
 F32B E5                   PUSH     H       ;SPECIAL CASE FOR MEMORY REGISTER
 F32C 1A                   LDAX     D       ;BUILD ADDRESS IN (H,L)
 F32D 67                   MOV      H,A
 F32E 1B                   DCX      D
 F32F 1A                   LDAX     D
 F330 6F                   MOV      L,A
 F331 7E                   MOV      A,M     ;GET THE MEMORY VALUE
 F332 E1                   POP      H       ;RESTORE (H,L)
                           DJNZ     PV2     ;ALWAYS JUMP
 F333+1001
 F335 1A         PV1:      LDAX     D       ;GET THE REGISTER CONTENTS
 F336 CDE6F5     PV2:      CALL     HEX1    ;OUTPUT THE VALUE
 F339 1B                   DCX      D       ;ADJUST THE MEMORY POINTER
                           DJNZ     PV1
```

```
CP/M MACRO ASSEM 2.0    #012    MOSS 2.2 MONITOR

 F33A+10F9
 F33C C9                        RET
                        ;
 F33D C115              ACTBL:  DB      80H+'A',ALOC
 F33F 4213                      DB      'B',BLOC
 F341 4312                      DB      'C',CLOC
 F343 4411                      DB      'D',DLOC
 F345 4510                      DB      'E',ELOC
 F347 4614                      DB      'F',FLOC
 F349 4831                      DB      'H',HLOC
 F34B 4C30                      DB      'L',LLOC
 F34D CDF1                      DB      80H+'M',HLOC+0C0H
 F34F 50B4                      DB      'P',PLOC+80H
 F351 5397                      DB      'S',SLOC+80H
 F353 4903                      DB      'I',ILOC
                        ;
                        ; REST OF Z-80 REGISTER OFFSETS
                        ;
 F355 C109              PRMTB:  DB      80H+'A',APLOC
 F357 420B                      DB      'B',BPLOC
 F359 430A                      DB      'C',CPLOC
 F35B 440D                      DB      'D',DPLOC
 F35D 450C                      DB      'E',EPLOC
 F35F 4608                      DB      'F',FPLOC
 F361 480F                      DB      'H',HPLOC
 F363 4C0E                      DB      'L',LPLOC
 F365 CDCF                      DB      80H+'M',HPLOC+0C0H
 F367 5887                      DB      'X',XLOC+80H
 F369 5985                      DB      'Y',YLOC+80H
 F36B 5202                      DB      'R',RLOC
 F36D FF                        DB      0FFH
                        ;
                        ; GENERAL PURPOSE ROUTINES
                        ;
                        ;
                        ; ROUTINE CONV CONVERTS THE LOW ORDER NIBBLE OF THE
                        ;       ACCUMULATOR TO ITS ASCII EQUIVALENT.  IT
                        ;       PUTS THE RESULT INTO C FOR LATER OUTPUT.
                        ;
 F36E E60F              CONV:   ANI     0FH     ;STRIP OFF BITS 4-7
 F370 C690                      ADI     90H     ;PUT ON THE ASCII ZONE
 F372 27                        DAA
 F373 CE40                      ACI     40H
 F375 27                        DAA
 F376 4F                        MOV     C,A     ;PUT IN OUTPUT PASS REGISTER
 F377 C9                        RET
                        ;
                        ; ROUTINE ECHO READS A BYTE FROM A HALF-DUPLEX CONSOLE
                        ;       DEVICE, THEN ECHOES THE CHARACTER BACK TO THE
                        ;       CONSOLE.
                        ;
 F378 CDF7F5            DECHO:  CALL    DASH    ;PRINT A DASH
 F37B CD8FF6            ECHO:   CALL    CONI    ;CONSOLE READ, WRITE ROUTINE
 F37E C5                ECH1:   PUSH    B       ;  SAVE (B,C)
 F37F 4F                        MOV     C,A     ;  PASS CHARACTER IN C REGISTER
 F380 CD09F0                    CALL    CONOUT  ;  OUTPUT IT
 F383 79                        MOV     A,C     ;  PUT CHARACTER BACK INTO A
 F384 C1                        POP     B       ;  RESTORE (B,C)
 F385 C9                        RET
                        ;
                        ; ROUTINE EXPR3 GETS THREE PARAMETERS, DOES A CR, LF AND
                        ;     THEN LOADS (B,C), (D,E), AND (H,L) WITH THE PARAMETERS.
                        ;
 F386 04                EXPR3:  INR     B       ;2 IS ALREADY IN THE B REGISTER
 F387 CDD9F0                    CALL    EXPR    ;GET THE PARAMETERS
 F38A C1                        POP     B       ;PUT PARAMETERS INTO REGISTERS
 F38B D1                        POP     D
 F38C C3AAF6                    JMP     CRLFA   ;GO DO THE CARRIAGE RETURN SEQUENCE
```

```
CP/M MACRO ASSEM 2.0    #013    MOSS 2.2 MONITOR

               ;
               ; ROUTINE HILO INCREMENTS (H,L).  IT THEN CHECKS FOR (AND
               ;       DISALLOWS) A WRAP-AROUND SITUATION.  IF IT OCCURS,
               ;       THE CARRY BIT WILL BE SET ON RETURN.  IF NO WRAP-
               ;       AROUND OCCURRED, (H,L) IS COMPARED TO (D,E) AND
               ;       THE FLAG BITS SET ACCORDINGLY.
               ;
F38F 23        HILO:   INX     H       ;INCREMENT (H,L)
F390 7C                MOV     A,H     ;TEST IF ZERO
F391 B5                ORA     L       ;  IN (H,L)
F392 37                STC             ;SET CARRY FOR (H,L)=0
F393 C8                RZ              ;RETURN IF (H,L) = 0
F394 7B                MOV     A,E     ;COMPARE (H,L) TO (D,E)
F395 95                SUB     L
F396 7A                MOV     A,D
F397 9C                SBB     H
F398 C9                RET             ;RETURN WITH FLAGS SET
               ;
               ; ROUTINE HILOX INCREMENTS (H,L), COMPARES IT TO (D,E) AND
               ;   IF EQUAL, RETURNS CONTROL TO THE MONITOR EXECUTIVE.
               ;   OTHERWISE, CONTROL RETURNS TO THE CALLING ROUTINE.
               ;
F399 D1        HILOD:  POP     D       ;GET RID OF RETURN ADDRESS
F39A C9                RET             ;RETURN TO MONITOR
F39B 03        HILOXB: INX     B       ;INCREMENT (B,C)
F39C CD8FF3    HILOX:  CALL    HILO    ;INC AND CHECK (H,L)

                       JRC     HILOD   ;DONE IF CARRY SET
F39F+38F8
F3A1 CD12F0            CALL    CONST   ;SEE IF CONSOLE BREAK PENDING
F3A4 B7                ORA     A
F3A5 C8                RZ              ;NONE, RETURN TO CONTINUE
F3A6 CD8FF6            CALL    CONI    ;SEE IF WAIT OR BREAK
F3A9 FE13              CPI     CTRLS
                       JRNZ    HILOD   ;JUMP IF BREAK
F3AB+20EC
F3AD C38FF6            JMP     CONI    ;GO WAIT FOR NEXT CHARACTER
               ;
               ; ROUTINE NIBBLE CONVERTS THE ASCII CHARACTERS 0-9 AND
               ;       A-F TO THEIR EQUIVALENT HEXADECIMAL VALUE.  IF
               ;       THE CHARACTER IS NOT IN RANGE, THE CARRY BIT IS SET TO
               ;       FLAG THE ERROR.
               ;
F3B0 D630      NIBBLE: SUI     '0'     ;ASCII TO HEX CONVERSION
F3B2 D8                RC              ;  DONE IF OUT OF RANGE
F3B3 FE17              CPI     'G'-'0' ;CHECK UPPER END
F3B5 3F                CMC             ;  TOGGLE THE CARRY BIT
F3B6 D8                RC              ;  DONE IF OUT OF RANGE
F3B7 FE0A              CPI     '9'-'0'+1  ;SEE IF NUMERIC
F3B9 3F                CMC             ;  TOGGLE THE CARRY BIT
F3BA D0                RNC             ;  DONE IF SO
F3BB D607              SUI     'A'-'9'-1  ;SUBTRACT THE ALPHA BIAS
F3BD FE0A              CPI     10      ;  SET CARRY FOR INVALID CHAR
F3BF C9                RET
               ;
               ; ROUTINE PCHK READS A CHARACTER FROM THE CONSOLE, THEN
               ;       CHECKS IT FOR A DELIMITER. IF IT IS NOT
               ;       A DELIMITER, A NON-ZERO CONDITION IS RETURNED.
               ;       IF IT IS A DELIMITER, A ZERO CONDITION IS RETURNED.
               ;       FURTHER, IF THE DELIMITER IS A CARRIAGE RETURN,
               ;       THE CARRY BIT IS SET.  A BLANK OR A COMMA RESETS
               ;       THE CARRY BIT.
               ;
F3C0 CD7BF3    PCHK:   CALL    ECHO    ;GET, TEST FOR DELIMITER
F3C3 FE20      P2C:    CPI     ' '     ;  BLANK?
F3C5 C8                RZ              ;  YES, DONE
F3C6 FE2C              CPI     ','     ;  NO, COMMA?
F3C8 C8                RZ              ;  YES, DONE
```

```
CP/M MACRO ASSEM 2.0    #014    MOSS 2.2 MONITOR

 F3C9 FE0D              CPI     CR      ;  NO, CARRIAGE RETURN?
 F3CB 37                STC             ;  SHOW IT IN CARRY BIT
 F3CC C8                RZ              ;  DONE IF CR
 F3CD 3F                CMC             ;CLEAR CARRY FOR NO DELIMITER
 F3CE C9                RET
                ;
                ; ROUTINE REST TRAPS ALL OF THE REGISTER CONTENTS WHENEVER A
                ;       RESTART 1 INSTRUCTION IS EXECUTED.  THE TRAPPED CONTEN
                ;       ARE STORED IN THE SYSTEM STACK AREA FOR LATER ACCESS A
                ;       USE BY THE GOTO AND THE EXAMINE REGISTERS COMMANDS.
                ;
                ;
                ; INSERT INTERRUPT DISABLER SOFTWARE AT START OF REST:
 F3CF E5        REST:   PUSH    H       ;SAVE ALL THE REGISTERS
 F3D0 D5                PUSH    D
 F3D1 C5                PUSH    B
 F3D2 F5                PUSH    PSW
 F3D3 CD6FF0            CALL    MEMSIZ  ;GET THE MONITOR'S STACK LOCATION
 F3D6 EB                XCHG
 F3D7 210A00            LXI     H,10    ;GO UP 10 BYTES IN THE STACK
 F3DA 39                DAD     SP      ;  TO SKIP OVER TEMP REGISTER SAVE
 F3DB 0604              MVI     B,4     ;PICK OFF THE REGISTER VALUES
 F3DD EB                XCHG
 F3DE 2B        RS1:    DCX     H
 F3DF 72                MOV     M,D     ;SAVE IN WORK AREA
 F3E0 2B                DCX     H
 F3E1 73                MOV     M,E
 F3E2 D1                POP     D
                        DJNZ    RS1
 F3E3+10F9
 F3E5 C1                POP     B       ;GET THE BREAKPOINT LOCATION
 F3E6 0B                DCX     B
 F3E7 F9                SPHL            ;SET THE MONITOR STACK
 F3E8 212500            LXI     H,TLOCX ;SET UP TO RESTORE BREAKPOINTS
 F3EB 39                DAD     SP
 F3EC D5                PUSH    D
 F3ED 1602              MVI     D,NBKPTS ;LOOP CONTROL FOR N BREAKPOINTS
 F3EF 7E        RS2:    MOV     A,M
 F3F0 91                SUB     C       ;SEE IF A SOFTWARE TRAP
 F3F1 23                INX     H
 F3F2 7E                MOV     A,M
 F3F3 98                SBB     B       ;MAYBE, TRY REST OF ADDRESS
                        JRZ     RS5     ;FOUND ONE, JUMP TO RESET IT
 F3F4+2806
 F3F6 23        RS3:    INX     H       ;NOT FOUND, TRY NEXT ONE
 F3F7 23                INX     H
 F3F8 15                DCR     D
                        JRNZ    RS2
 F3F9+20F4
 F3FB 03        RS4:    INX     B       ;NONE FOUND
 F3FC 212000    RS5:    LXI     H,LLOCX
 F3FF D1                POP     D
 F400 39                DAD     SP
 F401 73                MOV     M,E     ;STORE USER (H,L)
 F402 23                INX     H
 F403 72                MOV     M,D
 F404 C5                PUSH    B       ;SAVE (B,C)
 F405 0E2A              MVI     C,'*'   ;TYPE THE BREAK INDICATION
 F407 CD09F0            CALL    CONOUT
 F40A D1                POP     D       ;REGET THE BREAKPOINT LOCATION
 F40B 3EF4              MVI     A,RS9/256
 F40D BA                CMP     D       ;SEE IF A RET BREAKPOINT
                        JRZ     RS6
 F40E+2809
 F410 23                INX     H
 F411 23                INX     H
 F412 73                MOV     M,E     ;RESTORE USER PROGRAM COUNTER
 F413 23                INX     H
 F414 72                MOV     M,D
```

```
CP/M MACRO ASSEM 2.0     #015      MOSS 2.2 MONITOR

F415 EB                  XCHG                ;PRINT THE BREAKPOINT LOCATION
F416 CDE1F5              CALL      LADR
F419 212500      RS6:    LXI       H,TLOCX
F41C 39                  DAD       SP
F41D 010002              LXI       B,NBKPTS*256
F420 5E          RS7:    MOV       E,M       ;RESTORE BREAKPOINTED LOCATIONS
F421 71                  MOV       M,C       ;RESET SYSTEM BP SAVE AREA
F422 23                  INX       H
F423 56                  MOV       D,M
F424 71                  MOV       M,C
F425 23                  INX       H
F426 7B                  MOV       A,E
F427 B2                  ORA       D
                         JRZ       RS8       ;DO NOTHING IF ZERO
F428+2802
F42A 7E                  MOV       A,M
F42B 12                  STAX      D
F42C 23          RS8:    INX       H         ;SAME THING FOR OTHER
                         DJNZ      RS7       ; BREAKPOINT
F42D+10F1
                         EXAF                ;NOW SAVE THE Z-80 UNIQUES
F42F+08
                         EXX
F430+D9
F431 E5                  PUSH      H
F432 D5                  PUSH      D
F433 C5                  PUSH      B
F434 F5                  PUSH      PSW
                         PUSHIX
F435+DDE5
                         PUSHIY
F437+FDE5
                         LDAI
F439+ED57
F43B 47                  MOV       B,A
                         LDAR
F43C+ED5F
F43E 4F                  MOV       C,A
F43F C5                  PUSH      B
F440 C313F1              JMP       WINITA    ;RETURN TO MONITOR
F443 E5          RS9:    PUSH      H         ;RET BREAKPOINT ENCOUNTERED, ADJUST TH
F444 CF                  RST       1         ;DO THE BREAKPOINT
                 ;
F445 C1          EXIT:   POP       B
F446 79                  MOV       A,C
                         STAR
F447+ED4F
F449 78                  MOV       A,B
                         STAI
F44A+ED47
                         POPIX
F44C+DDE1
                         POPIY
F44E+FDE1
F450 F1                  POP       PSW
F451 C1                  POP       B
F452 D1                  POP       D
F453 E1                  POP       H
                         EXAF
F454+08
                         EXX
F455+D9
F456 D1                  POP       D
F457 C1                  POP       B
F458 F1                  POP       PSW
F459 E1                  POP       H
F45A F9                  SPHL
F45B 00                  DB        0         ;PLACE FOR EI
```

```
CP/M MACRO ASSEM 2.0     #016     MOSS 2.2 MONITOR

 F45C 210000              LXI     H,0
 F45F C30000              JMP     0
 F462 =           ENDX:   EQU     $
                  ;
                  ; ERROR HANDLERS
                  ;
                  ;       THREE TYPES OF ERRORS ARE DETECTED:  A RESTART
                  ;       ERROR; AN I/O ASSIGNMENT ERROR; AND CERTAIN PROGRAM
                  ;       ERRORS (DETERMINED BY THE PARTICULAR ROUTINE WHERE
                  ;       THE ERROR CONDITION WAS ENCOUNTERED.)  EACH CAUSES
                  ;       A UNIQUE MESSAGE TO BE PRINTED, THEN DOES A WARM
                  ;       INITIALIZATION OF THE MONITOR.  THE I/O ERROR
                  ;       CAUSES THE I/O ASSIGNMENTS TO BE RESET TO DEFAULT ASSI
                  ;
 F462 AF          IOER:   XRA     A        ;SET IOBYTE TO DEFAULT VALUE
 F463 320300              STA     IOBYTE
 F466 216CF4              LXI     H,IOMSG  ;GET ADDRESS OF I/O ERROR MSG
 F469 C3B5F6              JMP     COMERR   ;GO PROCESS IT
 F46C 492F4F2045IOMSG:    DB      'I/O ER','R'+80H
                  ;
                  ; BYTE ROUTINE READS TWO ASCII CHARACTERS FROM THE
                  ;    CURRENT PAPER TAPE READER AND ASSEMBLES THEM INTO TWO
                  ;    HEXADECIMAL BYTES OF DATA.  IT UPDATES A CHECKSUM
                  ;    ACCUMULATED IN REGISTER D.
                  ;
 F473 CDE8F6      BYTE:   CALL    BYT      ;GET NEXT BYTE
 F476 B0                  ORA     B        ;COMBINE THEM
 F477 47                  MOV     B,A
 F478 82                  ADD     D        ;UPDATE CHECKSUM
 F479 57                  MOV     D,A
 F47A 78                  MOV     A,B      ;RESTORE BYTE
 F47B C9                  RET
                  ;
 F47C 0E0D        PEOL:   MVI     C,CR
 F47E CD7CF6              CALL    PO
 F481 0E0A                MVI     C,LF
 F483 C37CF6              JMP     PO       ;GO PUNCH THE OUTPUT
                  ;
                  ; RIX ROUTINE READS ONE CHARACTER FROM THE CURRENT
                  ;    PAPER TAPE READER AND STRIPS OFF THE PARITY BIT.
                  ;
 F486 CD56F6      RIX:    CALL    RI
 F489 E67F                ANI     7FH
 F48B C9                  RET
                  ;
 F48C 3F3F3FBF    QMSG:   DB      '???','?'+80H
 F490 4D4F535320LOGMSG:   DB      'MOSS VERS 2.2'
 F49D 0D8A                DB      CR,LF+80H
                  ;
                  ; INITIALIZATION CODE FOR THE 8250 ASYNCHRONOUS COMMUNICATION
                  ;    ELEMENT.  THIS CODE WILL INITIALIZE THE BAUD RATE OF THE
                  ;    8250, AS WELL AS THE WORD FORMAT.  8 DATA BITS, 1 STOP BIT
                  ;    AND NO PARITY ARE SELECTED.  EITHER 2 OR 3 CARRIAGE RETURN
                  ;    MUST BE ENTERED TO ESTABLISH THE CORRECT BAUD RATE.
                  ;
 F49F 3E0F        I8250:  MVI     A,0FH    ;SET UP THE 8250
 F4A1 D324                OUT     SMDMCT
 F4A3 114000              LXI     D,40H    ;SET UP TO TIME THE START BIT
 F4A6 62                  MOV     H,D
 F4A7 6A                  MOV     L,D      ;ZEROES TO (H,L)
 F4A8 DB26        I8250A: IN      SMDMST   ;WAIT FOR START BIT
 F4AA A3                  ANA     E
                          JRZ     I8250A
 F4AB+28FB
 F4AD DB26        I8250B: IN      SMDMST   ;NOW, TIME THE START BIT DURATION
 F4AF 23                  INX     H
 F4B0 A3                  ANA     E
 F4B1 A3                  ANA     E
```

```
CP/M MACRO ASSEM 2.0    #017    MOSS 2.2 MONITOR

 F4B2 C2ADF4                JNZ     I8250B
 F4B5 E5                    PUSH    H           ;SAVE COUNT IN CASE OF 4 MHZ
 F4B6 29                    DAD     H           ;PREPARE THE 2 MHZ DIVISOR
 F4B7 5C                    MOV     E,H         ;SET UP THE FUDGE FACTOR
 F4B8 19                    DAD     D           ;APPLY THE FUDGE FACTOR
 F4B9 19                    DAD     D
 F4BA E5                    PUSH    H           ;SAVE FOR LATER USE
 F4BB 29                    DAD     H           ;WAIT FOR 8 BIT TIMES
 F4BC 29                    DAD     H
 F4BD DB20        I8250C:   IN      SDATA       ;WASTE SOME TIME
 F4BF 2B                    DCX     H
 F4C0 7D                    MOV     A,L
 F4C1 B4                    ORA     H
 F4C2 C2BDF4                JNZ     I8250C
 F4C5 E1                    POP     H           ;REGET 2 MHZ DIVISOR
 F4C6 3E83        I8250D:   MVI     A,83H       ;SET DIVISOR REGISTER ACCESS
 F4C8 D323                  OUT     SLCTRL
 F4CA 7D                    MOV     A,L         ;SET THE DIVISOR
 F4CB D320                  OUT     SDATA
 F4CD 7C                    MOV     A,H
 F4CE D321                  OUT     SINTEN
 F4D0 3E03                  MVI     A,3         ;SET DATA REGISTER ACCESS
 F4D2 D323                  OUT     SLCTRL
 F4D4 AF                    XRA     A           ;DISABLE INTERRUPTS
 F4D5 D321                  OUT     SINTEN
 F4D7 D325                  OUT     SLSTAT      ;AND RESET ERROR FLAGS
 F4D9 CDCEF6                CALL    TTYIN       ;GET A CHARACTER
 F4DC E67F                  ANI     7FH         ;STRIP OFF ANY PARITY BIT
 F4DE FE0D                  CPI     0DH         ;SEE IF IT IS A CARRIAGE RETURN
 F4E0 E1                    POP     H           ;SET THE STACK STRAIGHT
 F4E1 C8                    RZ                  ;DONE IF CARRIAGE RETURN RECEIVED
 F4E2 5D                    MOV     E,L         ;ELSE, MUST BE 4 MHZ SYSTEM
 F4E3 54                    MOV     D,H         ; SO, COUNT=COUNT*5/4
 F4E4 CDEEF4                CALL    DIV2
 F4E7 CDEEF4                CALL    DIV2
 F4EA 19                    DAD     D
 F4EB E5                    PUSH    H
                            JMPR    I8250D      ;GO SET THE NEW DIVISOR
 F4EC+18D8
                  ;
                  ;
 F4EE B7          DIV2:     ORA     A           ;CLEAR THE CARRY BIT
 F4EF 7C                    MOV     A,H         ;DO A 16-BIT RIGHT SHIFT
 F4F0 1F                    RAR
 F4F1 67                    MOV     H,A
 F4F2 7D                    MOV     A,L
 F4F3 1F                    RAR
 F4F4 6F                    MOV     L,A
 F4F5 C9                    RET
                  ;
                  ; EOF ROUTINE PUNCHES AN END OF FILE RECORD (INTEL HEX
                  ;    FORMAT) ONTO THE CURRENTLY ASSIGNED PAPER TAPE PUNCH
                  ;    DEVICE.  AN ENTRY POINT ADDRESS FOR THE FILE WILL ALSO
                  ;    BE PUNCHED, IF SPECIFIED.
                  ;
 F4F6 CDA4F6      EOF:      CALL    EXLF        ;GET JUMP ADDRESS
 F4F9 D5                    PUSH    D           ;SAVE THE # OF TRAILER NULLS
 F4FA CDC8F5      EOFA:     CALL    PSOR        ;PUNCH START OF RECORD
 F4FD AF                    XRA     A           ;ZERO OUT THE CHECKSUM
 F4FE 57                    MOV     D,A
 F4FF CDF6F6                CALL    PBADR       ;OUTPUT THE RECORD LENGTH AND EP
 F502 3E01                  MVI     A,1         ;PUNCH RECORD TYPE = 1
 F504 CDFEF6                CALL    PBYTE
 F507 AF                    XRA     A
 F508 92                    SUB     D           ;OUTPUT THE CHECKSUM
 F509 CDFEF6                CALL    PBYTE
                            JMPR    LEO         ;GO DO THE TRAILER
 F50C+1803
```

```
                    ;
                    ; LEADER ROUTINE "PUNCHES" SIX INCHES (OR AS SPECIFIED)
                    ;   OF LEADER ON THE PAPER TAPE PUNCH.  NULLS ARE PUNCHED
                    ;   TO FORM THE LEADER (OR TRAILER).
                    ;
F50E CDD7F0         LEADER: CALL    EXPR1   ;SEE IF SOME OTHER LENGTH WANTED
F511 C1             LEO:    POP     B       ;GET THE VALUE
F512 78                     MOV     A,B
F513 B1                     ORA     C       ;TEST FOR DEFAULT SELECT
F514 41                     MOV     B,C     ;MOVE NEW VALUE IN JUST IN CASE
F515 0E00                   MVI     C,0     ;GET A NULL CHARACTER
                            JRNZ    LE1     ;JUMP IF NEW VALUE WANTED
F517+2002
F519 063C                   MVI     B,60    ;DEFAULT, SET 60 NULLS
F51B CD0CF0         LE1:    CALL    PUNCH   ;PUNCH ONE NULL
                            DJNZ    LE1     ;KEEP GOING TIL DONE
F51E+10FB
F520 C9                     RET
                    ;
                    ; QUERY ROUTINE WILL TELL THE OPERATOR WHAT HIS CURRENT LOGICA
                    ;       PHYSICAL PERIPHERAL DEVICE ASSIGNMENTS ARE.  NO PARAME
                    ;       (OTHER THAN A CARRIAGE RETURN) ARE REQUIRED ON ENTRY.
                    ;
F521 3A0300         QUERY:  LDA     IOBYTE  ;GET THE ASSIGNMENT CONTROL BYTE
F524 0604                   MVI     B,4     ;SET UP FOR FOUR LOGICAL DEVICES
F526 217DF1                 LXI     H,ACT   ;ADDRESS OF CONVERSION TABLE
F529 11FBFF                 LXI     D,ALT-APT  ;NEGATIVE OFFSET FOR LOGICAL TABLE
F52C F5             QUE1:   PUSH    PSW
F52D CDFEF5                 CALL    BLK     ;FORMAT THE PRINT-OUT
F530 4E                     MOV     C,M     ;GET THE CURRENT LOGICAL DEVICE CODE
F531 CD09F0                 CALL    CONOUT  ;OUTPUT IT
F534 CDF7F5                 CALL    DASH    ;OUTPUT A DASH
F537 F1                     POP     PSW     ;REGET THE CONTROL BYTE
F538 F5                     PUSH    PSW     ;RESAVE IT
F539 E5                     PUSH    H       ;SAVE THE TABLE POINTER
F53A 23             QUE2:   INX     H       ;ADJUST POINTER TO CURRENT PHYSICAL DE
F53B 3C                     INR     A
F53C E603                   ANI     3       ;BITS 0 AND 1 ARE 0 WHEN ON CURRENT AS
                            JRNZ    QUE2    ;NOT THERE YET, TRY AGAIN
F53E+20FA
F540 4E                     MOV     C,M     ;FOUND IT, NOW PRINT IT
F541 CD09F0                 CALL    CONOUT
F544 E1                     POP     H
F545 F1                     POP     PSW     ;GO TO NEXT LOGICAL DEVICE
F546 1F                     RAR             ;ADJUST THE IOBYTE
F547 1F                     RAR
F548 19                     DAD     D       ;ADJUST THE TABLE POINTER
                            DJNZ    QUE1    ;GO DO NEXT LOGICAL DEVICE
F549+10E1
F54B C9                     RET             ;RETURN TO MONITOR
                    ;
                    ; READ ROUTINE READS AN INTEL HEX FORMAT PAPER TAPE FROM
                    ;   THE CURRENT PAPER TAPE READER.  IF A NON-ZERO ADDRESS
                    ;   IS SPECIFIED IN THE END OF FILE RECORD, CONTROL WILL
                    ;   BE TRANSFERRED TO THAT ADDRESS.  OTHERWISE, CONTROL
                    ;   WILL REVERT TO THE EXECUTIVE.
                    ;
F54C CDD7F0         READ:   CALL    EXPR1   ;GET OFFSET BIAS
F54F E1             REDO:   POP     H       ;  INTO (H,L)
F550 E5                     PUSH    H       ;SAVE THE BIAS
F551 CD86F4         RED1:   CALL    RIX     ;READ A BYTE
F554 DE3A                   SBI     ':'     ;LOOK FOR START OF RECORD
                            JRNZ    RED1    ;JUMP TO KEEP LOOKING
F556+20F9
F558 57                     MOV     D,A     ;INITIALIZE CHECKSUM
F559 CD73F4                 CALL    BYTE    ;GET RECORD LENGTH
                            JRZ     RED3    ;JUMP IF EOF RECORD
F55C+2823
```

```
CP/M MACRO ASSEM 2.0    #019    MOSS 2.2 MONITOR

F55E 5F                 MOV     E,A     ;ELSE, ASSUME DATA RECORD
F55F CD73F4             CALL    BYTE    ;GET LOAD ADDRESS HIGH BYTE
F562 F5                 PUSH    PSW     ;SAVE IT
F563 CD73F4             CALL    BYTE    ;GET LOAD ADDRESS LOW BYTE
F566 C1                 POP     B       ;BUILD ADDRESS IN (B,C)
F567 4F                 MOV     C,A
F568 09                 DAD     B       ;ADD ON THE BIAS
F569 CD73F4             CALL    BYTE    ;SKIP OVER RECORD TYPE
F56C CD73F4     RED2:   CALL    BYTE    ;GET A DATA BYTE
F56F 77                 MOV     M,A     ;PUT IT INTO MEMORY
F570 2F                 CMA             ;DO A QUICK CHECK
F571 AE                 XRA     M       ;  RESULT SHOULD BE ZERO
F572 C4A1F2             CNZ     BITS    ;IF ERROR, PRINT ADDRESS AND DATA
F575 23                 INX     H       ;INCREMENT MEMORY POINTER
F576 1D                 DCR     E       ;RECORD LENGTH FOR LOOP CONTROL
                        JRNZ    RED2    ;DO REST OF THE RECORD
F577+20F3
F579 CD73F4             CALL    BYTE    ;GET THE CHECKSUM
F57C C209F1             JNZ     QPRT    ;ABORT IF ERROR
                        JMPR    REDO    ;GO DO NEXT RECORD
F57F+18CE
F581 CD73F4     RED3:   CALL    BYTE    ;EOF RECORD, GET ENTRY POINT
F584 67                 MOV     H,A     ;HIGH BYTE TO (H)
F585 CD73F4             CALL    BYTE    ;GET THE LOW BYTE
F588 6F                 MOV     L,A
F589 B4                 ORA     H       ;SEE IF IT IS ZERO
F58A D1                 POP     D       ;RESTORE THE STACK
F58B C8                 RZ              ;RETURN TO MONITOR IF EP=0
F58C E9                 PCHL            ;ELSE, GO TO THE ENTRY POINT
                ;
                ; WRITE ROUTINE IS USED TO PUNCH AN INTEL HEX FORMAT
                ;   PAPER TAPE ON THE CURRENT ASSIGNED PUNCH UNIT.
                ;
F58D CD86F3     WRITE:  CALL    EXPR3   ;GET 3 PARAMETERS, DO CRLF
F590 AF                 XRA     A       ;SEE IF RECORD LENGTH CHANGE
F591 47                 MOV     B,A     ;SET HIGH BYTE TO ZERO
F592 B1                 ORA     C       ;NOW SEE IF CHANGE WANTED
                        JRNZ    WRI1    ;YES, JUMP AND SET IT UP
F593+2002
F595 0E10               MVI     C,16    ;NO, DEFAULT TO 16 BYTES/RECORD
F597 E5         WRI1:   PUSH    H       ;SAVE MEMORY POINTER
F598 09                 DAD     B       ;ADD THE RECORD LENGTH
F599 B7                 ORA     A       ;CLEAR THE CARRY BIT
                        DSBC    D       ;SEE IF FULL RECORD REMAINS
F59A+ED52
F59C E1                 POP     H       ;RESTORE (H,L)
                        JRC     WRI2    ;GO DO A FULL RECORD
F59D+380A
F59F D5                 PUSH    D       ;SAVE LAST BYTE ADDRESS
F5A0 EB                 XCHG            ;SWAP (D,E) AND (H,L)
F5A1 B7                 ORA     A       ;RESET THE CARRY BIT
                        DSBC    D       ;FIND # OF BYTE REMAINING
F5A2+ED52
F5A4 23                 INX     H       ;ADJUST TO INCLUDE LAST BYTE
F5A5 E3                 XTHL            ;SWAP TOP OF STACK
F5A6 EB                 XCHG            ;SET (D,E), (H,L) TO NORMAL
F5A7 C1                 POP     B       ;NEW RECORD LENGTH TO (B,C)
F5A8 D8                 RC              ;DONE IF ZERO LENGTH RECORD
F5A9 C5         WRI2:   PUSH    B       ;SAVE LOOP COUNT
F5AA D5                 PUSH    D
F5AB 50                 MOV     D,B     ;ZERO THE CHECKSUM
F5AC 41                 MOV     B,C     ;MOVE LOOP CONTROL TO B
F5AD CDC8F5             CALL    PSOR    ;PUNCH START OF RECORD
F5B0 78                 MOV     A,B     ;GET RECORD LENGTH
F5B1 CDF6F6             CALL    PBADR   ;PUNCH IT
F5B4 AF                 XRA     A       ;PUNCH RECORD TYPE '0'
F5B5 CDFEF6             CALL    PBYTE
F5B8 7E         WRI3:   MOV     A,M     ;GET NEXT DATA BYTE
```

```
CP/M MACRO ASSEM 2.0     #020     MOSS 2.2 MONITOR

 F5B9 23                  INX     H        ;BUMP THE POINTER
 F5BA CDFEF6              CALL    PBYTE    ;PUNCH THE DATA
                          DJNZ    WRI3     ;DO REST OF RECORD
 F5BD+10F9
 F5BF AF                  XRA     A        ;NOW, DO THE CHECKSUM
 F5C0 92                  SUB     D
 F5C1 CDFEF6              CALL    PBYTE    ;PUNCH IT
 F5C4 D1                  POP     D        ;RESTORE THE REGISTERS
 F5C5 C1                  POP     B
                          JMPR    WRI1     ;GO DO NEXT RECORD
 F5C6+18CF
                 ;
 F5C8 CD7CF4     PSOR:    CALL    PEOL
 F5CB 0E3A                MVI     C,':'
 F5CD C37CF6              JMP     PO
                 ;
                 ; HEXN ROUTINE
                 ;
                 ;THIS ROUTINE ADDS AND SUBTRACTS TWO HEXADECIMAL 16-BIT
                 ;        UNSIGNED NUMBERS AND DISPLAYS THE RESULTS ON THE
                 ;        CONSOLE.
                 ;
 F5D0 CDA4F6     HEXN:    CALL    EXLF     ;GET THE TWO NUMBERS
 F5D3 E5                  PUSH    H        ;SAVE IT FOR THE SUBTRACT
 F5D4 19                  DAD     D        ;ADD THEM
 F5D5 CDFBF5              CALL    LADRB    ;OUTPUT THEM
 F5D8 E1                  POP     H        ;REGET THE FIRST NUMBER
 F5D9 B7                  ORA     A        ;CLEAR THE CARRY BIT
                          DSBC    D        ;DO THE SUBTRACT
 F5DA+ED52
                          JMPR    LADR     ;GO OUTPUT THE RESULT
 F5DC+1803
                 ;
                 ; ROUTINE LADR PRINTS THE CONTENTS OF (H,L) ON THE
                 ;        CURRENT CONSOLE, EITHER AT THE START OF A NEW
                 ;        LINE (EP = LADRA) OR AT THE CURRENT LOCATION (EP
                 ;        = LADR).
                 ;
 F5DE CDA9F6     LADRA:   CALL    CRLF     ;START A NEW LINE
 F5E1 7C         LADR:    MOV     A,H      ;GET HIGH TWO DIGITS
 F5E2 CDE6F5              CALL    HEX1     ;PRINT THEM
 F5E5 7D                  MOV     A,L      ;GET LOW TWO DIGITS
 F5E6 F5         HEX1:    PUSH    PSW      ;SAVE THE LOW DIGIT
 F5E7 0F                  RRC              ;PUT HIGH NIBBLE INTO BITS 0-3
 F5E8 0F                  RRC
 F5E9 0F                  RRC
 F5EA 0F                  RRC
 F5EB CDEFF5              CALL    HEX2     ;GO PRINT SINGLE DIGIT
 F5EE F1                  POP     PSW      ;REGET THE LOW DIGIT
 F5EF CD6EF3     HEX2:    CALL    CONV     ;GO INSERT ASCII ZONE
                          JMPR    CO       ;DO THE CHARACTER OUTPUT
 F5F2+180C
                 ;
                 ; ROUTINE DASH TYPES A DASH ON THE CURRENT CONSOLE DEVICE.
                 ;
 F5F4 CDE6F5     DASH1:   CALL    HEX1     ;FIRST, PRINT ACCUM AS TWO HEX DIGITS
 F5F7 0E2D       DASH:    MVI     C,'-'    ;GET AN ASCII DASH
                          JMPR    CO       ;GO TYPE IT
 F5F9+1805
                 ;
                 ;  IOBYTE HANDLERS
                 ;
 F5FB                     ORG     MOSS+5FBH
                 ;
 F5FB CDDEF5     LADRB:   CALL    LADRA    ;OUTPUT (H,L) AS 4 ASCII DIGITS
                 ;
 F5FE 0E20       BLK:     MVI     C,' '    ;OUTPUT A BLANK
                 ;
```

```
CP/M MACRO ASSEM 2.0     #021     MOSS 2.2 MONITOR

 F600 3A0300      CO:     LDA     IOBYTE
 F603 E603                ANI     3         ;ISOLATE CONSOLE ASGT
 F605 CADEF6              JZ      TTYOUT    ;TTY DEVICE ACTIVE
 F608 FE02                CPI     2
 F60A FA62F4              JM      CRTOUT    ;CRT ACTIVE
 F60D C262F4              JNZ     CUSO1     ;USER CONSOLE 1 ACTIVE
                  ;
 F610 3A0300      LO:     LDA     IOBYTE
 F613 E6C0                ANI     0C0H      ;ISOLATE LIST ASGT
 F615 CADEF6              JZ      TTYOUT    ;TTY DEVICE ACTIVE
 F618 FE80                CPI     80H
 F61A FA62F4              JM      CRTOUT    ;CRT ACTIVE
 F61D CA62F4              JZ      LPRT      ;LINE PRINTER ACTIVE
 F620 C362F4              JMP     LUSE1     ;USER PRINTER 1 ACTIVE
                  ;
 F623 3A0300      CSTS:   LDA     IOBYTE
 F626 E603                ANI     3         ;ISOLATE CONSOLE ASGT
 F628 CAC6F6              JZ      TTST      ;TTY ACTIVE
 F62B FE02                CPI     2
 F62D FA62F4              JM      CRTST     ;CRT ACTIVE
 F630 C262F4              JNZ     CUST1     ;USER CONSOLE 1 ACTIVE
                  ;
 F633 3A0300      BATST:  LDA     IOBYTE
 F636 E60C                ANI     0CH       ;ISOLATE BATCH ASGT
 F638 CAC6F6              JZ      TTST      ;TTY ACTIVE
 F63B FE08                CPI     8
 F63D FA62F4              JM      PTRST     ;PAPER TAPE READER ACTIVE
 F640 CA62F4              JZ      RUST1     ;USER READER 1 ACTIVE
 F643 C362F4              JMP     RUST2     ;USER READER 2 ACTIVE
                  ;
 F646 3A0300      CI:     LDA     IOBYTE
 F649 E603                ANI     3         ;ISOLATE CONSOLE ASGT
 F64B CACEF6              JZ      TTYIN     ;TTY DEVICE ACTIVE
 F64E FE02                CPI     2
 F650 FA62F4              JM      CRTIN     ;CRT ACTIVE
 F653 C262F4              JNZ     CUSI1     ;USER CONSOLE 1 ACTIVE
                  ;
 F656 3A0300      RI:     LDA     IOBYTE
 F659 E60C                ANI     0CH       ;ISOLATE BATCH ASGT
 F65B CACEF6              JZ      TTYRDR    ;TTY ACTIVE
 F65E FE08                CPI     8
 F660 FA62F4              JM      PTRIN     ;PAPER TAPE READER ACTIVE
 F663 CA62F4              JZ      RUSI1     ;USER READER 1 ACTIVE
 F666 C362F4              JMP     RUSI2     ;USER READER 2 ACTIVE
                  ;
 F669 3A0300      LSTAT:  LDA     IOBYTE
 F66C E6C0                ANI     0C0H      ;ISOLATE THE LIST DEVICE ASSIGNMENT
 F66E CAD6F6              JZ      TTOST
 F671 FE80                CPI     80H
 F673 FA62F4              JM      CRTOST
 F676 CA62F4              JZ      LPRST
 F679 C362F4              JMP     LUST1
                  ;
 F67C 3A0300      PO:     LDA     IOBYTE
 F67F E630                ANI     30H       ;ISOLATE PUNCH ASGT
 F681 CADEF6              JZ      TTPNCH    ;TTY ACTIVE
 F684 FE20                CPI     20H
 F686 FA62F4              JM      HSP       ;HIGH SPEED PUNCH ACTIVE
 F689 CA62F4              JZ      PUSO1     ;USER PUNCH 1 ACTIVE
 F68C C362F4              JMP     PUSO2     ;USER PUNCH 2 ACTIVE
                  ;
                  ; ROUTINE CONI READS THE CONSOLE AND STRIPS OFF THE ASCII
                  ;       PARITY BIT.
                  ;
 F68F CD46F6      CONI:   CALL    CI        ;GET THE NEXT CHARACTER
 F692 E67F                ANI     7FH       ;STRIP OFF THE PARITY BIT
 F694 C9          RTS:    RET
                  ;
```

```
CP/M MACRO ASSEM 2.0    #022    MOSS 2.2 MONITOR

                       ; ROUTINE PRTWD PRINTS AN ASCII STRING ONTO THE CONSOLE.
                       ;       THE STRING MUST BE TERMINATED BY BIT 7 SET IN THE
                       ;       LAST CHARACTER OF THE STRING.  THE STRING WILL START
                       ;       A NEW LINE (EP = PRTWD) OR CONTINUE ON THE SAME
                       ;       LINE (EP = PRTWA)
                       ;
F695 CDA9F6            PRTWD:  CALL    CRLF     ;START A NEW LINE
F698 C5                PRTWA:  PUSH    B        ;SAVE (B,C)
F699 4E                PRTA:   MOV     C,M      ;GET NEXT CHARACTER FROM MEMORY
F69A CD00F6                    CALL    CO       ;OUTPUT IT
F69D 23                        INX     H        ;INCREMENT MEMORY POINTER
F69E 79                        MOV     A,C
F69F 07                        RLC              ;TEST FOR BIT 7 DELIMITER
                               JRNC    PRTA     ;NO DELIMITER, GO DO NEXT CHARACTER
F6A0+30F7
F6A2 C1                PRTB:   POP     B        ;RESTORE (B,C)
F6A3 C9                        RET
                       ;
                       ; ROUTINE EXLF READS TWO PARAMETERS, PUTS THEM INTO THE
                       ;       D,E AND H,L REGISTERS, THEN DOES A CARRIAGE RETURN,
                       ;       LINE FEED SEQUENCE.
                       ;
F6A4 CDD9F0            EXLF:   CALL    EXPR     ;GO GET TWO PARAMETERS
F6A7 D1                        POP     D
F6A8 E1                        POP     H
                       ;
                       ; ROUTINE CRLF GENERATES A CARRIAGE RETURN, LINE FEED
                       ;       SEQUENCE ON THE CURRENT CONSOLE TO START A NEW LINE
                       ;       IT INCLUDES TRHEE NULL CHARACTERS FOR TTY TYPE
                       ;       DEVICES FOR THE HEAD MOVEMENT TIME.
                       ;
F6A9 E5                CRLF:   PUSH    H        ;SAVE THE CONTENTS OF (H,L)
F6AA 21C2F6            CRLFA:  LXI     H,CRMSG  ;ADDRESS OF CR,LF MESSAGE
F6AD CD98F6                    CALL    PRTWA    ;  OUTPUT IT
F6B0 E1                        POP     H        ;RESTORE (H,L)
F6B1 C9                        RET
                       ;
F6B2 21BBF6            RSTER:  LXI     H,RSTMSG ;GET ADDRESS OF RESTART ERROR MSG
F6B5 CD95F6            COMERR: CALL    PRTWD    ;PRINT IT ON NEW LINE
F6B8 C30000                    JMP     WSVEC    ;GO TO WARM BOOT
                       ;
F6BB 5253542045RSTMSG: DB      'RST ER','R'+80H
F6C2 0D0A0080          CRMSG:  DB      CR,LF,0,80H
                       ;
                       ; I/O DRIVERS FOR THE 8250 ASYNC COMM ELEMENT
                       ;
F6C6 DB25              TTST:   IN      SLSTAT   ;GET 8250 LINE STATUS
F6C8 E601                      ANI     1        ;SEE IF RECEIVE DATA AVAILABLE
F6CA C8                        RZ               ;RETURN IF NOT
F6CB C6FE                      ADI     0FEH     ;FLAG THAT DATA IS AVAILABLE
F6CD C9                        RET
                       ;
F6CE DB25              TTYIN:  IN      SLSTAT   ;GET 8250 LINE STATUS
F6D0 1F                        RAR              ;MOVE RX DATA READY BIT INTO CARRY
                               JRNC    TTYIN    ;LOOP UNTIL DATA IS IN
F6D1+30FB
F6D3 DB20                      IN      SDATA    ;READ THE DATA
F6D5 C9                        RET
                       ;
F6D6 DB25              TTOST:  IN      SLSTAT   ;GET 8250 LINE STATUS
F6D8 E620                      ANI     20H      ;ISOLATE TX BUFFER EMPTY BIT
F6DA C8                        RZ               ;RETURN IF NOT EMPTY
F6DB C6BF                      ADI     0BFH     ;FLAG THE EMPTY STATE
F6DD C9                        RET
                       ;
F6DE DB25              TTYOUT: IN      SLSTAT   ;GET 8250 LINE STATUS
F6E0 E620                      ANI     20H      ;ISOLATE THRE BIT
                               JRZ     TTYOUT   ;WAIT UNTIL ONE OF THE REGISTERS EMPTI
```

```
CP/M MACRO ASSEM 2.0    #023    MOSS 2.2 MONITOR

 F6E2+28FA
 F6E4 79               MOV     A,C      ;MOVE THE DATA OVER
 F6E5 D320             OUT     SDATA    ;OUTPUT THE DATA
 F6E7 C9               RET
                 ;
                 ; EQUATES FOR ADDITIONAL CONSOLE DEVICES
                 ;
 F462 =          CRTIN:  EQU     IOER
 F462 =          CRTOUT: EQU     IOER
 F462 =          CRTST:  EQU     IOER
 F462 =          CRTOST: EQU     IOER     ;UNASSIGNED CRT OUTPUT STATUS
 F462 =          CUSI1:  EQU     IOER     ;UNASSIGNED USER CONSOLE (INPUT)
 F462 =          CUSO1:  EQU     IOER     ;UNASSIGNED USER CONSOLE (OUTPUT)
 F462 =          CUST1:  EQU     IOER
                 ;
                 ;  EQUATES FOR ADDITIONAL PAPER TAPE PUNCH DEVICES
                 ;
 F6DE =          TTPNCH: EQU     TTYOUT   ;UNASSIGNED TELETYPE PUNCH
 F462 =          HSP:    EQU     IOER     ;UNASSIGNED HIGH SPEED PUNCH
 F462 =          HSPST:  EQU     IOER     ;UNASSIGNED HIGH SPEED PUNCH STATUS
 F462 =          PUSO1:  EQU     IOER     ;UNASSIGNED USER PUNCH 1
 F462 =          PUSO2:  EQU     IOER     ;UNASSIGNED USER PUNCH 2
                 ;
                 ;  EQUATES FOR ADDITIONAL LIST DEVICES
                 ;
 F462 =          LPRT:   EQU     IOER     ;UNASSIGNED LINE PRINTER
 F462 =          LPRST:  EQU     IOER     ;UNASSIGNED PRINTER STATUS
 F462 =          LUSE1:  EQU     IOER     ;LIST DEVICE 1
 F462 =          LUST1:  EQU     IOER     ;LIST DEVICE 1 STATUS
                 ;
                 ;  EQUATES FOR ADDITIONAL PAPER TAPE READER DEVICES
                 ;
 F6CE =          TTYRDR: EQU     TTYIN    ;UNASSIGNED TELETYPE PAPER TAPE READER
 F462 =          PTRIN:  EQU     IOER     ;UNASSIGNED HIGH SPEED PAPER TAPE READ
 F462 =          PTRST:  EQU     IOER     ;UNASSIGNED HS PTR STATUS
 F462 =          RUSI1:  EQU     IOER     ;UNASSIGNED PAPER TAPE READER 1
 F462 =          RUST1:  EQU     IOER     ;UNASSIGNED PAPER TAPE READER 1 (STATU
 F462 =          RUSI2:  EQU     IOER     ;UNASSIGNED PAPER TAPE READER 2
 F462 =          RUST2:  EQU     IOER     ;UNASSIGNED PAPER TAPE READER 2 (STATU
                 ;
 F6E8 CDF0F6     BYT:    CALL    RIBBLE   ;READ AND CONVERT ONE CHARACTER
 F6EB 07                 RLC              ;SHIFT INTO HIGH NIBBLE
 F6EC 07                 RLC
 F6ED 07                 RLC
 F6EE 07                 RLC
 F6EF 47                 MOV     B,A      ;SAVE IN B TEMPORARILY
 F6F0 CD86F4     RIBBLE: CALL    RIX      ;READ A CHARACTER
 F6F3 C3B0F3             JMP     NIBBLE   ;GO CONVERT TO HEX DIGIT
                 ;
                 ; PADR ROUTINE PUNCHES (H,L) AS FOUR ASCII CHARACTERS.
                 ;   IT IS USED TO PUT THE ADDRESS INTO AN INTEL HEX
                 ;   FORMAT RECORD.
                 ;
 F6F6 CDFEF6     PBADR:  CALL    PBYTE
 F6F9 7C         PADR:   MOV     A,H
 F6FA CDFEF6             CALL    PBYTE
 F6FD 7D                 MOV     A,L
                 ;
                 ; PBYTE ROUTINE PUNCHES (A) AS TWO ASCII CHARACTERS ON
                 ;   THE CURRENT PUNCH DEVICE.
                 ;
 F6FE F5         PBYTE:  PUSH    PSW      ;SAVE THE BYTE
 F6FF 0F                 RRC              ;DO HIGH NIBBLE FIRST
 F700 0F                 RRC
 F701 0F                 RRC
 F702 0F                 RRC
 F703 CD6EF3             CALL    CONV     ;CONVERT TO ASCII
 F706 CD0CF0             CALL    PUNCH    ;PUNCH IT
```

```
CP/M MACRO ASSEM 2.0      #024     MOSS 2.2 MONITOR

 F709 F1                  POP      PSW      ;GET LOW NIBBLE
 F70A F5                  PUSH     PSW      ;RESAVE FOR CHECKSUM
 F70B CD6EF3              CALL     CONV     ;CONVERT TO ASCII
 F70E CD0CF0              CALL     PUNCH    ;PUNCH IT
 F711 F1                  POP      PSW
 F712 82                  ADD      D        ;UPDATE CHECKSUM
 F713 57                  MOV      D,A
 F714 C9                  RET
                ;
 F715                     END
```

APPENDIX D

PARTS LIST, BOARD LAYOUT, SCHEMATIC, SPECIFICATIONS