```
                                         ;=================================
                                         ;= PRIMER OPERATING SYSTEM v 2.7 =
                                         ;= Copyright 1991-1996 EMAC Inc. =
                                         ;=================================

                                         ;
                                         ;ver 2.3 (6/3/93) Gives access to DDATA, and FUN1 in the MOS services and
                                         ;           adds EPROM burner code. Pressing "Func.", "2" does a CALL 1000
                                         ;           without changing the PC register.
                                         ;
                                         ;ver 2.4 (3/1/94) changes a/d routine to successive approximation algorithm,
                                         ;           more robust UART test, UART loopback test
                                         ;
                                         ;ver 2.5 Ports over better EPROM burner code from EMOS 1.7 and
                                         ;           adds hex download function key 3 to the MOS.
                                         ;
                                         ;ver 2.6 Changed serial protocal to 1 stop bit instead of 2.
                                         ;
                                         ;
                                         ;ver 2.7 (2/13/96) Added menu driven EPROM programmer.
                                         ;
FF00=                        begram:  equ            0ff00h               ; beginning of user ram
FFFF=                        maxram:  equ            0ffffh               ; end of ram

001A=                        sysbyts: equ            25 + 1
FFE5=                        monstk:  equ            maxram - sysbyts     ; start of monitor stack-#
                                                                                        ;  of bytes of sys vars
FFD6=                        userstk: equ            maxram - 0fH - sysbyts
0040=                        maxrom:  equ            40h                  ; maximum high byte of rom address (16k)
0011=                        porta:   equ            11h
0011=                        leds:    equ            11h
0012=                        portb:   equ            12h
0012=                        dip:     equ            12h
0013=                        portc:   equ            13h
0010=                        iocreg:  equ            10h

                                         ; ports to send the frequency
0014=                        fprtlo:     equ         14h
0015=                        fprthi:     equ         15h

                                         ; serial port
0080=                        serdta:     equ         80h
0081=                        sercom      equ         81h

000D=                        cr          equ         13          ; carriage return

                                                     aseg
                                                     org         0
0000   C3         004E                               jmp         start
0003   76         32 2E 37                           db          'v2.7'

                                                     org         8        ; rst1
0008   C3         004E                               jmp         start1
000B   33         2F 39 36                           db          '3/96'

                                                     org         10h      ; rst2
0010   C3         004E                               jmp         start2

                                                     org         18h      ; rst3
0018   C3         004E                               jmp         start3

                                                     org         20h      ; rst4
0020   C3         004E                               jmp         start4

                                                     org         28h      ; rst5
0028   C3         004E                               jmp         start5

                                                     org         2ch      ; 5.5
002C   C3         0042                               jmp         fivhlf   ; the code couldn't be put here because between
                                                                          ; 2ch and 30h there are only 4 bytes, so jump to it

                                                     org         30h      ; rst6
0030   C3         004E                               jmp         start6

                                                     org         34h      ; 6.5 interrupt is used for single stepping
0034   C3         0048                               jmp         sixhlf

                                                     org         38h
0038   C3         3241                               jmp         bpentry  ; when rst7, (software bp) jump to monitor


                                                     org         3ch      ; 7.5
003C   E5                                            push        h        ; save hl
003D   2A         FFE9                               lhld        vec7hlf  ; hl = address of interrupt service routine
0040   E3                                            xthl                 ; put address on stack, and restore hl
0041   C9                                            ret                  ; jump to address on stack

0042   E5                                fivhlf:  push     h        ; save hl
0043   2A         FFE5                            lhld     vec5hlf  ; hl = address of interrupt service routine
0046   E3                                         xthl              ; put address on stack, and restore hl
0047   C9                                         ret               ; jump to address on stack

0048   E5                                sixhlf:  push     h        ; save hl
0049   2A         FFE7                            lhld     vec6hlf  ; hl = address of interrupt service routine
004C   E3                                         xthl              ; put address on stack, and restore hl
004D   C9                                         ret               ; jump to address on stack



                                                  org      4eh      ; Start of primer operating system
004E                         start1:
004E                         start2:
004E                         start3:
004E                         start4:
004E                         start5:
004E                         start6:
004E                         start:
004E   31         FFE5                            lxi      sp,monstk         ; point at monitor stack
0051   3E         0F                              mvi      a,0fh
0053   30                                         sim                        ; disable all interrupts
                                                  ; set up the ppi
0054   3E         CD                              mvi      a,0cdh
0056   D3         10                              out      iocreg
0058   AF                                         xra      a
0059   D3         12                              out      portb
005B   3D                                         dcr      a                 ; store FF to PORTA
005C   D3         11                              out      porta

                                                  ; set up the uart (it may not exist)
005E   AF                                         xra      a
005F   D3         81                              out      sercom
0061   D3         81                                 out      sercom
0063   D3         81                              out      sercom
0065   3E         40                              mvi      a,40h
0067   D3         81                              out      sercom                      ; reset internal
0069   3E         4E                              mvi      a,01001110b    ; 1 stop, no parity, 8 data, 1/16
006B   D3         81                              out      sercom
006D   3E         27                              mvi      a,00100111b    ; set RTS, DTR and enable TX and RX
006F   D3         81                              out      sercom
                                                  ; Initialize the display section of the 8279
0071   3E         00                              mvi      a,0
0073   D3         41                              out      dspcmd
0075   3E         3F                              mvi      a,3fh
0077   D3         41                              out      dspcmd
0079   3E         C1                              mvi      a,0c1h
007B   D3         41                              out      dspcmd

                                                  ; delay for a moment
007D   21         4000                            lxi      h,04000h
0080   CD         0181                            call     dlay
0083   3E         80                              mvi      a,80h
0085   D3         41                              out      dspcmd
0087   C3         2F01                            jmp      moscode           ; jump to start of mos or target app



                                         ; DIGIT2 puts number from rdkey into the low nibble of l after shifting
                                         ;  l 1 nibble left.
008A   47                                digit2:  mov      b,a      ; b= byte from rdkey
```

```
008B   7D                                          mov      a,l      ; shift 1 nibble left
008C   0F                                          rrc
008D   0F                                          rrc
008E   0F                                          rrc
008F   0F                                          rrc               ; shift number into hi nibble
0090   C3         0099                             jmp      twonib

                                          ;Same as digit2 but shift hl left 1 nibble
0093   47                                 digit4:  mov            b,a      ; b= byte from rdkey
0094   29                                          dad            h
0095   29                                          dad            h
0096   29                                          dad            h
0097   29                                          dad            h        ; rotate left 1 nibble ( 4 bits )
0098   7D                                          mov            a,l

0099   E6         F0                      twonib:  ani            0f0h     ; clear lo nibble
009B   B0                                          ora            b        ; put new nibble into low byte
009C   6F                                          mov            l,a      ; save in l
009D   C9                                          ret

                                          ;
                                          ;
                                          ; DISPLAY DRIVERS
                                          ;


009E   17         C1 45 8D 00 00          FUNMSG:     DEFB        17h,0c1h,45h,08dH,0,0   ; "Func.__"

                                                      ; bit patterns for register pairs
00A4   7F         1F CF 9B    regmsg:     defb        07fh,01fh,  0cfh,09bh                ; af,bc
00A8   ED         9F 6F 8B                defb        0edh,09fh,  06fh,08bh                ; de,hl
00AC   DE         3F 3F 9B                defb        0deh,03fh,  03fh,09bh                ; sp,pc
00B0   CF         3F DE 9B                defb        0cfh,03fh,  0deh,09bh                ; bp,sc

00B4   0D         ED                                  defb        0dh,0edh                 ; reg 8 "rd" for ram diagnostics
00B6   CF         0D                                  defb        0cfh,0dh                 ; reg 9 "br" for bad ram
00B8   9F         ED                                  defb        09fh,0edh                ; reg 10 "E.d." for EPROM diagnostics
00BA   CF         9F                                  defb        0cfh,09fh                ; reg 11 "b.E." for bad EPROM
00BC   CF         DE                                  defb        0cfh,0deh                ; reg 12 "b.S." for bad serial
00BE   7B         EB                                  defb        07bh,0ebh                ; reg 13 "N.U." for no UART
00C0   8B         8B                                  defb        8bh,8bh                  ; reg 14 "L.L." for local loopback


                                          ; prints the message "Func."  in the left 4 displays
00C2                          funprnt:
00C2   06         85                                  mvi         b,rgtdsp+5              ; point to left display
00C4   21         009E                                lxi         h,funmsg
00C7   0E         06                                  mvi         c,6                     ; number of bytes in msg
00C9   C3         00DA                                jmp         fun1

                                          ; print the register names in the right two displays
                                          ; Upon entry, a=register number 0-7
                                          ;  af = 0, bc=1, de=2, hl=3, sp=4,pc=5,brk = 6, sc = 7
                                          ; af,bc,hl are used
00CC   06         81                      regprnt:    mvi         b,rgtdsp+1 ; point to the second digit from right
00CE   21         00A4                    regprn1:    lxi         h,regmsg
00D1   07                                             rlc                  ; a=a*2
00D2   85                                             add         l
00D3   6F                                             mov         l,a
00D4   3E         00                                  mvi         a,0
00D6   8C                                             adc         h
00D7   67                                             mov         h,a      ; hl=hl+2*a
00D8   0E         02                                  mvi         c,2      ; print 2 chars

00DA   78                                 fun1:       mov         a,b
00DB   D3         41                                  out         dspcmd   ; select display
00DD   7E                                             mov         a,m      ; get bit map from (hl)
00DE   23                                             inx         h
00DF   D3         40                                  out         dspout   ; output bit pattern to the display
00E1   05                                             dcr         b
00E2   0D                                             dcr         c
00E3   C2         00DA                                jnz         fun1     ; loop until c=0
00E6   C9                                             ret

00E7   F3         60 B5 F4 66 D6          dmap:       defb        0f3h,60h,0b5h,0f4h,66h,0d6h,0d7h,70h,0f7h
00F0   F6         77 C7 93 E5 97                      defb        0f6h,77h,0c7h,93h,0e5h,97h,17h      ; zero thru F

                                          ; This outputs the digit in A (0-f) to the display number in B (80-85 hex)
                                          ; only hl is preserved

00F7   E5                                 digout:     push        h
00F8   21         00E7                                lxi         h,dmap   ; point to bit map table
00FB   85                                             add         l        ; add A to HL
00FC   6F                                             mov         l,a
00FD   3E         00                                  mvi         a,0
00FF   8C                                             adc         h
0100   67                                             mov         h,a      ; hl=hl+a

0101   78                                             mov         a,b      ; b= 80h-85h
0102   D3         41                                  out         dspcmd   ; select display
0104   7E                                             mov         a,m      ; get bit map from (hl)
0105   D3         40                                  out         dspout   ; output bit pattern to the display

0107   E1                                             pop         h
0108   C9                                             ret

0060=                         rdrgtdsp    equ         60h         ; 8279 internal addr of right display for reading
0080=                         rgtdsp      equ         80h         ; 8279 internal address of right display
0041=                         dspcmd      equ         41h         ; i/o addr of display command
0040=                         dspout      equ         40h         ; i/o addr of display output
0040=                         keyin       equ         40h         ; i/o addr of scanned key


                                          ; DDATA displays value in A on the right two displays
0109   06         80                      ddata:      mvi         b,rgtdsp ; select rightmost digit
                                          ; DISBYT displays a byte at the display pair pointed to by B
010B   4F                                 disbyt:  mov            c,a      ; save A in c
010C   E6         0F                                  ani         0fh      ; mask off left nibble
010E   CD         00F7                                call        digout   ; show A on rightmost display
0111   79                                             mov         a,c      ; restore original A
0112   0F                                             rrc
0113   0F                                             rrc
0114   0F                                             rrc
0115   0F                                             rrc                  ; these 4 move left nibble to right
0116   E6         0F                                  ani         0fh      ; mask off left nibble
0118   04                                             inr         b        ; select digit to the left
0119   CD         00F7                                call        digout
011C   C9                                             ret

                                          ; DADDR  displays the address in hl on the left 4 displays
                                          ; de is not changed
011D   06         82                      daddr:      mvi         b,rgtdsp+2
011F   7D                                             mov         a,l
0120   CD         010B                                call        disbyt                  ; display l
0123   04                                             inr         b                       ; move to next digit pair
0124   7C                                             mov         a,h
0125   CD         010B                                call        disbyt                  ; display h
0128   C9                                             ret


                                          ; RDKEY polls the 5.5 interrupt and if high it will get the keypad value and
                                          ; return it in A
                                          ; keys 0-f will return 00-0fh
                                          ; 10h-13h are not used
                                          ; step,func,dec,ent/inc = 14h-17h respectively
                                          ; HL,DE NOT AFFECTED
                                          ; no regs preserved
0014=                         STEP:       EQU         14H
0015=                         FUNC:       EQU         15H
0016=                         DECPC:      EQU         16H
0017=                         ENTER:      EQU         17H

0129   DB         41                      plkpad:     in          dspcmd   ; see if key waiting
012B   E6         07                                  ani         7
012D   C2         013D                                jnz         rdkey1   ; calculate key val if key waiting
0130   3E         FF                                  mvi         a,0ffh   ; indicate no key ready
0132   C9                                             ret                  ; ret if key not waiting

0133   DB         41                      rdkey:      IN          DSPCMD   ; SEE IF A KEY IS WAITING (KEY BUFFER WILL BE > 0)
```

```
0135   E6         07                                  ANI         07H
0137   CA         0133                                jz          rdkey    ; loop until key pressed
013A   CD         016C                                call        beep

013D   3E         40                      rdkey1:     mvi         a,40h
013F   D3         41                                  out         dspcmd
0141   DB         40                                  in          keyin    ; get the scanned key.
                                                      ; bit pattern is  CNTRL,SHFT,3 bits SCAN,3 bits RETURN

0143   E6         3F                                  ani         3fh                     ; bit 7 not used
0145   47                                             mov         b,a                     ; preserve A
0146   E6         38                                  ani         00111000b               ; masking off all but scan and..
0148   0F                                             rrc                                 ; dividing by 2 makes scan=scan*4
0149   4F                                             mov         c,a                     ; c=scan*4
014A   78                                             mov         a,b                     ; A=original scanned key
014B   E6         07                                  ani         00000111b               ; mask all but the value for RETURN
014D   81                                             add         c                       ; add to scan*4
014E   C9                                             ret
                                          ;
                                          ; Sound Port drivers
014F   06         C0                      buzzon:     mvi         b,0c0h
0151   C3         0156                                jmp         sod
0154   06         40                      buzzoff:    mvi         b,40h

0156   20                                 sod:        rim
0157   E6         1F                                  ani         1fh
0159   B0                                             ora         b
015A   30                                             sim
015B   C9                                             ret


                                          ; Send the frequency in HL to sound chip.  HL is limited to <=3fffh
015C   7D                                 sdiv:       mov         a,l
015D   D3         14                                  out         fprtlo
015F   7C                                             mov         a,h
0160   3E         3F                                  mvi         a,3fh
0162   A4                                             ana         h
0163   F6         40                                  ori         40h
0165   D3         15                                  out         fprthi
0167   3E         CD                                  mvi         a,0cdh
0169   D3         10                                  out         iocreg
016B   C9                                             ret

016C   E5                                 beep:       push        h        ; do a beep
016D   21         0200                                lxi         h,0200h
0170   CD         015C                                call        sdiv     ; set the frequency
0173   CD         014F                                call        buzzon
0176   21         3000                                lxi         h,03000h
0179   CD         0181                                call        dlay
017C   CD         0154                                call        buzzoff
017F   E1                                             pop         h
0180   C9                                             ret

0181   2B                                 dlay:       dcx         h
0182   7C                                             mov         a,h
0183   B5                                             ora         l
0184   C2         0181                                jnz         dlay
0187   C9                                             ret




                                                      org         1000h
                                          ; monitor services: jump to the service number held in C
1000                          mservices:
1000   F5                                             push        psw
1001   E5                                             push        h
1002   21         102E                                lxi         h,servtbl
1005   79                                             mov         a,c
1006   FE         25                                  cpi         mxsrvnm  ; if C > mxsrvnm then CY=0
1008   D2         1017                                jnc         serverr  ; if cy = 0 then there is a service error
100B   87                                             add         a        ; service # *2 = index to service table
100C   85                                             add         l
100D   6F                                             mov         l,a
100E   3E         00                                  mvi         a,0
1010   8C                                             adc         h
1011   67                                             mov         h,a      ; hl = hl + c*2
1012   7E                                             mov         a,m
1013   23                                             inx         h
1014   66                                             mov         h,m
1015   6F                                             mov         l,a      ; hl = service address
1016   E9                                             pchl                 ; jmp to hl

1017   E1                                 serverr:    pop         h
1018   F1                                             pop         psw      ; just return if bad service #
1019   C9                                             ret

                                                      org         1020h
1020   11         0304                                lxi         d,0304h  ; This is a ROM resident example program
1023   2E         06                                  mvi         l,06h    ; which the student may view and run
1025   63                                             mov         h,e
1026   24                                             inr         h
1027   3E         01                                  mvi         a,1
1029   47                                             mov         b,a
102A   07                                             rlc
102B   4F                                             mov         c,a
102C   97                                             sub         a
102D   FF                                             rst         7

102E   1078 10A2 10B1         servtbl:    dw          serv0,serv1,serv2,serv3,serv4,serv5
103A   1107 111F 114C                     dw          serv6,serv7,serv8,serv9,servA,servB
1046   11CF 11D6 11FC                     dw          servC,servD,servE,servF,serv10,serv11
1052   122C 1237 1245                     dw          serv12,serv13,serv14,serv15,serv16,serv17
105E   126F 13AA 148C                     dw          serv18,serv19,serv1A,serv1B,serv1c,serv1D,serv1E,serv1F
106E   16E0 172E 1767                     dw          serv20,serv21,serv22,serv23,serv24
0025=                         mxsrvnm:    equ         ($ - servtbl) / 2    ; this is the max service number

                                          ; Demo program:  This outputs an increasing frequency while flashing LEDs at
                                          ; an increasing rate.
1078   C5                                 serv0:      push        b
1079   D5                                             push        d
107A   CD         014F                                call        buzzon   ; turn on the speaker
107D   01         3FFF                                lxi         b,3fffh  ; start with long delay and low freq.
1080   16         7F                                  mvi         d,07fh   ; bit pattern to light 1 LED

1082   60                                 serv0a:     mov         h,b
1083   2E         00                                  mvi         l,0
1085   CD         015C                                call        sdiv     ; set the frequency to hl
1088   CD         0181                                call        dlay     ; delay  according to hl
108B   7A                                             mov         a,d      ; get bit pattern
108C   0F                                             rrc                  ; rotate the bit pattern
108D   57                                             mov         d,a      ; save in d again
108E   D3         11                                  out         leds     ; display the bit pattern
1090   05                                             dcr         b        ; increase freq. and decrease delay
1091   C2         1082                                jnz         serv0a   ; loop until bc=0
1094   CD         0154                                call        buzzoff  ; shut off the speaker
1097   3E         FF                                  mvi         a,0ffh   ; bit pattern for all LEDs off
1099   D3         11                                  out         leds     ; turn off the display
109B   D1                                             pop         d
109C   C1                                             pop         b
109D   E1                                             pop         h
109E   F1                                             pop         psw
109F   C9                                             ret

                                          ; This waits for a key to be typed at the terminal and returns it in L
10A0   F5                                 getkey:     push        psw
10A1   E5                                             push        h

10A2   E1                                 serv1:      pop         h        ; restore hl
10A3   DB         81                      srv1a:      in          sercom   ; get serial port status
10A5   E6         02                                  ani         2        ; isolate receive ready bit
10A7   CA         10A3                                jz          srv1a    ; loop until bit set
10AA   DB         80                                  in          serdta   ; get the character
10AC   6F                                             mov         l,a      ; put char in l
10AD   F1                                             pop         psw      ; restore psw
10AE   C9                                             ret

                                          ; This returns L holding a FF if a terminal key was pressed or 0 if not
10AF   F5                                 polkey:     push        psw
```

```
10B0   E5                                             push        h

10B1   E1                                 serv2:      pop         h        ; restore hl
10B2   2E         00                                  mvi         l,0
10B4   DB         81                                  in          sercom
10B6   E6         02                                  ani         2        ; isolate receive ready bit
10B8   CA         10BC                                jz          serv2a   ; if 0, L is correct
10BB   2D                                             dcr         l        ; decrement to 0ffh
10BC   F1                                 serv2a:     pop         psw
10BD   C9                                             ret

                                          ; Same as service 3, but a direct call
10BE   F5                                 conout:     push        psw
10BF   E5                                             push        h
                                          ; This sends the character in E to the terminal display
10C0   DB         81                      serv3:      in          sercom
10C2   E6         01                                  ani         1        ; isolate transmit ready bit
10C4   CA         10C0                                jz          serv3    ; if 0, loop again
10C7   7B                                             mov         a,e      ; put char in a
10C8   D3         80                                  out         serdta   ; output to terminal
10CA   E1                                             pop         h
10CB   F1                                             pop         psw
10CC   C9                                             ret

                                          ;This reads the characters starting at the address in DE and sends them to the
                                          ; terminal until a "$" is encountered.  The "$" is not sent.
10CD   E5                                 pstrng:     push        h
10CE   F5                                             push        psw
10CF   EB                                 serv4:      xchg                 ; hl=de
10D0   7E                                 serv4b:     mov         a,m      ; a = byte from address hl
10D1   FE         24                                  cpi         "$"
10D3   23                                             inx         h        ; point to next char
10D4   CA         10DE                                jz          serv4a   ; exit without transmitting if character was '$'
10D7   5F                                             mov         e,a
10D8   CD         10BE                                call        conout   ; send e to the terminal
10DB   C3         10D0                                jmp         serv4b
10DE   EB                                 serv4a:     xchg                 ; de = hl = pointer to character after the "$"
10DF   E1                                             pop         h
10E0   F1                                             pop         psw
10E1   C9                                             ret

                                          ; same as service 5 only it is a direct call
10E2   F5                                 disp16:     push        psw
10E3   E5                                             push        h
                                          ;
                                          ; Send the unsigned number in DE to the terminal as decimal.
10E4   D5                                 serv5:      push        d
10E5   C5                                             push        b
10E6   EB                                             xchg                 ; hl= de
10E7   06         00                                  mvi         b,0      ; b=0 = number of digits
10E9   11         000A                    serv5a:     lxi         d,10
10EC   CD         114A                                call        div16    ; hl=hl/10decimal
10EF   7B                                             mov         a,e      ; a= remainder (decimal digit)
10F0   C6         30                                  adi      '0'         ; make it an ascii digit
10F2   5F                                             mov         e,a
10F3   D5                                             push        d        ; push the digit
10F4   04                                             inr         b        ; 1 more digit on the stack
10F5   7D                                             mov         a,l
10F6   B4                                             ora         h
10F7   C2         10E9                                jnz         serv5a   ; divide by ten again if hl<>0
                                                      ; now pop off the digits and display them
10FA   D1                                 serv5c:     pop         d
10FB   CD         10BE                                call        conout
10FE   05                                             dcr         b
10FF   C2         10FA                                jnz         serv5c
1102   C1                                             pop         b
1103   D1                                             pop         d
1104   E1                                             pop         h
1105   F1                                             pop         psw
1106   C9                                             ret

                                          ; Send the signed number in DE to the terminal as decimal.
1107                          serv6:      ; check for sign bit
                                                      ; if high, make de 2's complement
                                                      ; send out a '-'
                                                      ; call disp16

1107   7A                                             mov         a,d
1108   B7                                             ora         a
1109   F2         10E4                                jp       serv5       ; display de in decimal if positive
                                                      ; de is negative, show sign and make 2's complement
110C   D5                                             push        d
110D   2F                                             cma
110E   57                                             mov         d,a      ; complement d
110F   7B                                             mov         a,e
1110   2F                                             cma         ; complement a
                                                      ; while E is preserved, output the minus sign
1111   1E         2D                                  mvi         e,'-'    ; minus sign
1113   CD         10BE                                call        conout   ; print '-'

1116   5F                                             mov         e,a      ; save complemented Accum.
1117   13                                             inx         d        ; 2's complement
1118   CD         10E2                                call        disp16   ; display de in decimal
111B   D1                                             pop         d        ; return original value of de
111C   E1                                             pop         h
111D   F1                                             pop         psw
111E   C9                                             ret


                                          ; This multiplies hl*de and returns the result in hl and de with hl being
                                          ; the high word and de being the low word.
111F   E1                                 serv7:      POP         H        ; hl = multiplicand, de = multiplier
1120   C5                                             PUSH        B        ; save bc
1121   44                                             MOV         B,H
1122   7D                                             MOV         A,L
1123   CD         1138                                CALL        smpyx
1126   E5                                             PUSH        H
1127   67                                             MOV         H,A
1128   78                                             MOV         A,B
1129   44                                             MOV         B,H
112A   CD         1138                                CALL        smpyx
112D   D1                                             POP         D
112E   4A                                             MOV         C,D
112F   09                                             DAD         B
1130   CE         00                                  ACI         0
1132   55                                             MOV         D,L
1133   6C                                             MOV         L,H
1134   67                                             MOV         H,A

1135   C1                                             POP         B        ; exit with result in hl:de
1136   F1                                             POP         PSW
1137   C9                                             RET

1138   21         0000                    smpyx:      LXI         H,0
113B   0E         08                                  MVI         C,8
113D   29                                 smpyx1:     DAD         H
113E   17                                             RAL
113F   D2         1145                                JNC         smpyx2
1142   19                                             DAD         D
1143   CE         00                                  ACI         0
1145   0D                                 smpyx2:     DCR         C
1146   C2         113D                                JNZ         smpyx1
1149   C9                                             RET

                                          ; This is the same as service 8
114A   F5                                 div16:      push        psw
114B   E5                                             push        h
                                          ; Divide  HL by DE and return the quotient in HL and remainder in DE
114C                          serv8:
114C   E1                                             POP         H        ; GET THE NUMERATOR
114D   C5                                             PUSH        B        ; SAVE USER'S BC
114E   42                                             MOV         B,D      ; PUT DE IN BC BECAUSE THE ORIGINAL VERSION OF
114F   4B                                             MOV         C,E      ; THIS PROGRAM HAD BC AS THE DIVISOR
1150   11         0000                                LXI         D,0
1153   EB                                             XCHG                 ; DE = HIGH WORD AND HL = LOW WORD

1154   7B                                             MOV         A,E      ; SEE IF DENOMINATOR >= NUMERATOR
1155   91                                             SUB         C
1156   7A                                             MOV         A,D
```

```
1157  98                                     SBB       B
1158  D2       1162                          JNC       SUSLA1
                                             ; IF DENOMINATOR IS > NUMERATOR,  QUOTIENT = 0 AND REMAINDER = NUMERATOR
115B  EB                                     XCHG             ; HL= DE NUMERATOR
115C  11       0000                          LXI       D,0    ; DE= 0
115F  C3       118D                          JMP       susla7
1162  3E       10                  susla1:   MVI       A,16
1164  29                           susla2:   DAD       H
1165  17                                     RAL
1166  EB                                     XCHG
1167  29                                     DAD       H
1168  D2       116D                          JNC       susla3
116B  13                                     INX       D
116C  A7                                     ANA       A
116D  EB                           susla3:   XCHG
116E  1F                                     RAR
116F  F5                                     PUSH      PSW
1170  D2       117C                          JNC       susla4
1173  7D                                     MOV       A,L
1174  91                                     SUB       C
1175  6F                                     MOV       L,A
1176  7C                                     MOV       A,H
1177  98                                     SBB       B
1178  67                                     MOV       H,A
1179  C3       1187                          JMP       susla5
117C  7D                           susla4:   MOV       A,L
117D  91                                     SUB       C
117E  6F                                     MOV       L,A
117F  7C                                     MOV       A,H
1180  98                                     SBB       B
1181  67                                     MOV       H,A
1182  D2       1187                          JNC       susla5
1185  09                                     DAD       B
1186  1B                                     DCX       D
1187  13                           susla5:   INX       D
1188  F1                           susla6:   POP       PSW
1189  3D                                     DCR       A
118A  C2       1164                          JNZ       susla2

118D  EB                           susla7:   XCHG             ; SWAP QUOTIENT AND REMAINDER
118E  C1                                     POP       B
118F  F1                                     POP       PSW
1190  C9                                     RET

                                   ;
                                   ; ADCIN converts a voltage to a 6 bit number in L using successive approx.
                                   ;
0009=                              settle    equ       9       ; settling time for A/D circuit

1191  F5                           adcin:    push      psw
1192  E5                                     push      h

1193  C5                           serv9:    push      b                   ; save bc
1194  0E       06                            mvi       c,6                 ; A/D resolution
1196  06       20                            mvi       b,00100000b         ; b=mask
1198  2E       00                            mvi       l,0                 ; l=result
119A  78                           adcin1:   mov       a,b                 ; test=
119B  B5                                     ora       l                   ; result OR mask
119C  67                                     mov       h,a                 ; save test
119D  D3       13                            out       portc               ; test=>r,2r ladder (D/A)
119F  B7                                     ora       a                   ; rotate mask now...
11A0  78                                     mov       a,b                 ; to allow for op-amp ...
11A1  1F                                     rar                           ; slew (24uS) and...
11A2  47                                     mov       b,a                 ; transistor on/off (10 uS).
11A3  3E       09                            mvi       a,settle            ; settling time
11A5  3D                           adcin2:   dcr       a
11A6  C2       11A5                          jnz       adcin2              ; delay loop
11A9  20                                     rim                           ; get comparison
11AA  B7                                     ora       a                   ; 1 if input<D/A
11AB  FA       11B2                          jm        adcin3              ; if 1 result unchanged
11AE  6C                                     mov       l,h                 ; else result = test
11AF  C3       11B5                          jmp       adcin4              ; see if done
11B2  3E       00                  adcin3:   mvi       a,0                 ; this is dummy code...
11B4  00                                     nop                           ; to balance out timing.
11B5  0D                           adcin4:   dcr       c                   ; dec # of bits
11B6  C2       119A                          jnz       adcin1              ; if not 0, new test
11B9  C1                                     pop       b                   ; restore bc
11BA  7D                                     mov       a,l                 ; a=A/D value
11BB  E1                                     pop       h                   ; restore hl
11BC  6F                                     mov       l,a                 ; return result in l
11BD  F1                                     pop       psw                 ; restore psw
11BE  C9                                     ret


                                   ; returns L with the complemented value of the dip switch
11BF  E1                           servA:    pop       h
11C0  DB       12                            in      dip
11C2  2F                                     cma
11C3  6F                                     mov       l,a
11C4  F1                                     pop       psw
11C5  C9                                     ret

                                   ; Waits for a key press and returns a value from the keypad in L
11C6  E1                           servB:    pop       h
11C7  C5                                     push      b
11C8  CD       0133                          call      rdkey   ; read the keypad
11CB  6F                                     mov       l,a
11CC  C1                                     pop       b
11CD  F1                                     pop       psw
11CE  C9                                     ret

                                   ; Writes the complement of E to portA
11CF  E1                           servC:    pop       h
11D0  7B                                     mov       a,E
11D1  2F                                     cma
11D2  D3       11                            out       porta
11D4  F1                                     pop       psw
11D5  C9                                     ret

                                   ; This prints the hex value of DE to the terminal
                                   ;  Copyright 1990 Softaid Inc. modified by (ME)
11D6                               servD:
11D6  D5                           HEX4:     PUSH      D
11D7  EB                                     XCHG             ; HL = DE
11D8  7C                                     MOV       A,H
11D9  CD       11E4                          CALL      HEX2    ; PRINT MSB
11DC  7D                                     MOV       A,L
11DD  CD       11E4                          CALL      HEX2
11E0  D1                                     POP       D
11E1  E1                                     POP       H
11E2  F1                                     POP       PSW
11E3  C9                                     RET

11E4  F5                           HEX2:     PUSH      psw     ; PRINT LSB
11E5  0F                                     RRC
11E6  0F                                     RRC
11E7  0F                                     RRC
11E8  0F                                     RRC
11E9  E6       0F                            ANI       0FH     ; UPPER 4 BITS
11EB  CD       11F1                          CALL      HXD
11EE  F1                                     POP       PSW
11EF  E6       0F                            ANI       0FH     ; LOWER 4 BITS
                                   ;
                                   ; CONVERT NIBBLE IN A TO ASCII AND DISPLAY IT
                                   ;
11F1  C6       90                  HXD:      ADI       90H     ; SET UP SO A-F MAKE CARRY
11F3  27                                     DAA
11F4  CE       40                            ACI       40H
11F6  27                                     DAA
11F7  5F                                     MOV       E,A
11F8  CD       10BE                          CALL      CONOUT  ; Display digit
11FB  C9                                     RET

                                   ; DACout generates a voltage from the low 6 bits of E
11FC  7B                           servE:    mov       a,E
11FD  E6       3F                            ani       111111b  ; mask off all but lower 6 bits
11FF  D3       13                            out       portc
1201  E1                                     pop       h
1202  F1                                     pop       psw
```

```
1203   C9                                            ret

1204   E1                                 servF:     pop         h
1205   F1                                            pop         psw
1206   C9                                            ret

                                          ; This sets the frequency of the speaker timer according to the value
                                          ; of DE and turns on the speaker.  If de = 0 then the speaker is turned off
1207   D5                                 serv10:    push        d
1208   C5                                            push        b        ; buzzon and buzzoff use B
1209   EB                                            xchg                 ; hl = de
120A   CD         015C                               call        sdiv     ; set the new frequency
120D   7A                                            mov         a,d
120E   B3                                            ora         e
120F   F5                                            push        psw      ; save z flag
1210   C4         014F                               cnz         buzzon
1213   F1                                            pop         psw      ; restore z flag
1214   CC         0154                               cz       buzzoff
1217   C1                                            pop         b
1218   D1                                            pop         d
1219   E1                                            pop         h
121A   F1                                            pop         psw
121B   C9                                            ret

                                          ; This sends the bit pattern in E to the LED display number in D.
                                          ; The leftmost display is 5 and the rightmost is 0
121C   3E         80                      serv11:    mvi         a,rgtdsp
121E   82                                            add         d
121F   FE         86                                 cpi         rgtdsp+6
1221   D2         1229                               jnc         serv11a  ; if > rgtdsp+6 we are out of range
1224   D3         41                                 out         dspcmd   ; select display
1226   7B                                            mov         a,e      ; a = bit pattern
1227   D3         40                                 out         dspout   ; output bit pattern to the display
1229   E1                                 serv11a:   pop         h
122A   F1                                            pop         psw
122B   C9                                            ret

                                          ; display de in the 4 leftmost digits as hex
122C   D5                                 serv12:    push        d
122D   EB                                            xchg                 ; hl=de
122E   C5                                            push        b
122F   CD         011D                               call        daddr
1232   C1                                            pop         b
1233   D1                                            pop         d
1234   E1                                            pop         h
1235   F1                                            pop         psw
1236   C9                                            ret

                                          ; display de in the 4 leftmost digits as decimal (9999 is the max)
1237   C5                                 serv13:    push        b
1238   D5                                            push        d
1239   CD         1765                               call        bin2bcd
123C   EB                                            xchg
123D   CD         011D                               call        daddr    ; print the BCD value of hl
1240   D1                                            pop         d
1241   C1                                            pop         b
1242   E1                                            pop         h
1243   F1                                            pop         psw
1244   C9                                            ret



                                          ; Delay according to the value of hl
1245   E1                                 serv14:    pop         h
1246   E5                                            push        h
1247   CD         0181                               call        dlay
124A   E1                                            pop         h
124B   F1                                            pop         psw
124C   C9                                            ret


                                          ; return the complement of input port B, (same as DIPSWIN)
124D   C3         11BF                    serv15:    jmp         servA

                                          ; KEYSTAT: If no key pressed HL is returned as 0.  If key pressed
                                          ; H is returned as 1 and L with the value of the key
1250   E1                                 serv16:    pop         h
1251   C5                                            push        b
1252   CD         0129                               call        plkpad   ; this returns FF if no key
1255   FE         FF                                 cpi      -1
1257   21         0000                               lxi         h,0      ; assume no key
125A   CA         125F                               jz       sv16ex      ; exit if no key
125D   24                                            inr         h        ; make h=1
125E   6F                                            mov         l,a      ; l = key
125F   C1                                 sv16ex:    pop         b
1260   F1                                            pop         psw
1261   C9                                            ret

                                          ; DIGOUT: Display the hex digit in E on display #D
1262   C5                                 serv17:    push        b
1263   7A                                            mov         a,D      ; put display number in A
1264   C6         80                                 adi         rgtdsp   ; offset from the right display
1266   47                                            mov         b,a      ; B points to the display
1267   7B                                            mov         a,E      ; A is the digit
1268   CD         00F7                               call        digout
126B   C1                                            pop         b
126C   E1                                            pop         h
126D   F1                                            pop         psw
126E   C9                                            ret
                                          ;
                                          ; write to RTC
                                          ;
126F   E1                                 serv18:    POP         H
1270   F1                                            POP         PSW

                                          ;
                                          ; Write to the SMARTCLOCK
                                          ; This must be done in two passes because registers are used to load the data
                                          ;  since the RAM is the only bank available
                                          ; The DE register must point to 1st of 8 bytes to write to RTC.
1271   F5                                 WRSCL:     PUSH        PSW
1272   C5                                            PUSH        B
1273   D5                                            PUSH        D
1274   E5                                            PUSH        H
1275   20                                            RIM
1276   F5                                            PUSH        PSW      ; SAVE IE STATUS
1277   D5                                            PUSH        D
1278   CD         136E                               CALL        CLOCKOFF ; SHUT OFF THE CLOCK WHILE WRITING TO IT
127B   D1                                            POP         D
127C   CD         135E                               CALL        LHLBC    ; LOAD HL AND BC WITH RTC DATA
127F   D5                                            PUSH        D        ; SAVE POINTER TO ARRAY TO PUT RTC DATA
1280   16         41                                 MVI         D,65
1282   3A         FFF5                    WRSCS:     LDA         CLKDUM
1285   15                                            DCR         D
1286   C2         1282                               JNZ         WRSCS

                                                     ; SEND PATTERN RECOGNITION SEQUENCE
1289   16         04                                 MVI         D,4      ; NUMBER OF SWAP/COMPLEMENT OPERATIONS
128B   3E         5C                                 MVI         A,5CH    ; INITIAL VALUE WILL BE CHANGED TO 5C
128D   0F                                 WRC1:      RRC
128E   0F                                            RRC
128F   0F                                            RRC
1290   0F                                            RRC                  ; SWAP NIBBLES
1291   1E         08                      WRC3:      MVI         E,8
1293   32         FFF5                    WRSC0:     STA         CLKDUM
1296   0F                                            RRC
1297   1D                                            DCR         E
1298   C2         1293                               JNZ         WRSC0
129B   B7                                            ORA         A        ; SEE IF BIT 7 IS 0
129C   F2         12A3                               JP          WRC2     ; JUMP IF POSITIVE
129F   2F                                            CMA                  ; NOW COMPLEMENT
12A0   C3         1291                               JMP         WRC3     ; SEND THE COMPLEMENT
12A3   15                                 WRC2:      DCR         D        ; LOOP 4 TIMES
12A4   C2         128D                               JNZ         WRC1

                                                     ; THE PATTERN HAS BEEN SENT SO NOW STORE THE DATA IN THE CLOCK
12A7   16         08                      WRSPAT:    MVI         D,8
12A9   79                                            MOV         A,C      ; STORE HUNDREDS
12AA   32         FFF5                    WRSC2:     STA         CLKDUM
```

```
12AD   0F                                         RRC
12AE   15                                         DCR         D
12AF   C2        12AA                             JNZ         WRSC2

12B2   16        08                               MVI         D,8
12B4   78                                         MOV         A,B      ; STORE SEC
12B5   32        FFF5                 WRSC3:      STA         CLKDUM
12B8   0F                                         RRC
12B9   15                                         DCR         D
12BA   C2        12B5                             JNZ         WRSC3

12BD   16        08                               MVI         D,8
12BF   7D                                         MOV         A,L      ; STORE MIN
12C0   32        FFF5                 WRSC4:      STA         CLKDUM
12C3   0F                                         RRC
12C4   15                                         DCR         D
12C5   C2        12C0                             JNZ         WRSC4

12C8   16        08                               MVI         D,8
12CA   7C                                         MOV         A,H      ; STORE HOUR
12CB   32        FFF5                 WRSC5:      STA         CLKDUM
12CE   0F                                         RRC
12CF   15                                         DCR         D
12D0   C2        12CB                             JNZ         WRSC5

                                                  ; IGNORE THE REST OF THE VARIABLES ON THIS PASS
12D3   06        20                               MVI         B,8*4
12D5   3A        FFF5                 WRSC7:      LDA         CLKDUM
12D8   05                                         DCR         B
12D9   C2        12D5                             JNZ         WRSC7

                                                  ; **PASS 2**
12DC   D1                                         POP         D
12DD   CD        135E                             CALL        LHLBC    ; LOAD HL AND BC WITH RTC DATA
                                                                       ; C = DAY, B= DATE, H = YEAR, L= MONTH

12E0   16        41                               MVI         D,65
12E2   3A        FFF5                 WRSCSA:      LDA        CLKDUM
12E5   15                                         DCR         D
12E6   C2        12E2                             JNZ         WRSCSA

                                                  ; SEND PATTERN RECOGNITION SEQUENCE
12E9   16        04                               MVI         D,4      ; NUMBER OF SWAP/COMPLEMENT OPERATIONS
12EB   3E        5C                               MVI         A,5CH
12ED   0F                             WRC1A:      RRC
12EE   0F                                         RRC
12EF   0F                                         RRC
12F0   0F                                         RRC                  ; SWAP NIBBLES
12F1   1E        08                   WRC3A:      MVI         E,8
12F3   32        FFF5                 WRSC0A:     STA         CLKDUM
12F6   0F                                         RRC
12F7   1D                                         DCR         E
12F8   C2        12F3                             JNZ         WRSC0A
12FB   B7                                         ORA         A        ; SEE IF BIT 7 IS 0
12FC   F2        1303                             JP       WRC2A       ; JUMP IF POSITIVE
12FF   2F                                         CMA                  ; NOW COMPLEMENT
1300   C3        12F1                             JMP         WRC3A    ; SEND THE COMPLEMENT
1303   15                             WRC2A:      DCR         D        ; LOOP 4 TIMES
1304   C2        12ED                             JNZ         WRC1A


                                                  ; SKIP THE FIRST 4 REGISTERS
1307   16        20                   WRSC2A:     MVI         D,8*4
1309   3A        FFF5                 WRSC11:     LDA         CLKDUM
130C   15                                         DCR         D
130D   C2        1309                             JNZ         WRSC11

                                                  ; LOAD THE CLOCK REGISTERS,DAY-DATE-MONTH-YEAR
1310   16        08                               MVI         D,8
1312   79                                         MOV         A,C      ; STORE DAY
1313   F6        30                               ORI         110000B  ; DON'T TURN ON CLOCK YET
1315   32        FFF5                 WRSC8:      STA         CLKDUM
1318   0F                                         RRC
1319   15                                         DCR         D
131A   C2        1315                             JNZ         WRSC8

131D   16        08                               MVI         D,8
131F   78                                         MOV         A,B      ; DATE
1320   32        FFF5                 WRSC9:      STA         CLKDUM
1323   0F                                         RRC
1324   15                                         DCR         D
1325   C2        1320                             JNZ         WRSC9

1328   06        08                               MVI         B,8
132A   7D                                         MOV         A,L      ; STORE MONTH
132B   32        FFF5                 WRSC10:     STA         CLKDUM
132E   0F                                         RRC
132F   05                                         DCR         B
1330   C2        132B                             JNZ         WRSC10

1333   06        08                               MVI         B,8
1335   7C                                         MOV         A,H      ; STORE YEAR
1336   32        FFF5                 WRSC14:     STA         CLKDUM
1339   0F                                         RRC
133A   05                                         DCR         B
133B   C2        1336                             JNZ         WRSC14

133E   79                                         MOV         A,C      ; CHECK OSC BIT
133F   E6        20                               ANI         100000B
1341   C2        1347                             JNZ         WRSC15   ; SKIP IF OSC BIT SET
1344   CD        1369                             CALL        CLOCKON  ; TURN ON THE CLOCK
1347   F1                             WRSC15:     POP         PSW
1348   E6        08                               ANI         1000B    ; CHECK IE STATUS
134A   E1                                         POP         H
134B   D1                                         POP         D
134C   C1                                         POP         B
134D   C2        1352                             JNZ         WRSC12   ; JMP IF EI
1350   F1                                         POP         PSW
1351   C9                                         RET
1352   F1                             WRSC12:     POP         PSW
1353   FB                                         EI
1354   C9                                         RET


                                      ;
1355   C5        3A A3 5C C5 3A       CLKTBL:  DB 0C5H,03AH,0A3H,05CH,0C5H,03AH,0A3H,05CH,0       ; A ZERO ADDED AT END

                                      ; This loads HL BC with the data pointed to by DE and returns DE = DE + 4
135E   EB                             LHLBC:      XCHG                 ; HL POINTS TO RTC DATA
135F   4E                                         MOV         C,M      ; C= HUNDREDS
1360   23                                         INX         H
1361   46                                         MOV         B,M      ; B= SECONDS
1362   23                                         INX         H
1363   5E                                         MOV         E,M
1364   23                                         INX         H
1365   56                                         MOV         D,M      ; D = HOURS, E = MINUTES
1366   23                                         INX         H
1367   EB                                         XCHG                 ; DE = POINTER TO RTC DATA, HL = HRS, MIN
1368   C9                                         RET


                                      ; Turn on the real time clock
1369   06        00                   clockon:    MVI         B,0
136B   C3        1370                             JMP         clk00

                                      ; Turn off the real time clock
136E   06        01                   clockoff: MVI       B,1

1370                             clk00:
1370   16        41                               MVI         D,65
1372   3A        FFF5                 CLK5:       LDA         CLKDUM
1375   15                                         DCR         D
1376   C2        1372                             JNZ         CLK5

                                                  ; SEND OUT THE SMARTWATCH COMPARISON PATTERN
1379   21        1355                             LXI         H,CLKTBL
137C   1E        08                               MVI         E,8
137E   16        08                   clk0:       MVI         D,8
1380   7E                                         MOV         A,(HL)
```

```
1381   32         FFF5                    clk1:       STA         CLKDUM
1384   0F                                             RRC
1385   15                                             DCR         D
1386   C2         1381                                JNZ         clk1
1389   23                                             INX         H
138A   1D                                             DCR         E
138B   C2         137E                                JNZ         clk0

                                                      ; This writes to the OSC bit within the DAY register
138E   16         24                                  MVI         D,8*4 + 4                ; SKIP 4 REGS AND 4 BITS TO GET ON/OFF BIT
1390   3A         FFF5                    CLK3:       LDA         CLKDUM                   ; READ THE BIT
1393   15                                             DCR         D
1394   C2         1390                                JNZ         CLK3

1397   3E         01                                  MVI         A,1
1399   32         FFF5                                STA         CLKDUM                   ; SET THE RST BIT
139C   78                                             MOV         A,B                      ; GET ON/OFF FLAG FROM B REGISTER
139D   32         FFF5                                STA         CLKDUM                   ; AND THE OSC BIT
13A0   16         1A                                  MVI         D,8*3 + 2                ; SKIP 2 BITS OF DAY AND 3 REGS
13A2   3A         FFF5                    CLK4:       LDA         CLKDUM                   ; READ THE BIT
13A5   15                                             DCR         D
13A6   C2         13A2                                JNZ         CLK4
13A9   C9                                             RET


                                          ; READ RTC
13AA   E1                                 serv19:     POP         H
13AB   F1                                             POP         PSW
                                          ; The DE register must be a pointer to an array to store the data from RTC
13AC   F5                                 RDSCLK:     PUSH        PSW
13AD   C5                                             PUSH        B
13AE   D5                                             PUSH        D
13AF   E5                                             PUSH        H
13B0   20                                             RIM
13B1   F3                                             DI
13B2   F5                                             PUSH        PSW       ; SAVE EI STATUS
13B3   D5                                             PUSH        D         ; SAVE POINTER TO RTC ARRAY

                                                      ; We will use the MOS register storage to store the SP reg since it
                                                      ; is only needed when single stepping.
13B4   21         0000                                LXI         H,0       ; SAVE THE STACK POINTER
13B7   39                                             DAD         SP        ; BECAUSE WE NEED ALL THE
13B8   22         FFF3                                SHLD        SPREG     ; REGISTERS WE CAN GET

13BB   16         41                                  MVI         D,65
13BD   3A         FFF5                    RDSCS:      LDA         CLKDUM
13C0   15                                             DCR         D
13C1   C2         13BD                                JNZ         RDSCS

                                                      ; SEND OUT THE SMARTWATCH COMPARISON PATTERN
13C4   21         1355                                LXI         H,CLKTBL
13C7   1E         08                                  MVI         E,8
13C9   16         08                      RDSC0:      MVI         D,8
13CB   7E                                             MOV         A,M
13CC   32         FFF5                    RDSC1:      STA         CLKDUM
13CF   0F                                             RRC
13D0   15                                             DCR         D
13D1   C2         13CC                                JNZ         RDSC1
13D4   23                                             INX         H
13D5   1D                                             DCR         E
13D6   C2         13C9                                JNZ         RDSC0

                                                      ;THE 64 BIT PATTERN HAS BEEN SENT, SO NOW READ THE 64 BITS OF DATA
                                                      ; FIRST IS HUNDREDS THEN SEC, MIN, HOUR, DAY, DATE, MONTH AND YEAR
13D9   16         08                                  MVI         D,8
13DB   1E         00                                  MVI         E,0
13DD   3A         FFF5                    RDSC3:      LDA         CLKDUM
13E0   1F                                             RAR
13E1   7B                                             MOV         A,E
13E2   1F                                             RAR
13E3   5F                                             MOV         E,A       ; E IS THE HUNDREDS
13E4   15                                             DCR         D
13E5   C2         13DD                                JNZ         RDSC3

13E8   16         08                                  MVI         D,8
13EA   0E         00                                  MVI         C,0
13EC   3A         FFF5                    RDSC4:      LDA         CLKDUM
13EF   1F                                             RAR
13F0   79                                             MOV         A,C
13F1   1F                                             RAR
13F2   4F                                             MOV         C,A
13F3   15                                             DCR         D
13F4   C2         13EC                                JNZ         RDSC4     ; C IS THE SECONDS

13F7   16         08                                  MVI         D,8
13F9   06         00                                  MVI         B,0
13FB   3A         FFF5                    RDSC5:      LDA         CLKDUM
13FE   1F                                             RAR
13FF   78                                             MOV       A,B
1400   1F                                             RAR
1401   47                                             MOV       B,A
1402   15                                             DCR         D
1403   C2         13FB                                JNZ         RDSC5     ; B = MINUTES

1406   16         08                                  MVI         D,8
1408   2E         00                                  MVI         L,0
140A   3A         FFF5                    RDSC6:      LDA         CLKDUM
140D   1F                                             RAR
140E   7D                                             MOV         A,L
140F   1F                                             RAR
1410   6F                                             MOV         L,A
1411   15                                             DCR         D
1412   C2         140A                                JNZ         RDSC6     ; THIS WILL BE PUT IN SP LATER

1415   16         08                                  MVI         D,8
1417   26         00                                  MVI         H,0
1419   3A         FFF5                    RDSC7:      LDA         CLKDUM
141C   1F                                             RAR
141D   7C                                             MOV         A,H
141E   1F                                             RAR
141F   67                                             MOV         H,A
1420   15                                             DCR         D
1421   C2         1419                                JNZ         RDSC7
1424   F9                                             SPHL                  ; SP HOLDS HOUR AND DAY

1425   16         08                                  MVI         D,8
1427   2E         00                                  MVI         L,0
1429   3A         FFF5                    RDSC8:      LDA         CLKDUM
142C   1F                                             RAR
142D   7D                                             MOV         A,L
142E   1F                                             RAR
142F   6F                                             MOV         L,A
1430   15                                             DCR         D
1431   C2         1429                                JNZ         RDSC8     ; L IS THE DATE

1434   16         08                                  MVI         D,8
1436   26         00                                  MVI         H,0
1438   3A         FFF5                    RDSC9:      LDA         CLKDUM
143B   1F                                             RAR
143C   7C                                             MOV         A,H
143D   1F                                             RAR
143E   67                                             MOV         H,A
143F   15                                             DCR         D
1440   C2         1438                                JNZ         RDSC9     ; H IS THE MONTH

                                                      ; Since the upper 3 bits of month are not used, this will be the
                                                      ; counter for the loop
1443   16         00                                  MVI         D,0
1445   3A         FFF5                    RDSC10:     LDA         CLKDUM
1448   1F                                             RAR
1449   7A                                             MOV         A,D
144A   1F                                             RAR
144B   57                                             MOV         D,A
144C   7C                                             MOV         A,H
144D   C6         20                                  ADI         00100000B                ; INCREMENT THE UPPER 3 BIT COUNTER
144F   67                                             MOV         H,A                      ; SAVE COUNTER VALUE
1450   D2         1445                                JNC         RDSC10
                                                      ; UPPER 3 BITS OF H ARE ZERO AGAIN (A=H)
```

```
1453   7A                                             MOV         A,D                     ; D IS THE YEAR
1454   32         FFF2                                STA         AFREG+7
1457   22         FFF0                                SHLD        AFREG+5                 ; STORE MONTH AND DATE
145A   21         0000                                LXI         H,0
145D   39                                             DAD         SP
145E   22         FFEE                                SHLD        AFREG+3                 ; STORE DAY AND HOUR
1461   60                                             MOV         H,B
1462   69                                             MOV         L,C
1463   22         FFEC                                SHLD        AFREG+1                 ; STORE MIN AND SEC
1466   7B                                             MOV         A,E
1467   32         FFEB                                STA         AFREG                   ; STORE HUNDREDS

146A   2A         FFF3                                LHLD        SPREG                   ; GET SP
146D   F9                                             SPHL                                ; RESTORE SP
146E   D1                                             POP         D                       ; DE POINTS TO ARRAY TO PUT CLOCK DATA
146F   21         FFEB                                LXI         H,AFREG                 ; HL POINTS TO DATA THAT WAS JUST STORED
1472   06         08                                  MVI         B,8
1474   7E                                 RDSC11:     MOV         A,M                     ; READ IT
1475   12                                             STAX        D                       ; AND STORE IT
1476   23                                             INX         H
1477   13                                             INX         D
1478   05                                             DCR         B
1479   C2         1474                                JNZ         RDSC11
147C   F1                                             POP         PSW
147D   E6         08                                  ANI         1000B                   ; CHECK IE STATUS
147F   E1                                             POP         H
1480   D1                                             POP         D
1481   C1                                             POP         B
1482   CA         1488                                JZ       RDSC12                     ; JMP IF INTERRUPTS WERE NOT ENABLED
1485   F1                                             POP         PSW
1486   FB                                             EI
1487   C9                                             RET
1488   F1                                 RDSC12:     POP         PSW
1489   C9                                             RET



                                          ;
                                          ; Output E bytes of the string of bit patterns pointed to by HL, starting
                                          ; at display D and counting down.
                                          ;
148A   F5                                 LEDSTR:     push        psw
148B   E5                                             push        h
148C   E1                                 serv1A:     pop         h        ; get HL
148D   E5                                             push        h        ; put it back
148E   C5                                             push        b
148F   7A                                             mov         a,d      ; a= display number (rightmost = 0)
1490   C6         80                                  adi         rgtdsp   ; offset by rgtdsp
1492   47                                             mov         b,a
1493   4B                                             mov         c,e      ; e = # of bytes to display
1494   CD         00DA                                call        fun1     ; display string of bit patterns
1497   C1                                             pop         b
1498   E1                                             pop         h
1499   F1                                             pop         psw
149A   C9                                             ret

                                          ;
                                          ; Display the hex byte in E on the two displays on the right
                                          ;
149B   C5                                 serv1B:     push        b
149C   7B                                             mov         a,e
149D   CD         0109                                call        ddata
14A0   C1                                             pop         b
14A1   E1                                             pop         h
14A2   F1                                             pop         psw
14A3   C9                                             ret



                                          ;++++++++++++++++++++++++++++++++++++++++
                                          ;
                                          ; EPROM BURNING CODE
                                          ;
00E0=                         bnporta     equ         0e0h        ; EPROM board
00E1=                         bnportb     equ         0e1h
00E2=                         bnportc     equ         0e2h
00E3=                         bncntrl     equ         0e3h
00E4=                         bnportd     equ         0e4h


                                          ;
                                          ; Delays needed to program EPROM
                                          ;
14A4   F5                                 DLAYA:      PUSH        PSW      ; approx 5ms (6ms)
14A5   E5                                             PUSH        H
14A6   21         0281                                LXI         H,641    ; DELAY FOR 3.072 MHZ CLOCK
14A9   C3         14B9                                JMP         DLAY2

14AC   F5                                 DLAYB:      PUSH        PSW      ; approx 2.5ms (3ms)
14AD   E5                                             PUSH        H
14AE   21         0140                                LXI         H,320
14B1   C3         14B9                                JMP         DLAY2

14B4   F5                                 DLAYC:      PUSH        PSW      ; approx 1.25ms (1.5)
14B5   E5                                             PUSH        H
14B6   21         00A0                                LXI         H,160

                                          ;
                                          ; TIME delay               ; for 8085 is 24 t states
14B9   2B                                 DLAY2:      DCX         H        ; 6 T STATES
14BA   7C                                             MOV         A,H      ; 4 T STATES
14BB   B5                                             ORA         L        ; 4 T STATES
14BC   C2         14B9                                JNZ         DLAY2    ; 10 T STATES
14BF   E1                                             POP         H
14C0   F1                                             POP         PSW
14C1   C9                                             RET

                                          ;
                                          ; settling time for relays
                                          ;
14C2   CD         14A4                    rlystl:     call        dlaya
14C5   CD         14A4                                call        dlaya
14C8   CD         14A4                                call        dlaya
14CB   C9                                             ret

                                          ;
                                          ; TURN OFF EPROM PROGRAMMER.  Turn off main power, then the control relays.
                                          ;
14CC   F5                                 eoff:       push        PSW
14CD   3E         80                                  MVI         a,80h
14CF   D3         E3                                  out         bncntrl                 ; set port C as output
14D1   3E         FF                                  MVI         a,0ffh
14D3   D3         E2                                  out         bnportc                 ; write FF to input of EPROM in case of WR
14D5   7D                                             MOV         a,l                     ; type byte from TYPECHK
14D6   E6         7F                                  ani         01111111b               ; disable main power
14D8   D3         E4                                  out         bnportd
14DA   CD         14C2                                call        rlystl
14DD   AF                                             XRA         a
14DE   D3         E4                                  out         bnportd                 ; now shut off control relays
14E0   CD         14C2                                call        rlystl
14E3   CD         14E8                                call        bncnt                   ; make port C input again
14E6   F1                                             pop         PSW
14E7   C9                                             ret

14E8   3E         89                      bncnt:      MVI         a,89h
14EA   D3         E3                                  out         bncntrl
14EC   C9                                             ret


                                          ;
                                          ; Send DE to the address lines of EPROM programmer
                                          ;
14ED   F5                                 OUTAD:      PUSH        PSW
14EE   7B                                             MOV         A,E
14EF   D3         E0                                  OUT         BNPORTA
14F1   7A                                             MOV         A,D
14F2   D3         E1                                  OUT         BNPORTB
14F4   F1                                             POP         PSW
14F5   C9                                             RET
```

```



                                        ;
                                        ; RETURN A WITH BYTE FROM EPROM ADDRESS DE
                                        ; L MUST CONTAIN CONTROL BYTE FROM TYPECHECK
                                        ;
14F6   CD         14ED                  RDEPR:      CALL        OUTAD
14F9   2C                                           INR         L
14FA   7D                                           MOV         A,L
14FB   D3         E4                                OUT         BNPORTD
14FD   DB         E2                                IN          BNPORTC  ; READ EPROM BYTE
14FF   F5                                           PUSH        PSW
1500   2D                                           DCR         L
1501   7D                                           MOV         A,L
1502   D3         E4                                OUT         BNPORTD
1504   F1                                           POP         PSW
1505   C9                                           RET

                                        ;
                                        ; GARG:
                                        ; Put the arguments from the registers on the stack
                                        ; H=  page of RAM
                                        ; L=  page of EPROM
                                        ; DE= # of bytes
                                        ; B=  EPROM type
                                        ;
1506   78                               garg:       mov         a,b
1507   C1                                           pop         b        ; get ret address
1508   E5                                           push        h
1509   2E         00                                mvi         l,0      ; make RAM address
150B   E3                                           xthl                 ; put on stack and restore H
150C   65                                           mov         h,l      ; put in hi byte
150D   2E         00                                mvi         l,0      ; clear low byte
150F   E5                                           push        h        ; push EPROM address
1510   D5                                           push        d        ; push # of bytes
1511   26         00                                mvi         h,0
1513   6F                                           mov         l,a      ; HL = type
1514   E5                                           push        h        ; push type
1515   C5                                           push        b        ; save ret addr
1516   C9                                           ret



                                        ;
                                        ; The selected EPROM type is on top of stack. The routine POPs this and
                                        ; pushes second byte of the byte pairs in table E512 on stack
                                        ; cy = 0 if invalid type
1517   D1                               TYPECHK:    POP         D        ; DE= RET ADDRESS
1518   E1                                           POP         H        ; GET TYPE BYTE
1519   7D                                           MOV         A,L
151A   3D                                           DCR         A        ; CHANGE 1-6 TO 0-5
151B   FE         06                                CPI         6
151D   D2         1536                              JNC         argerr   ; IF A>=5 THEN THERE IS A TYPE ERROR
1520   21         153E                              LXI         H,E512
1523   07                                           RLC
1524   85                                           ADD         L
1525   6F                                           MOV         L,A
1526   D2         152A                              JNC         NOINC
1529   24                                           INR         H
152A   7E                               NOINC:      MOV       A,M
152B   D3         E4                                OUT         BNPORTD
152D   CD         14C2                              call        rlystl
1530   23                                           INX         H
1531   6E                                           MOV         L,M      ; GET BYTE FROM THE TABLE
1532   7D                                           MOV         A,L
1533   D3         E4                                OUT         BNPORTD
1535   37                                           STC                  ; SET CY = NO ERROR
1536   26         00                    argerr:     MVI         H,0
1538   E5                                           PUSH        H        ; TYPE IS NOW A BYTE FROM TABLE E512
1539   CD         14C2                              CALL        rlystl   ; settle the relays
153C   D5                                           PUSH        D        ; PUSH RET ADDRESS
153D   C9                                           RET
                                        ;
                                        ; EPROM BOARD CONTROL BYTES
153E   30         B0                    E512:       DEFB        30H,0B0H ;TYPE 1   27512 EPROM
1540   10         90                                DEFB        10H,090H ;TYPE 2   27256 EPROM
1542   80         80                                DEFB        80H,080H ;TYPE 3   27128 EPROM 12.5 VOLT
1544   40         C0                                DEFB        40H,0C0H ;TYPE 4   27128 EPROM 21 VOLT
1546   80         80                                DEFB        80H,080H ;TYPE 5   2764  EPROM 12.5 VOLT
1548   40         C0                                DEFB        40H,0C0H ;TYPE 6   2764  EPROM 21 VOLT

0011=                        numbrn:  equ            17          ; 50ms / 3ms (max ms)/(delay ms)

                                        ;
                                        ;SMOKE BURNS THE BYTE IN THE ACCUMULATOR
                                        ;
154A   F5                               SMOKE:      PUSH        PSW
154B   3E         80                                MVI         A,80H
154D   D3         E3                                OUT         BNCNTRL
154F   CD         14ED                              CALL        OUTAD
1552   F1                                           POP         PSW      ; GET A
1553   F5                                           PUSH        PSW      ; SAVE IT AGAIN
1554   D3         E2                                OUT         BNPORTC                  ; SEND A TO PORT C (DATA)
1556   2C                                           INR         L
1557   2C                                           INR         L        ; turn WE on
1558   7D                                           MOV         A,L
1559   D3         E4                                OUT         BNPORTD
155B   CD         14AC                              CALL        DLAYB    ; 2.5ms 8/20/91
155E   2D                                           DCR         L
155F   2D                                           DCR         L        ; turn WE off
1560   7D                                           MOV         A,L
1561   D3         E4                                OUT         BNPORTD
1563   3E         89                                MVI         A,89H
1565   D3         E3                                OUT         BNCNTRL  ; CONTROL PORT
1567   CD         14ED                              CALL        OUTAD
156A   F1                                           POP         PSW
156B   C9                                           RET

                                        ;
                                        ; Called by BRNIT.  Burns the byte in A and  returns z=false if okay
                                        ; and Z=true if burn error.
                                        ;
156C   D5                               BRN:        PUSH        D
156D   C5                                           PUSH        B
156E   0E         11                                MVI         C,NUMBRN ; see DLAY
1570   CD         154A                  BRN0:       CALL        SMOKE
1573   47                                           MOV         B,A      ; SAVE A
1574   2C                                           INR         L
1575   7D                                           MOV         A,L
1576   D3         E4                                OUT         BNPORTD  ; OUTPUT ENABLE
1578   DB         E2                                IN       BNPORTC     ; READ WHAT WAS WRITTEN AT BNPORTC (DATA)
157A   67                                           MOV         H,A
157B   2D                                           DCR         L
157C   7D                                           MOV         A,L
157D   D3         E4                                OUT         BNPORTD  ; DISABLE OUTPUT
157F   78                                           MOV         A,B      ; RESTORE BURN BYTE
1580   BC                                           CMP         H
1581   CA         158D                              JZ          BRN1     ; IF IT BURNED CORRECTLY, SKIP BELOW
1584   0D                                           DCR         C
1585   C2         1570                              JNZ         BRN0                     ; BURN IT AGAIN (BUT ONLY TOTAL OF 50mS)
1588   6F                                           MOV         L,A      ; L = RAM BYTE
1589   7C                                           MOV         A,H      ; A= EPROM BYTE
158A   C1                                           POP         B
158B   D1                                           POP         D        ; Z=TRUE = BURN PROBLEM
158C   C9                                           RET

158D   78                               BRN1:       MOV         A,B      ; RESTORE BURN BYTE
158E   CD         154A                              CALL        SMOKE    ; DO IT AGAIN TO BE SURE
1591   3E         01                                MVI         A,1
1593   B7                                           ORA         A        ; Z=0  = BURNT OKAY
1594   C1                                           POP         B
1595   D1                                           POP         D
1596   C9                                           RET
                                        ;
                                        ; MAIN EPROM BURNING SUBROUTINE
                                        ; Z=FALSE IF NO ERROR, TRUE OTHERWISE
                                        ;
1597   E1                               BRNIT:      POP         H        ; GET RET ADDRESS
1598   C1                                           POP         B        ; BC = TYPE BYTE
```

```
1599  79                           MOV       A,C      ; A = TYPE
159A  C1                           POP       B        ; # of bytes
159B  D1                           POP       D        ; DE= EPROM ADDRESS
159C  E3                           XTHL               ; HL = SOURCE ADDRESS, SAVE RET ADDRESS
159D  C5                           PUSH      B        ; # OF BYTES WILL BE  ON TOP OF STACK
159E  44                           MOV       B,H
159F  4D                           MOV       C,L      ; BC = SOURCE ADDRESS
15A0  6F                           MOV       L,A      ; L = TYPE BYTE
15A1  CD    14ED         BRNIT0:   CALL      OUTAD    ; OUTPUT DE AS THE EPROM ADDRESS
15A4  2C                           INR       L
15A5  7D                           MOV       A,L
15A6  D3    E4                     OUT       BNPORTD  ; turn OE on
15A8  DB    E2                     IN     BNPORTC     ;  read data from BNPORTC
15AA  67                           MOV       H,A      ; SAVE DATA FROM PORTC
15AB  2D                           DCR       L
15AC  7D                           MOV       A,L
15AD  D3    E4                     OUT       BNPORTD  ; turn OE off (BNPORTD)
15AF  0A                           LDAX      B        ; read byte from source
15B0  BC                           CMP       H
15B1  CA    15C2                   JZ        BRNIT1   ; if source=dest, don't burn it
15B4  CD    156C                   CALL      BRN
15B7  C2    15C2                   JNZ       BRNIT1   ; if no error, do next
15BA  CD    14CC                   CALL      EOFF
15BD  E1                           POP       H        ; REMOVE # OF BYTES
15BE  6F                           MOV       L,A      ; L = EPROM BYTE
15BF  0A                           LDAX      B        ;
15C0  67                           MOV       H,A      ; H = RAM BYTE

                                   ; BC = BAD RAM ADDRESS
                                   ; DE = BAD ROM ADDRESS
                                   ; H = RAM BYTE
                                   ; L = ROM BYTE

15C1  C9                           ret       ; Z=TRUE IF ERROR

15C2  03                 BRNIT1:   INX       B
15C3  13                           INX       D
15C4  E3                           XTHL               ; EXCHANGE TYPE WITH # OF BYTES LEFT
15C5  E5                           PUSH      H
15C6  C5                           PUSH      B
15C7  EB                           XCHG               ; HL = EPROM addr
15C8  CD    011D                   CALL      DADDR    ; display EPROM addr in left 4 displays
15CB  EB                           XCHG
15CC  C1                           POP       B
15CD  E1                           POP       H

15CE  2B                           DCX       H        ; dec # of bytes
15CF  7D                           MOV       A,L
15D0  B4                           ORA       h
15D1  E3                           XTHL               ; EXCHANGE # OF BYTES LEFT WITH TYPE
15D2  C2    15A1                   JNZ       BRNIT0               ; CONTINUE IF HL<>0
15D5  CD    14CC                   CALL      EOFF
15D8  F1                           POP       PSW      ; REMOVE THE NUMBER OF BYTES FROM STACK
15D9  3E    01                     MVI       A,1
15DB  B7                           ORA       A        ; Z= FALSE
15DC  C9                           RET                ; Z=FALSE IF NO ERROR

                         ;
                         ; UPON ENTRY  TOP OF STACK = TYPE FROM TYPECHK FOLLOWED BY ADDRESS
                         ; TO CHECK FOR ERASED EPROM.
                         ;
15DD  C1                 ERASE:    POP       B        ; GET RET ADDRESS
15DE  E1                           POP       H        ; GET TYPE
15DF  D1                           POP       D        ; GET EPROM ADDRESS
15E0  C5                           PUSH      B        ; SAVE RET ADDRESS
15E1  13                           INX       D
15E2  26    FF                     MVI       H,0FFH   ; ERASED EPROM BYTE = FF

15E4  1B                 ERASE0:   DCX       D
15E5  CD    14F6                   CALL      RDEPR    ; A = data at address DE in EPROM
15E8  BC                           CMP       H
15E9  C2    15F1                   Jnz       erase2   ; exit if not erased
15EC  7A                 ERASE1:   MOV       A,D
15ED  B3                           ORA       E
15EE  C2    15E4                   JNZ       ERASE0
                                   ;SUCCESS!  Z=1 IF ERASED
15F1  CD    14CC         erase2:   call      eoff     ; turn off power
15F4  C9                           ret


                         ; Stack must have on top: the type from TYPECHK, the number of bytes to check,
                         ; the EPROM address, then the RAM address.
15F5  E1                 VERIF:    POP       H        ; GET RET ADDRESS
15F6  C1                           POP       B        ; BC = TYPE BYTE
15F7  79                           MOV       A,C      ; A = TYPE
15F8  C1                           POP       B        ; # of bytes
15F9  D1                           POP       D        ; DE= EPROM ADDRESS
15FA  E3                           XTHL               ; HL = SOURCE ADDRESS, SAVE RET ADDRESS
15FB  C5                           PUSH      B        ; # OF BYTES WILL BE  ON TOP OF STACK
15FC  44                           MOV       B,H
15FD  4D                           MOV       C,L      ; BC = SOURCE ADDRESS
15FE  6F                           MOV       L,A      ; L = TYPE BYTE

15FF  CD    14F6         VERIF0:   CALL      RDEPR
1602  67                           MOV       H,A
1603  0A                           LDAX      B
1604  BC                           CMP       H
1605  CA    1611                   JZ        VERIF1   ; IF EQUAL, CONTINUE
1608  CD    14CC                   CALL      EOFF

                                   ; BC = BAD RAM ADDRESS              DE = BAD ROM ADDRESS
                                   ;  H = RAM BYTE             L = ROM BYTE
160B  6C                           MOV       L,H
160C  67                           MOV       H,A
160D  F1                           POP       PSW      ; REMOVE THE # OF BYTES FROM STACK
160E  AF                           XRA       A        ; SET z
160F  3C                           INR       A        ; CLEAR z
1610  C9                           RET                ; Z=0 = error

1611  13                 VERIF1:   INX       D
1612  03                           INX       B
1613  E3                           XTHL               ; EXCHANGE TYPE WITH # OF BYTES
1614  2B                           DCX       H
1615  7D                           MOV       A,L
1616  B4                           ORA       H
1617  E3                           XTHL               ; RESTORE THE TYPE AND SAVE # OF BYTES
1618  C2    15FF                   JNZ       VERIF0
161B  CD    14CC                   CALL      EOFF
161E  E1                           POP       H        ; REMOVE THE NUMBER OF BYTES FROM STACK
161F  C9                           RET                ; Z=1 = NO ERROR

                         ; Stack must have on top: the type from TYPECHK, the number of bytes to check,
                         ; the EPROM address, then the RAM address.
1620  E1                 READEPR: POP        H        ; get ret addr
1621  C1                           POP       B        ; get type
1622  79                           MOV       A,C      ; a=type
1623  C1                           POP       B        ; # of bytes
1624  D1                           POP       D        ; EPROM addr
1625  E3                           XTHL               ; save RET address and get RAM addr
1626  C5                           PUSH      B        ; # of bytes is on TOS
1627  44                           MOV       B,H
1628  4D                           MOV       C,L      ; bc = # RAM addr
1629  6F                           MOV       L,A      ; l = type
162A  CD    14F6         READE1:   CALL      RDEPR    ; read EPROM address DE
162D  02                           STAX      B        ; save it at RAM address BC
162E  03                           INX       B
162F  13                           INX       D
1630  E3                           XTHL               ; hl = # of bytes
1631  2B                           DCX       H
1632  7D                           MOV       A,L
1633  B4                           ORA       H
1634  E3                           XTHL               ; l = type
1635  C2    162A                   JNZ       READE1   ; loop till no more bytes
1638  CD    14CC                   CALL      EOFF
163B  E1                           POP       H        ; remove # of bytes
163C  C9                           RET


                         ; RDEPROM Copy DE bytes from EPROM to RAM
```

```
                                          ; H = page of RAM       L = page of EPROM
                                          ; DE = # of bytes       B = type of EPROM
                                          ;
163D   E1                                 SERV1C:     pop         H
163E   F1                                             pop         PSW

163F   F5                                 rdeprom:    PUSH        PSW
1640   C5                                             PUSH        B
1641   D5                                             PUSH        D
1642   E5                                             PUSH        H
1643   CD         1506                                CALL        GARG
1646   CD         14E8                                CALL        BNCNT
1649   CD         1517                                CALL        TYPECHK
164C   D2         1680                                JNC         TYPER1   ; JP IF INVALID TYPE
164F   CD         1620                                CALL        READEPR
1652   C2         16B2                                JNZ         BRNERR   ; SKIP IF ERRORS
1655   E1                                             POP         H
1656   D1                                             POP         D
1657   C1                                             POP         B
1658   F1                                             POP         PSW
1659   C9                                             RET


                                          ;
                                          ; VERIFY confirms that code is burnt
                                          ; H = page of RAM       L = page of EPROM
                                          ; DE = # of bytes       B = type of EPROM
                                          ;
165A   E1                                 SERV1D:     pop         H
165B   F1                                             pop         PSW
165C   F5                                 VERIFY:     PUSH        PSW
165D   C5                                             PUSH        B
165E   D5                                             PUSH        D
165F   E5                                             PUSH        H
1660   CD         1506                                CALL        GARG
1663   CD         14E8                                CALL        BNCNT
1666   CD         1517                                CALL        TYPECHK
1669   D2         1680                                JNC         TYPER1
166C   CD         15F5                                CALL        VERIF
166F   C2         16B2                                JNZ         BRNERR   ; SKIP IF ERRORS
1672   E1                                             POP         H
1673   D1                                             POP         D
1674   C1                                             POP         B
1675   F1                                             POP         PSW
1676   3E         00                                  MVI         A,0      ; indicates no errors
1678   C9                                             RET

                                          ; Give error # in A and show contradiction in source and dest. data
                                          ; BC = SOURCE ADDRESS   DE = EPROM ADDRESS
                                          ;  H = SOURCE BYTE         L  = EPROM BYTE
1679   F1                                 VERERR:      POP        PSW      ; DISCARD STUFF ON STACK
167A   F1                                             POP         PSW
167B   F1                                             POP         PSW
167C   F1                                             POP         PSW
167D   3E         02                                  MVI         A,2      ; ERROR
167F   C9                                             RET




1680   E1                                 TYPER1:     pop         H
1681   E1                                             pop         H
1682   E1                                 typer2:     pop         H        ; error exit point for ERASECHK
1683   E1                                             pop         H        ; discard data from garg
1684   E1                                             POP         H
1685   D1                                             POP         D
1686   C1                                             POP         B
1687   F1                                             POP         PSW
1688   3E         04                                  MVI         A,4      ; ERROR = INVALID TYPE
168A   C9                                             RET


                                          ; BURN burns the data in RAM to the EPROM
                                          ; H = page of RAM       L = page of EPROM
                                          ; DE = # of bytes       B = type of EPROM
                                          ;
168B                          SERV1E:
168B   E1                                             POP         H
168C   7C                                             MOV         A,H      ; ERROR IF HL < 2000
168D   FE         02                                  CPI         2
168F   D2         1694                                JNC         BRNSRV   ; IF ADDRESS >= 2000, CONTINUE
1692   F1                                             POP         PSW      ; DON'T COPY MOS
1693   C9                                             RET
1694   F1                                 BRNSRV:     POP         PSW

1695   F5                                 BURN:       PUSH        PSW
1696   C5                                             PUSH        B
1697   D5                                             PUSH        D
1698   E5                                             PUSH        H
1699   CD         1506                                CALL        GARG     ; GET ADDR1,ADDR2,NUM,TYPE

169C   CD         14E8                    BURNA:      CALL        BNCNT
169F   CD         1517                                CALL        TYPECHK
16A2   D2         1680                                JNC         TYPER1
16A5   CD         1597                                CALL        BRNIT
16A8   CA         16B2                                JZ          BRNERR   ; JUMP IF BURN ERROR
16AB   E1                                             POP         H
16AC   D1                                             POP         D
16AD   C1                                             POP         B
16AE   F1                                             POP         PSW
16AF   3E         00                                  MVI         A,0
16B1   C9                                             RET

                                          ; Give error # in A and show contradiction in source and dest. data
                                          ; BC = SOURCE ADDRESS   DE = EPROM ADDRESS
                                          ;  H = SOURCE BYTE         L  = EPROM BYTE
16B2   F1                                 BRNERR:     POP         PSW      ; DISCARD STUFF ON STACK
16B3   F1                                             POP         PSW
16B4   F1                                             POP         PSW
16B5   F1                                             POP         PSW
16B6   3E         01                                  MVI         A,1      ; ERROR
16B8   C9                                             RET


                                          ;
                                          ; ERASCHK
                                          ; Scans from address in DE, down to 0 of the EPROM type in B
                                          ; if a=0 then erased
                                          ; if a=1 then DE points to byte that isn't erased
                                          ;
16B9   E1                                 SERV1F:     pop         H
16BA   F1                                             pop         PSW
16BB   F5                                 ERASCHK:    PUSH        PSW
16BC   C5                                             PUSH        B
16BD   D5                                             PUSH        D
16BE   E5                                             PUSH        H
16BF   D5                                             PUSH        D        ; PUSH   # OF BYTES
16C0   48                                             MOV         C,B
16C1   06         00                                  MVI         B,0
16C3   C5                                             PUSH        B        ; PUSH TYPE
16C4   CD         14E8                                CALL        BNCNT
16C7   CD         1517                                CALL        TYPECHK
16CA   D2         1682                                JNC         TYPER2
16CD   CD         15DD                                CALL        ERASE
16D0   E1                                             POP         H
16D1   C2         16DA                                JNZ         ECHK1    ; if NZ then not erased
16D4   D1                                             POP         D
16D5   C1                                             POP         B
16D6   F1                                             POP         PSW
16D7   3E         00                                  MVI         A,0      ; ERASED...
16D9   C9                                             RET

16DA   C1                                 ECHK1:      POP         B        ; DISCARD OLD DE
16DB   C1                                             POP         B
16DC   F1                                             POP         PSW
16DD   3E         03                                  MVI         A,3      ; not erased
16DF   C9                                             RET
```

```
                                          ;
                                          ; ZAP: BURN INITIALIZATION CODE AND MOS SERVICES FOLLOWED BY AN APPLICATION
                                          ; PROGRAM TO A 32K EPROM.
                                          ;  H=upper byte of start address of application
                                          ;  DE=number of bytes in application
                                          ;
                                          ; Error code is returned in A
                                          ;  a=0 no error
                                          ;  a=1 burn error
                                          ;  a=2 verify error
                                          ;  a=3 not erased (DE = ADDRESS OF BYTE NOT ERASED)
                                          ; BC=address of source data, H=source byte
                                          ; DE=address of dest data, L=dest byte
                                          ;
                                          ; no registers preserved


16E0   E1                                 serv20:     pop         h
16E1   F1                                             pop         psw
16E2   F3                                 zap:        di
16E3   C5                                             push        b
16E4   E5                                             push        h
16E5   D5                                             push        d

16E6   11         7FFF                                lxi         d ,07fffh               ; point to top of EPROM
16E9   06         02                                  mvi         b,2                     ; 27256 EPROM
16EB   CD         16BB                                call        eraschk                 ; see if erased
16EE   B7                                             ora         a
16EF   CA         16F6                                jz          zeras                   ; jmp if erased
                                                      ; erase error
16F2   E1                                             pop         h                       ; discard old de
16F3   E1                                             pop         h
16F4   C1                                             pop         b
16F5   C9                                             ret

16F6   26         00                      zeras:      mvi         h,0                     ; high byte of source addr
16F8   2E         00                                  mvi         l,0                     ; high byte of dest addr
16FA   11         2F00                                lxi         d ,02f00h               ; number of bytes to burn
16FD   06         02                                  mvi         b,2                     ; 27256 EPROM
16FF   CD         1695                                call        burn                    ; burn init code and MOS services
1702   B7                                             ora         a
1703   C2         1727                                jnz         zap_err                 ; if not 0, error
1706   CD         165C                                call        verify                  ; see if data is there
1709   B7                                             ora         a
170A   C2         1727                                jnz         zap_err                 ; if not 0, error
170D   D1                                             pop         d                       ; get number of bytes
170E   E1                                             pop         h                       ; get hi bytes of start
170F   E5                                             push        h                       ; put them back
1710   D5                                             push        d
1711   06         02                                  mvi         b,2                     ; 27256 EPROM
1713   2E         2F                                  mvi         l,2fh                   ; burn at 2f00h

1715   CD         1695                                call        burn                    ; burn user's application
1718   B7                                             ora         a
1719   C2         1727                                jnz         zap_err                 ; if not 0, error
171C   CD         165C                                call        verify
171F   B7                                             ora         a
1720   C2         1727                                jnz         zap_err                 ; if not 0, error
1723   D1                                 zap_ext:    pop         d
1724   E1                                             pop         h
1725   C1                                             pop         b
1726   C9                                             ret

1727                          zap_err:
                                                      ; error occured in BURN or VERIFY, returning error data in
                                                      ; de,hl,bc so remove old values from stack
1727   33                                             INX         sp
1728   33                                             INX         sp       ; discard de
1729   33                                             INX         sp
172A   33                                             INX         sp       ; discard hl
172B   33                                             INX         sp
172C   33                                             INX         sp       ; discard bc
172D   C9                                             ret


                                          ;************************************************************
                                          ; D = bits indicate what dec pnts on
172E                          serv21:
172E   C5                                             push        b
172F   D5                                             push        d

1730   06         00                                  mvi         b,0
1732   05                                             dcr         b
1733   7A                                             mov         a,d
1734   F5                                             push        psw
1735                          dplup:
1735   04                                             inr         b
1736   78                                             mov         a,b
1737   FE         06                                  cpi         6
1739   CA         175F                                jz          serv21a

173C   3E         60                                  mvi         a,rdrgtdsp              ;comand to read dsp
173E   80                                             add         b                       ;COMMAND TO READ DIGIT
173F   D3         41                                  out         dspcmd

1741   DB         40                                  in          dspout                  ;GET SEGMENT VALUES
1743   57                                             mov         d,a                     ;SAVE A REG

1744   3E         80                                  mvi         a,rgtdsp                ;COMMAND TO WRITE DIGIT
1746   80                                             add         b
1747   D3         41                                  out         dspcmd

1749   F1                                             pop         psw
174A   1F                                             rar
174B   F5                                             push        psw
174C   DA         1757                                jc          dpon
174F                          dpoff:
174F   7A                                             mov         a,d
1750   E6         F7                                  ani         11110111b               ;TURN OFF DECIMAL POINT
1752   D3         40                                  out         dspout                  ;WRITE A TO DIGIT
1754   C3         1735                                jmp         dplup
1757                          dpon:
1757   7A                                             mov         a,d
1758   F6         08                                  ori         00001000b
175A   D3         40                                  out         dspout
175C   C3         1735                                jmp         dplup
175F                          serv21a:
175F   F1                                             pop         psw
1760   D1                                             pop         d
1761   C1                                             pop         b
1762   E1                                             pop         h
1763   F1                                             pop         psw
1764   C9                                             ret

                                          ;***********************************************************
                                          ;
                                          ;BEFORE: DE = BIN #, AFTER: DE = BCD #

1765   F5                                 bin2bcd:    push        psw
1766   E5                                             push        h
1767                          serv22:
1767   C5                                             push        b
1768   0E         04                                  mvi         c,4      ; max number of digits to display
176A   21         0000                                lxi         h,0      ; init to 0

176D   E5                                 bcdlup:     push        h        ; this will hold the BCD value
176E   EB                                             xchg                 ; hl = de
176F   11         000A                                lxi         d,10
1772   CD         114A                                call        div16    ; divide hex value by 10 dec
1775   7B                                             mov         a,e      ; e= lowest decimal digit
1776   EB                                             xchg                 ; save hl in de
1777   E1                                             pop         h        ; get bcd value
1778   CD         0093                                call        digit4   ; rotate left 1 digit and insert Accumulator
                                                                           ; in the low nibble
177B   0D                                             dcr         c
177C   C2         176D                                jnz         bcdlup   ; loop until hl has 4 BCD digits
                                                      ; the number is inverted, change it
177F   7D                                             mov         a,l
1780   0F                                             rrc
```

```
1781   0F                                             rrc
1782   0F                                             rrc
1783   0F                                             rrc         ; swap nibbles
1784   6C                                             mov         l,h      ; save h in l
1785   67                                             mov         h,a      ; put new l in h
1786   7D                                             mov         a,l
1787   0F                                             rrc
1788   0F                                             rrc
1789   0F                                             rrc
178A   0F                                             rrc         ; swap the nibbles
178B   6F                                             mov         l,a
178C   EB                                             xchg
178D   C1                                             pop         b
178E   E1                                             pop         h
178F   F1                                             pop         psw
1790   C9                                             ret
                                          ;----------------------------------------
                                          ; PRE: BCD # IN DE  POST: BIN # IN DE
                                          ;
1791   F5                                 bcd2bin:    push        psw
1792   E5                                             push        h

1793   C5                                 serv23:     push        b
1794   D5                                             push        d        ;save D
1795   7B                                             mov         a,e
1796   E6         0F                                  ani         0Fh
1798   4F                                             mov         c,a      ;store 1st nibble in C
1799   7B                                             mov         a,e
179A   E6         F0                                  ani         0f0h
179C   0F                                             rrc
179D   0F                                             rrc
179E   0F                                             rrc
179F   0F                                             rrc
17A0   6F                                             mov         l,a      ;store 2nd nibble in l
17A1   26         00                                  mvi         h,0      ;clear h
17A3   CD         17D5                                call        times10  ;mult 2nd nibble by 10
17A6   06         00                                  mvi         b,0
17A8   09                                             dad         b        ;add 1st nibble to 2nd

17A9   7A                                             mov         a,d
17AA   E6         0F                                  ani         0fh      ;A = 3rd nibble
17AC   EB                                             xchg                 ;save sum in DE
17AD   6F                                             mov         l,a      ;L = 3rd nibble
17AE   26         00                                  mvi         h,0
17B0   0E         02                                  mvi         c,2
17B2   CD         17D5                    tyms100:    call        times10  ;mult 3rd nible by 100
17B5   0D                                             dcr         c
17B6   C2         17B2                                jnz         tyms100
17B9   19                                             dad         d        ;add result to sum

17BA   D1                                             pop         d        ;restore original bcd num
17BB   7A                                             mov         a,d
17BC   E6         F0                                  ani         0f0h
17BE   0F                                             rrc
17BF   0F                                             rrc
17C0   0F                                             rrc
17C1   0F                                             rrc                  ;A = 4th nibble
17C2   EB                                             xchg                 ;save sum in DE
17C3   6F                                             mov         l,a
17C4   26         00                                  mvi         h,0
17C6   0E         03                                  mvi         c,3
17C8   CD         17D5                    tym1000:    call        times10  ;mult 4th nibble by 1000
17CB   0D                                             dcr         c
17CC   C2         17C8                                jnz         tym1000
17CF   19                                             dad         d        ;add result to sum
17D0   EB                                             xchg                 ;DE = answer
17D1   C1                                             pop         b
17D2   E1                                             pop         h
17D3   F1                                             pop         psw
17D4   C9                                             ret
                                          ;----------------------------------------------------
17D5                          times10:
17D5   D5                                             push        d
17D6   54                                             mov         d,h
17D7   5D                                             mov         e,l
17D8   19                                             dad         d
17D9   19                                             dad         d
17DA   19                                             dad         d
17DB   19                                             dad         d
17DC   19                                             dad         d
17DD   19                                             dad         d
17DE   19                                             dad         d
17DF   19                                             dad         d
17E0   19                                             dad         d
17E1   D1                                             pop         d
17E2   C9                                             ret
                                          ;*****************************************************
                                          ; IN: D = 0 IF LEDS OFF, 1 IF LEDS ON
                                          ; OUT: 4KEYS IN REG PAIR DE
17E3                          serv24:
17E3   C5                                             push        b
17E4   D5                                             push        d
17E5   21         0000                                lxi         h,0
17E8   AF                                             xra         a
17E9   BA                                             cmp         d
17EA   CA         17F0                                jz       readlup
17ED   CD         011D                                call        daddr
17F0                          readlup:
17F0   CD         0133                                call        rdkey
17F3   FE         17                                  cpi         17h
17F5   CA         180E                                jz       rdexit
17F8   FE         13                                  cpi         13h
17FA   D2         17F0                                jnc         readlup
17FD   CD         0093                                call        digit4
1800   D1                                             pop         d
1801   D5                                             push        d
1802   7A                                             mov         a,d
1803   FE         00                                  cpi         0
1805   CA         17F0                                jz       readlup
1808   CD         011D                                call        daddr
180B   C3         17F0                                jmp         readlup
180E                          rdexit:
180E   D1                                             pop         d
180F   EB                                             xchg
1810   C1                                             pop         b
1811   E1                                             pop         h
1812   F1                                             pop         psw
1813   C9                                             ret

                                          ;*******************************************************
                                          ; EPROM PROGRAMER
                                          ;-------------------------------------------------------
                                          ;-------------------------------------------------------
                                          ;CONSTANTS
                                          ;-------------------------------------------------------
0040=                         OFFSET          EQU     40H
00C0=                         OFST64          EQU     0C0H
                                          ;-------------------------------------------------------
1814                          EPRMPRO:
1814   3E         02                                  MVI         A,2      ;DEFAULT EPROM TYPE
1816   32         FFFE                                STA         ETYP     ;DEFAULT ETYP 2
1819   CD         0109                                CALL        DDATA

181C                          DSPLYMAIN:
181C   CD         1D6B                                CALL        CLRSCR
181F   11         1D7E                                LXI         D,MES_MAIN              ;PRINT MAIN MENU
1822   CD         10CD                                CALL        PSTRNG
1825   3A         FFFE                                LDA         ETYP
1828   FE         00                                  CPI         0
182A   CA         186A                                JZ          MESS0
182D   FE         01                                  CPI         1
182F   CA         1864                                JZ          MESS1
1832   FE         02                                  CPI         2
1834   CA         185E                                JZ          MESS2
1837   FE         03                                  CPI         3
1839   CA         1858                                JZ          MESS3
183C   FE         04                                  CPI         4
183E   CA         1852                                JZ          MESS4
1841   FE         05                                  CPI         5
```

```
1843   CA         184C                                  JZ          MESS5
1846                          MESS6:
1846   21         2002                                  LXI         H,TXT_6TYPE
1849   C3         186D                                  JMP         DSPLRST
184C                          MESS5:
184C   21         1FDF                                  LXI         H,TXT_5TYPE
184F   C3         186D                                  JMP         DSPLRST
1852                          MESS4:
1852   21         1FBC                                  LXI         H,TXT_4TYPE
1855   C3         186D                                  JMP         DSPLRST
1858                          MESS3:
1858   21         1F99                                  LXI         H,TXT_3TYPE
185B   C3         186D                                  JMP         DSPLRST
185E                          MESS2:
185E   21         1F76                                  LXI         H,TXT_2TYPE
1861   C3         186D                                  JMP         DSPLRST
1864                          MESS1:
1864   21         1F52                                  LXI         H,TXT_1TYPE
1867   C3         186D                                  JMP         DSPLRST
186A                          MESS0:
186A   21         1F2E                                  LXI         H,TXT_0TYPE
186D                          DSPLRST:
186D   16         1A                                    MVI         D,26
186F                          TYPLUP:
186F   5E                                               MOV         E,M
1870   23                                               INX         H
1871   CD         10BE                                  CALL        CONOUT
1874   15                                               DCR         D
1875   C2         186F                                  JNZ         TYPLUP

1878   11         1EBC                                  LXI         D,MES_RSTMAN
187B   CD         10CD                                  CALL        PSTRNG
187E                          WAIT:
187E   1E         04                                    MVI         E,4
1880   16         05                                    MVI         D,5
1882   21         2178                                  LXI         H,LED_B
1885   CD         148A                                  CALL        LEDSTR

1888   CD         1CB1                                  CALL        CRLF                    ;CARRAGE RETURN,LINE FEED
188B   CD         1CB1                                  CALL        CRLF
188E   CD         1CBE                                  CALL        PROMPT                  ;PLACE '>'
1891                          NOPRMT:
1891   CD         33F6                                  CALL        GETCHAR                 ;WAIT FOR KEYPRESS FROM TERMINAL
1894   FE         1B                                    CPI         1BH                     ;IF ESCAPE JMP WAIT
1896   CA         187E                                  JZ          WAIT

1899   5F                                               MOV         E,A                     ;E = CHAR
189A   CD         10BE                                  CALL        CONOUT                  ;ECHO CHARACTER TO TERMINAL
189D   FE         3F                                    CPI         '?'                     ;IF ? THEN REDRAW MAIN MENU
189F   CA         181C                                  JZ          DSPLYMAIN
18A2   E6         DF                                    ANI         11011111B               ;REMOVE CASE SENSITIVTY
18A4   FE         54                                    CPI         'T'
18A6   CA         18E3                                  JZ          S_LCTTYPE               ;IF S THEN JMP
18A9   FE         52                                    CPI         'R'
18AB   CA         1960                                  JZ          R_EAD
18AE   FE         41                                    CPI         'A'
18B0   CA         197C                                  JZ          A_UTO
18B3   FE         4D                                    CPI         'M'
18B5   CA         19A2                                  JZ          M_ODMEM
18B8   FE         44                                    CPI         'D'
18BA   CA         19D6                                  JZ          D_UMP
18BD   FE         42                                    CPI         'B'
18BF   CA         1A17                                  JZ          B_URN
18C2   FE         56                                    CPI         'V'
18C4   CA         1A43                                  JZ          V_ERIFY
18C7   FE         5A                                    CPI         'Z'
18C9   CA         1A6B                                  JZ          Z_AP
18CC   FE         45                                    CPI         'E'
18CE   CA         1AA3                                  JZ          E_RSECHK
18D1   FE         4C                                    CPI         'L'
18D3   CA         1AF6                                  JZ          L_OADMEM
18D6   FE         53                                    CPI         'S'
18D8   CA         1B51                                  JZ          U_PLOAD
18DB   FE         43                                    CPI         'C'
18DD   CA         1B3B                                  JZ          C_LRMEM
18E0   C3         187E                                  JMP         WAIT                    ;IF NONE OF THE ABOVE, WAIT FOR NEW CHAR
                                          ;------------------------------------------------------------
                                          ; DISPLAYS "ETYPS" AND WAITS FOR USER TO SELECT TYPE

18E3                          S_LCTTYPE:
18E3   CD         1D6B                                  CALL        CLRSCR
18E6   11         1F25                                  LXI         D,MES_LSTYPE            ;PRINT SELECT EPROM MESS
18E9   CD         10CD                                  CALL        PSTRNG
18EC   11         1EF9                                  LXI         D,MES_SLCT              ;PRINT 'ENTER SELECTION'
18EF   CD         10CD                                  CALL        PSTRNG
18F2   CD         1CBE                                  CALL        PROMPT                  ;PLACE PROMPT
18F5   CD         33F6                                  CALL        GETCHAR                 ;WAIT FOR CHAR FROM TRMINL
18F8   5F                                               MOV         E,A
18F9   CD         10BE                                  CALL        CONOUT                  ;ECHO CHARACTER
18FC   FE         30                                    CPI         '0'
18FE   CA         1922                                  JZ          MTYPE0                  ;JMP IF 0
1901   FE         31                                    CPI         '1'
1903   CA         192A                                  JZ          MTYPE1                  ;JMP IF 1
1906   FE         32                                    CPI         '2'
1908   CA         1932                                  JZ          MTYPE2
190B   FE         33                                    CPI         '3'
190D   CA         193A                                  JZ          MTYPE3
1910   FE         34                                    CPI         '4'
1912   CA         1942                                  JZ          MTYPE4
1915   FE         35                                    CPI         '5'
1917   CA         194A                                  JZ          MTYPE5
191A   FE         36                                    CPI         '6'
191C   CA         1952                                  JZ          MTYPE6
191F   C3         18E3                                  JMP         S_LCTTYPE
1922                          MTYPE0:
1922   3E         00                                    MVI         A,0
1924   32         FFFE                                  STA         ETYP     ;STORE 0 IN TYPE
1927   C3         1957                                  JMP         MOVE
192A                          MTYPE1:
192A   3E         01                                    MVI         A,1
192C   32         FFFE                                  STA         ETYP     ;STORE 1 IN TYPE
192F   C3         1957                                  JMP         MOVE
1932                          MTYPE2:
1932   3E         02                                    MVI         A,2
1934   32         FFFE                                  STA         ETYP
1937   C3         1957                                  JMP         MOVE
193A                          MTYPE3:
193A   3E         03                                    MVI         A,3
193C   32         FFFE                                  STA         ETYP
193F   C3         1957                                  JMP         MOVE
1942                          MTYPE4:
1942   3E         04                                    MVI         A,4
1944   32         FFFE                                  STA         ETYP
1947   C3         1957                                  JMP         MOVE
194A                          MTYPE5:
194A   3E         05                                    MVI         A,5
194C   32         FFFE                                  STA         ETYP
194F   C3         1957                                  JMP         MOVE
1952                          MTYPE6:
1952   3E         06                                    MVI         A,6
1954   32         FFFE                                  STA         ETYP
1957                          MOVE:
1957   3A         FFFE                                  LDA         ETYP
195A   CD         0109                                  CALL        DDATA
195D   C3         181C                                  JMP         DSPLYMAIN               ;GOTO MAIN

                                          ;----------------------------------------------------
                                          ; GETS START ADD AND # BYTES AND LOADS MEM WITH EPRM DATA

1960                          R_EAD:
1960   CD         1C8B                                  CALL        S1MENU   ;GET ADD AND #B
1963   CD         1969                                  CALL        R__EAD   ;CALL READ
1966   C3         187E                                  JMP         WAIT     ;GOTO MAIN

1969                          R__EAD:
1969   78                                               MOV         A,B      ;ABORT IF ESC CHAR
196A   FE         00                                    CPI         0
196C   C2         197B                                  JNZ         EXITRD
196F   CD         1D43                                  CALL        LDSTK    ;PUT DATA ON STACK
```

```
1972  CD       14E8                          CALL      BNCNT
1975  CD       1517                          CALL      TYPECHK
                                             ;JNC      ERRTYP
1978  CD       1620                          CALL      READEPR
197B                     EXITRD:
197B  C9                                     RET

                                   ;-------------------------------------------
                                   ;CALLS  ERASE CHK, BURN, THEN VIFY
197C                     A_UTO:
197C  CD       1C8B                          CALL      S1MENU
197F  78                                     MOV       A,B
1980  FE       00                            CPI       0        ;IF ESCAPE
1982  C2       187E                          JNZ       WAIT     ; THEN ABORT
1985  E5                                     PUSH      H
1986  D5                                     PUSH      D
1987  C5                                     PUSH      B
1988  CD       1AA9                          CALL      E__RSECHK
198B  C1                                     POP       B
198C  D1                                     POP       D
198D  E1                                     POP       H
198E  FE       00                            CPI       0        ;IF ESCAPE
1990  C2       187E                          JNZ       WAIT     ; THEN ABORT
1993  E5                                     PUSH      H
1994  D5                                     PUSH      D
1995  C5                                     PUSH      B
1996  CD       1A20                          CALL      B__URN
1999  C1                                     POP       B
199A  D1                                     POP       D
199B  E1                                     POP       H
199C  CD       1A4C                          CALL      V__ERIFY
199F  C3       187E                          JMP       WAIT     ;GOTO MAIN MENU


                                   ;-------------------------------------------------------
19A2                     M_ODMEM:
19A2  CD       1C50                          CALL      S2MENU               ;GET START ADD
19A5  78                                     MOV       A,B
19A6  FE       00                            CPI       0                    ;ABORT IF ESC
19A8  C2       19D3                          JNZ       XITMOD
19AB  CD       1CB1                          CALL      CRLF                 ;CARRIAGE RETURN
19AE                     MODAGAN:
19AE  CD       1C15                          CALL      DSPLYADD             ;DISPLAY ADD
19B1  CD       1C37                          CALL      DSPLDATA             ;DISPLAY DTA
19B4  D5                                     PUSH      D
19B5  1E       1D                            MVI       E,29                 ;LEFT ARROW
19B7  CD       10BE                          CALL      CONOUT               ;SENT TO TRMINAL
19BA  1E       1D                            MVI       E,29                 ;LEFT ARROW
19BC  CD       10BE                          CALL      CONOUT               ;SND TO TRMINAL
19BF  D1                                     POP       D
19C0  CD       1CC6                          CALL      ASC_TRM2H            ;GET NEW DATA
19C3  78                                     MOV       A,B
19C4  FE       03                            CPI       3                    ;IF "ENTER" THEN NO MODIFY
19C6  CA       19CF                          JZ        NOMOD
19C9  FE       00                            CPI       0                    ;IF ESC THEN EXIT
19CB  C2       19D3                          JNZ       XITMOD
19CE  71                                     MOV       M,C                  ;MOV NEW VAL TO MEM
19CF                     NOMOD:
19CF  23                                     INX       H                    ;INCREMENT POINTER
19D0  C3       19AE                          JMP       MODAGAN              ;LOOP
19D3                     XITMOD:
19D3  C3       187E                          JMP       WAIT

                                   ;---------------------------------------------
19D6                     D_UMP:
19D6  CD       1C8B                          CALL      S1MENU   ;GET ADDRESS AND # BYTES
19D9  78                                     MOV       A,B
19DA  FE       00                            CPI       0
19DC  C2       1A14                          JNZ       EXITDMP  ;EXIT IF ESC
19DF  CD       1CB1                          CALL      CRLF
19E2                     NEWSCR:
19E2  0E       10                            MVI       C,16     ;16 LINES AT A TIME
19E4                     NEWROW:
19E4  06       10                            MVI       B,16     ;B = NUM BYTES PER LINE
19E6  CD       1C15                          CALL      DSPLYADD
19E9                     LUPER:
19E9  CD       1C37                          CALL      DSPLDATA
19EC  23                                     INX       H        ;INCREMENT POINTER
19ED  1B                                     DCX       D        ;DECREMENT BYTE COUNTER
19EE  AF                                     XRA       A        ;A = 0
19EF  BB                                     CMP       E        ;IF E NOT 0
19F0  C2       19F7                          JNZ       DMPAGAIN ; THEN DUMP ANOTHER BYTE
19F3  BA                                     CMP       D        ;IF E AND D 0
                                                                ; THEN EXIT
19F4  CA       1A14                          JZ        EXITDMP
19F7                     DMPAGAIN:
19F7  05                                     DCR       B        ;DECREMENT B
19F8  C2       19E9                          JNZ       LUPER    ;IF NOT 0 THEN DUMP AGAIN
19FB  0D                                     DCR       C
19FC  C2       19E4                          JNZ       NEWROW
19FF  D5                                     PUSH      D
1A00  E5                                     PUSH      H
1A01  CD       1CB1                          CALL      CRLF     ;CAIRAGE RETURN
1A04  11       201F                          LXI       D,MES_MORE
1A07  CD       10CD                          CALL      PSTRNG   ;DISPLAY MORE MESS
1A0A  CD       33F6                          CALL      GETCHAR
1A0D  FE       1B                            CPI       1BH      ;IF NOT ESC THEN DO AGAIN
1A0F  E1                                     POP       H        ; ELSE EXIT
1A10  D1                                     POP       D
1A11  C2       19E2                          JNZ       NEWSCR

1A14                     EXITDMP:
1A14  C3       187E                          JMP       WAIT
                                   ;---------------------------------------------------
1A17                     B_URN:
1A17  CD       1C8B                          CALL      S1MENU   ;GET STRT ADD AND # BYTES
1A1A  CD       1A20                          CALL      B__URN
1A1D  C3       187E                          JMP       WAIT     ;GOTO MAIN
1A20                     B__URN:
1A20  78                                     MOV       A,B      ;IF A NON HEXVAL
1A21  FE       00                            CPI       0        ;THEN RET
1A23  C2       1A3B                          JNZ       EXITBRN
1A26  CD       1D43                          CALL      LDSTK
1A29  CD       14E8                          CALL      BNCNT
1A2C  CD       1517                          CALL      TYPECHK
1A2F  D2       1A42                          JNC       ERRTYP
1A32  CD       1597                          CALL      BRNIT
1A35  CA       1A3C                          JZ        ERRBRN   ;IF NO ERR THEN DISPLAY NO ERR MES
1A38  CD       1BF7                          CALL      EPMGOOD
1A3B                     EXITBRN:
1A3B  C9                                     RET
1A3C                     ERRBRN:
1A3C  CD       1C06                          CALL      EPMERR
1A3F  C3       1A3B                          JMP       EXITBRN
1A42                     ERRTYP:
1A42  FF                                     RST       7
                                   ;-------------------------------------------------------------
1A43                     V_ERIFY:
1A43  CD       1C8B                          CALL      S1MENU   ;GET ADD & #BYTES
1A46  CD       1A4C                          CALL      V__ERIFY ;CALL VERIFY
1A49  C3       187E                          JMP       WAIT     ;GOTO WAIT
1A4C                     V__ERIFY:
1A4C  78                                     MOV       A,B
1A4D  FE       00                            CPI       0
1A4F  C2       1A64                          JNZ       EXITVRFY
1A52  CD       1D43                          CALL      LDSTK
1A55  CD       14E8                          CALL      BNCNT
1A58  CD       1517                          CALL      TYPECHK
1A5B  CD       15F5                          CALL      VERIF
1A5E  C2       1A65                          JNZ       ERRVER
1A61  CD       1BF7                          CALL      EPMGOOD
1A64                     EXITVRFY:
1A64  C9                                     RET
1A65                     ERRVER:
1A65  CD       1C06                          CALL      EPMERR
1A68  C3       1A64                          JMP       EXITVRFY
                                   ;-------------------------------------------------------------
1A6B                     Z_AP:
1A6B  3A       FFFE                          LDA       ETYP
```

```
1A6E   FE         02                                    CPI        2
1A70   CA         1A7F                                  JZ         DOZAP
1A73   CD         1CB1                                  CALL       CRLF
1A76   11         2156                                  LXI        D,MES_MUSTTYP2
1A79   CD         10CD                                  CALL       PSTRNG
1A7C   C3         187E                                  JMP        WAIT
1A7F                          DOZAP:
1A7F   CD         1C8B                                  CALL       S1MENU
1A82   78                                               MOV        A,B
1A83   FE         00                                    CPI        0
1A85   C2         187E                                  JNZ        WAIT
1A88   CD         16E2                                  CALL       ZAP
1A8B   FE         03                                    CPI        3
1A8D   CC         1AE8                                  CZ         NOTERASE
1A90   FE         02                                    CPI        2
1A92   CC         1C06                                  CZ         EPMERR
1A95   FE         00                                    CPI        0
1A97   CC         1BF7                                  CZ         EPMGOOD
1A9A   C3         187E                                  JMP        WAIT
1A9D                          ZAPBAD:
1A9D   CD         1C06                                  CALL       EPMERR
1AA0   C3         187E                                  JMP        WAIT
                                          ;--------------------------------------------------------------
1AA3                          E_RSECHK:
1AA3   CD         1AA9                                  CALL       E__RSECHK
1AA6   C3         187E                                  JMP        WAIT

1AA9                          E__RSECHK:
1AA9   3A         FFFE                                  LDA        ETYP
1AAC   FE         05                                    CPI        5
1AAE   D2         1ACD                                  JNC        SZ8K
1AB1   FE         03                                    CPI        3
1AB3   D2         1AC7                                  JNC        SZ16K
1AB6   FE         02                                    CPI        2
1AB8   D2         1AC1                                  JNC        SZ32K
1ABB                          SZ64K:
1ABB   11         FFFF                                  LXI        D,0FFFFH
1ABE   C3         1AD0                                  JMP        BLNKCHK
1AC1                          SZ32K:
1AC1   11         7FFF                                  LXI        D,07FFFH
1AC4   C3         1AD0                                  JMP        BLNKCHK
1AC7                          SZ16K:
1AC7   11         3FFF                                  LXI        D,03FFFH
1ACA   C3         1AD0                                  JMP        BLNKCHK
1ACD                          SZ8K:
1ACD   11         1FFF                                  LXI        D,01FFFH
1AD0                          BLNKCHK:
1AD0   3A         FFFE                                  LDA        ETYP
1AD3   FE         00                                    CPI        0
1AD5   C2         1AD9                                  JNZ        SKPOS4
1AD8   3C                                               INR        A
1AD9                          SKPOS4:
1AD9   47                                               MOV        B,A
1ADA   CD         16BB                                  CALL       ERASCHK
1ADD   FE         00                                    CPI        0
1ADF   C2         1AE8                                  JNZ        NOTERASE
1AE2                          ISERASED:
1AE2   11         2089                                  LXI        D,MES_BLNK
1AE5   C3         1AEB                                  JMP        ERSMESS
1AE8                          NOTERASE:
1AE8   11         206F                                  LXI        D,MES_NBLNK
1AEB                          ERSMESS:
1AEB   F5                                               PUSH       PSW
1AEC   E5                                               PUSH       H
1AED   CD         1CB1                                  CALL       CRLF
1AF0   CD         10CD                                  CALL       PSTRNG
1AF3   E1                                               POP        H
1AF4   F1                                               POP        PSW
1AF5   C9                                               RET

                                          ;--------------------------------------------------------------
1AF6                          L_OADMEM:
1AF6   CD         1CB1                                  CALL       CRLF
1AF9   11         211C                                  LXI        D,MES_RDYUPLD
1AFC   CD         10CD                                  CALL       PSTRNG
1AFF   CD         1CB1                                  CALL       CRLF
1B02   1E         00                                    MVI        E,0
                                                        ;
1B04   3A         FFFE                                  LDA        ETYP
1B07   FE         00                                    CPI        0
1B09   16         C0                                    MVI        D,OFST64
1B0B   CA         1B10                                  JZ         SKPOS2
1B0E   16         40                                    MVI        D,OFFSET
1B10                          SKPOS2:
1B10   CD         332D                                  CALL       HEX1CON
1B13   11         209F                                  LXI        D,MES_UPLDCMPLT
1B16   1F                                               RAR
1B17   1F                                               RAR
1B18   D2         1B21                                  JNC        CHKNOHEX
1B1B                          CHKSUMER:
1B1B   11         2104                                  LXI        D,MES_CHKSUM
1B1E   C3         1B32                                  JMP        XITL_MEM
1B21                          CHKNOHEX:
1B21   1F                                               RAR
1B22   D2         1B2B                                  JNC        ESCERR
1B25   11         20B8                                  LXI        D,MES_NONHEX
1B28   C3         1B32                                  JMP        XITL_MEM
1B2B                          ESCERR:
1B2B   1F                                               RAR
1B2C   D2         1B32                                  JNC        XITL_MEM
1B2F   11         20DF                                  LXI        D,MES_ESCERR
1B32                          XITL_MEM:
1B32   CD         1CB1                                  CALL       CRLF
1B35   CD         10CD                                  CALL       PSTRNG
1B38   C3         187E                                  JMP        WAIT
                                          ;*********************************************************
                                          ; WRITES FF TO ALL MEM LOCATIONS FROM 4000 TO BFFF
                                          ;---------------------------------------------------------
1B3B                          C_LRMEM:
1B3B   21         4000                                  LXI        H,4000H
1B3E   06         FF                                    MVI        B,0FFH
1B40                          CLRAGN:
1B40   70                                               MOV        M,B
1B41   23                                               INX        H
1B42   7D                                               MOV        A,L
1B43   FE         00                                    CPI        0
1B45   C2         1B40                                  JNZ        CLRAGN
1B48   7C                                               MOV        A,H
1B49   FE         C0                                    CPI        0C0H     ;CLEAR TO C000
1B4B   C2         1B40                                  JNZ        CLRAGN
1B4E   C3         187E                                  JMP        WAIT

                                          ;------------------------------------------------------------
1B51                          U_PLOAD:
1B51   CD         1C8B                                  CALL       S1MENU
1B54   78                                               MOV        A,B
1B55   FE         00                                    CPI        0
1B57   C2         187E                                  JNZ        WAIT     ;EXIT IF ESC
1B5A   D5                                               PUSH       D
1B5B   E5                                               PUSH       H
1B5C   CD         1CB1                                  CALL       CRLF
1B5F   11         2136                                  LXI        D,MES_PRSENTR
1B62   CD         10CD                                  CALL       PSTRNG
1B65   CD         33F6                                  CALL       GETCHAR
1B68   FE         1B                                    CPI        1BH
1B6A   E1                                               POP        H
1B6B   D1                                               POP        D
1B6C   CA         187E                                  JZ         WAIT
1B6F   CD         1B75                                  CALL       INTELUP
1B72   C3         1891                                  JMP        NOPRMT
                                          ;***************************************************************************
                                          ;
                                          ; THIS ROUTINE TAKES DATA THAT STARTS AT HL AND # BYTES AT DE AND SENDS IT TO
                                          ; THE OUTPUT PORT AS AN INTEL HEXFILE.
                                          ;
                                          ;------------------------------------------------------------------
1B75                          INTELUP:
1B75   E5                                               PUSH       H        ;SAVE H WHICH POINTS TO SYS RAM
1B76   13                                               INX        D
```

```
1B77     3A         FFFE                                LDA         ETYP
1B7A     FE         00                                  CPI         0
1B7C     06         40                                  MVI         B,OFFSET
1B7E     C2         1B83                                JNZ         NOTOS
1B81     06         C0                                  MVI         B,OFST64
1B83                            NOTOS:
1B83     7C                                             MOV         A,H
1B84     90                                             SUB         B
1B85     67                                             MOV         H,A      ;NEW H HAS OFFSET SUBTRACTED

1B86     06         10                      HXLUP:   MVI            B,10H    ;DATA BYTES PER LINE OF HEX

1B88     7A                                             MOV         A,D      ;MOVE D TO A TO CHECK FOR 0
1B89     FE         00                                  CPI         0        ;SEE IF 0
1B8B     C2         1B9A                                JNZ         DOHEXL   ;IF NOT 0 THEN DE > 10H

1B8E     7B                                             MOV         A,E
1B8F     FE         11                                  CPI         11H      ;SEE IF A FULL LINE OF HEX
1B91     D2         1B9A                                JNC         DOHEXL   ;OUTPUT A HEX LINE

                                                        ;THE DATA BYTES ARE NOW LESS THAN 10H
1B94     43                                             MOV         B,E      ;BYTES = E. THIS IS THE LAST DATA FIELD
1B95     AF                                             XRA         A        ;CLEAR A
1B96     B8                                             CMP         B
1B97     CA         1BA0                                JZ       FINHEX      ;IF B=0 THEN EXIT

1B9A     CD         1BAA                    DOHEXL:  CALL           HEXLINE  ;OUTPUT A LINE OF HEX
1B9D     C3         1B86                                JMP         HXLUP    ;DO NEXT LINE OF HEX

1BA0                            FINHEX:
1BA0                            LASTHX:
1BA0     21         0000                                LXI         H,0      ;ADDRESS = 0000
1BA3     06         00                                  MVI         B,0      ;NO MORE DATA, SO B=0
1BA5     CD         1BAA                                CALL        HEXLINE  ;DO THE LAST LINE OF HEX

1BA8     E1                                             POP         H
1BA9     C9                                             RET
                                            ;***** END OF MAIN LOOP *****

                                            ;******** HEXLINE STARTS AT HL AND SENDS OUT THE # OF BYTES (HELD IN B)******
1BAA     3E         3A                      HEXLINE: MVI            A,':'
1BAC     CD         1BE5                                CALL        PUTCHAR  ;OUTPUT THE COLON
1BAF     AF                                             XRA         A        ;CLEAR A. A HOLDS THE RUNNING SUM


                                                        ;OUTPUT # OF BYTES
1BB0     48                                             MOV         C,B
1BB1     81                                             ADD         C        ;ADD BYTE TO ACCUM
1BB2     CD         1BEC                                CALL        BIN2ASC  ;OUTPUT # OF BYTES AS ASCII

                                                        ;OUTPUT THE ADDRESS
1BB5     4C                                             MOV         C,H
1BB6     81                                             ADD         C        ;ADD TO ACCUM
1BB7     CD         1BEC                                CALL        BIN2ASC  ;OUTPUT ADDRESS HI AS ASCII

1BBA     4D                                             MOV         C,L
1BBB     81                                             ADD         C        ;ADD TO ACCUM
1BBC     CD         1BEC                                CALL        BIN2ASC  ;OUTPUT ADDRESS LO AS ASCII

                                                        ;OUTPUT TYPE
1BBF     0E         00                                  MVI         C,0      ;TYPE 0
1BC1     81                                             ADD         C        ;ADD TO ACCUM
1BC2     CD         1BEC                                CALL        BIN2ASC  ;OUTPUT TYPE AS ASCII

                                                        ;OUTPUT DATA (IF THERE IS SOME)
1BC5     04                                             INR         B        ;INC IN CASE OF 1 BYTE,(B=1)
1BC6     05                                 LINLUP:  DCR            B        ;DECREMENT BYTE COUNTER
1BC7     CA         1BDB                                JZ       CHCKSUM     ;JUMP IF B=0
                                                        ;**
1BCA     33                                             INX         SP
1BCB     33                                             INX         SP
1BCC     E3                                             XTHL
1BCD     4E                                             MOV         C,M      ;LOAD A BYTE FROM THE DATA STRING
1BCE     81                                             ADD         C        ;ADD TO ACCUM
1BCF     23                                             INX         H        ;POINT TO NEXT BYTE IN MEMORY
1BD0     E3                                             XTHL
1BD1     3B                                             DCX         SP
1BD2     3B                                             DCX         SP
                                                        ;**
1BD3     CD         1BEC                                CALL        BIN2ASC  ;OUTPUT DATA AS ASCII

1BD6     23                                             INX         H        ;POINT TO THE NEXT BYTE
1BD7     1B                                             DCX         D        ;1 LESS BYTE TO OUTPUT
1BD8     C3         1BC6                                JMP         LINLUP

                                                        ;OUTPUT CHECKSUM
1BDB     2F                                 CHCKSUM: CMA
1BDC     3C                                             INR         A        ;CHECKSUM = 2'S COMPLEMENT OF SUM
1BDD     4F                                             MOV         C,A
1BDE     CD         1BEC                                CALL        BIN2ASC  ;OUTPUT THE CHECKSUM
1BE1     CD         1CB1                                CALL        CRLF

1BE4     C9                                             RET
                                            ;************ END OF HEXLINE ***********
                                            ;PATCH FOR INTELUP
1BE5                            PUTCHAR:
1BE5     D5                                             PUSH        D
1BE6     5F                                             MOV         E,A
1BE7     CD         10BE                                CALL        CONOUT
1BEA     D1                                             POP         D
1BEB     C9                                             RET
                                            ;********************************************
                                            ;PATCH FOR INTELUP
1BEC                            BIN2ASC:
1BEC     F5                                             PUSH        PSW
1BED     D5                                             PUSH        D
1BEE     C5                                             PUSH        B
1BEF     79                                             MOV         A,C
1BF0     CD         11E4                                CALL        HEX2
1BF3     C1                                             POP         B
1BF4     D1                                             POP         D
1BF5     F1                                             POP         PSW
1BF6     C9                                             RET

                                            ;******************************************************
                                            ; DISPLAYS EPROM GOOD MESSAGE
                                            ;------------------------------------------------------
1BF7                            EPMGOOD:
1BF7     E5                                             PUSH        H
1BF8     CD         1CB1                                CALL        CRLF
1BFB     CD         1CB1                                CALL        CRLF
1BFE     11         203F                                LXI         D,MES_EGOOD
1C01     CD         10CD                                CALL        PSTRNG
1C04     E1                                             POP         H
1C05     C9                                             RET
                                            ;******************************************************
                                            ; DISPLAYS EPROM ERROR MESSAGE
                                            ;------------------------------------------------------
1C06                            EPMERR:
1C06     E5                                             PUSH        H
1C07     CD         1CB1                                CALL        CRLF
1C0A     CD         1CB1                                CALL        CRLF
1C0D     11         205A                                LXI         D,MES_EERR
1C10     CD         10CD                                CALL        PSTRNG
1C13     E1                                             POP         H
1C14     C9                                             RET
                                            ;*****************************************************************
                                            ; PRE:  ADDRESS IN HL
                                            ; POST: NONE
                                            ;
                                            ; DISPLAYS TO CONSOLE THE ADDRESS IN HL - OFFSET, PLACES ':' AFTER
                                            ;-----------------------------------------------------------------
1C15                            DSPLYADD:
1C15     D5                                             PUSH        D        ;SAVE D
1C16     CD         1CB1                                CALL        CRLF
1C19     3A         FFFE                                LDA         ETYP
1C1C     FE         00                                  CPI         0
1C1E     7C                                             MOV         A,H      ;DISPLAY H
1C1F     CA         1C27                                JZ          SKPOS1
1C22     D6         40                                  SUI         OFFSET
```

```
1C24   C3         1C29                                JMP         SKOS64
1C27                          SKPOS1:
1C27   D6         C0                                  SUI         OFST64
1C29                          SKOS64:
1C29   CD         11E4                                CALL        HEX2
1C2C   7D                                             MOV         A,L      ;DISPLAY L
1C2D   CD         11E4                                CALL        HEX2
1C30   1E         3A                                  MVI         E,':'    ;PLACE COLLEN
1C32   CD         10BE                                CALL        CONOUT   ;RESTORE D
1C35   D1                                             POP         D
1C36   C9                                             RET
                                          ;***********************************************************
                                          ; PRE:  HL = ADDRESS OF DATA
                                          ; POST: NONE
                                          ;
                                          ; PLACES ' ' THEN DATA AT HL
                                          ;-----------------------------------------------------------
1C37                          DSPLDATA:
1C37   D5                                             PUSH        D
1C38   1E         20                                  MVI         E,' '
1C3A   CD         10BE                                CALL        CONOUT
1C3D   7E                                             MOV         A,M
1C3E   CD         11E4                                CALL        HEX2
1C41   D1                                             POP         D
1C42   C9                                             RET
                                          ;***************************************************
                                          ; PRE:  HL = START ADD OF SORCE
                                          ;       DE = START ADD OF DEST
                                          ;       B  = # OF BYTES
                                          ; POST: NONE
                                          ;---------------------------------------------------
1C43                          MEMMOVE:
1C43   F5                                             PUSH        PSW
1C44   C5                                             PUSH        B
1C45                          MOVAGAIN:
1C45   7E                                             MOV         A,M      ;MOV DATA AT ADDRESS HL TO ACC
1C46   12                                             STAX        D        ;SAVE DATA IN ACC AT ADRESS DE
1C47   13                                             INX         D        ;INC DESTINATION ADDRESS
1C48   23                                             INX         H        ;INC SORCE ADDRESS
1C49   05                                             DCR         B        ;DEC BYTE COUNTER
1C4A   C2         1C45                                JNZ         MOVAGAIN ;IF 0
1C4D   C1                                             POP         B        ; THEN EXIT
1C4E   F1                                             POP         PSW
1C4F   C9                                             RET
                                          ;***************************************************
                                          ; PRE:  NONE
                                          ; POST: ADDRESS IN HL
                                          ;
                                          ; DISPLAYS "STARTING ADDRESS" ,GETS 4 HEX VALUES FROM TERMINAL IN HL
                                          ;---------------------------------------------------
1C50                          S2MENU:
1C50   D5                                             PUSH        D
1C51   CD         1CB1                                CALL        CRLF
1C54   CD         1CB1                                CALL        CRLF
1C57   11         1EFF                                LXI         D,MES_STRTAD
1C5A   CD         10CD                                CALL        PSTRNG
1C5D   CD         1CBE                                CALL        PROMPT
1C60   CD         1CC6                                CALL        ASC_TRM2H
1C63   78                                             MOV         A,B
1C64   FE         00                                  CPI         0
1C66   C2         1C89                                JNZ         EXITM2
1C69   61                                             MOV         H,C
1C6A   CD         1CC6                                CALL        ASC_TRM2H
1C6D   78                                             MOV         A,B
1C6E   FE         00                                  CPI         0
1C70   C2         1C89                                JNZ         EXITM2
1C73   69                                             MOV         L,C
1C74   F5                                             PUSH        PSW
1C75   3A         FFFE                                LDA         ETYP
1C78   FE         00                                  CPI         0
1C7A   CA         1C84                                JZ          SK64EPM
1C7D   7C                                             MOV         A,H
1C7E   C6         40                                  ADI         OFFSET   ;ADD OFFSET
1C80   67                                             MOV         H,A
1C81   C3         1C88                                JMP         SKPOS
1C84                          SK64EPM:
1C84   7C                                             MOV         A,H
1C85   C6         C0                                  ADI         OFST64
1C87   67                                             MOV         H,A
1C88                          SKPOS:
1C88   F1                                             POP         PSW
1C89                          EXITM2:
1C89   D1                                             POP         D
1C8A   C9                                             RET
                                          ;*****************************************************
                                          ; PRE:  NONE
                                          ; POST: HL = ADRESS
                                          ;       DE = # OF BYTES
                                          ;----------------------------------------------------
1C8B                          S1MENU:
1C8B   CD         1C50                                CALL        S2MENU
1C8E   C2         1CB0                                JNZ         EXITM1
1C91   E5                                             PUSH        H
1C92   11         1F12                                LXI         D,MES_BYTES
1C95   CD         10CD                                CALL        PSTRNG
1C98   E1                                             POP         H
1C99   CD         1CBE                                CALL        PROMPT
1C9C   CD         1CC6                                CALL        ASC_TRM2H
1C9F   78                                             MOV         A,B
1CA0   FE         00                                  CPI         0
1CA2   C2         1CB0                                JNZ         EXITM1
1CA5   51                                             MOV         D,C
1CA6   CD         1CC6                                CALL        ASC_TRM2H
1CA9   78                                             MOV         A,B
1CAA   FE         00                                  CPI         0
1CAC   C2         1CB0                                JNZ         EXITM1
1CAF   59                                             MOV         E,C
1CB0                          EXITM1:
1CB0   C9                                             RET
                                          ;******************************************************
                                          ;
                                          ;------------------------------------------------------
1CB1                          CRLF:
1CB1   D5                                             PUSH        D
1CB2   1E         0D                                  MVI         E,0DH
1CB4   CD         10BE                                CALL        CONOUT
1CB7   1E         0A                                  MVI         E,0AH
1CB9   CD         10BE                                CALL        CONOUT
1CBC   D1                                             POP         D
1CBD   C9                                             RET
                                          ;******************************************************
                                          ;
                                          ;------------------------------------------------------
1CBE                          PROMPT:
1CBE   D5                                             PUSH        D
1CBF   1E         3E                                  MVI         E,'>'
1CC1   CD         10BE                                CALL        CONOUT
1CC4   D1                                             POP         D
1CC5   C9                                             RET

                                          ;******************************************************
                                          ; PRE:  NONE
                                          ; POST: C = 2 HEX VALUES FROM TERMINAL
                                          ;       B = 1 IF ESCAPE CHAR, 2 IF NON HEX, 3 IF RETURN, 0 OTHERWISE
                                          ;---------------------------------------------------------
1CC6                          ASC_TRM2H:
1CC6   F5                                             PUSH        PSW
1CC7   D5                                             PUSH        D
1CC8   CD         33F6                                CALL        GETCHAR
1CCB   FE         1B                                  CPI         1BH
1CCD   CA         1CF7                                JZ          ERRATH
1CD0   FE         0D                                  CPI         0DH
1CD2   CA         1CF2                                JZ          RETATH
1CD5   5F                                             MOV         E,A
1CD6   47                                             MOV         B,A
1CD7   CD         10BE                                CALL        CONOUT
1CDA   CD         33F6                                CALL        GETCHAR
1CDD   FE         1B                                  CPI         1BH
1CDF   CA         1CF7                                JZ          ERRATH
```

```
1CE2   FE         0D                                  CPI         0DH
1CE4   CA         1CF2                                JZ          RETATH
1CE7   5F                                             MOV         E,A
1CE8   4F                                             MOV         C,A
1CE9   CD         10BE                                CALL        CONOUT
1CEC   CD         1CFC                                CALL        ASCII2HEX
1CEF   C3         1CF9                                JMP         XITATH
1CF2                          RETATH:
1CF2   06         03                                  MVI         B,3
1CF4   C3         1CF9                                JMP         XITATH
1CF7                          ERRATH:
1CF7   06         01                                  MVI         B,1
1CF9                          XITATH:
1CF9   D1                                             POP         D
1CFA   F1                                             POP         PSW
1CFB   C9                                             RET

                                          ;******************************************************
                                          ;PRE:  BC CONTAIN HIGH AND LO ASCII VALUES
                                          ;POST: C CONTAINS HEX VALUE.  B = 1 IF ESCAPE, 2 IF NON HEX, 0 OTHERWISE
                                          ;------------------------------------------------------
1CFC                          ASCII2HEX:
1CFC   F5                                             PUSH        PSW
1CFD   D5                                             PUSH        D
1CFE   16         02                                  MVI         D,2      ;D = COUNTER
1D00   1E         00                                  MVI         E,0      ;E = INITIAL RESULT
1D02   78                                             MOV         A,B      ;B = IS FIRST NUMBER
1D03                          NXTCHR:
1D03   FE         1B                                  CPI         1BH
1D05   CA         1D38                                JZ          ESCPE
1D08   FE         30                                  CPI         '0'
1D0A   DA         1D33                                JC          ERROR
1D0D   FE         3A                                  CPI         ':'
1D0F   D2         1D17                                JNC         LETTER
1D12   E6         0F                                  ANI         0FH
1D14   C3         1D25                                JMP         SKIP1
1D17                          LETTER:
1D17   E6         DF                                  ANI         0DFH     ;MAKE UPPER CASE
1D19   FE         41                                  CPI         'A'
1D1B   DA         1D33                                JC          ERROR
1D1E   FE         47                                  CPI         'G'
1D20   D2         1D33                                JNC         ERROR
1D23   D6         37                                  SUI         37H
1D25                          SKIP1:
1D25   B3                                             ORA         E
1D26   15                                             DCR         D
1D27   CA         1D3D                                JZ          DONE
1D2A   07                                             RLC
1D2B   07                                             RLC
1D2C   07                                             RLC
1D2D   07                                             RLC
1D2E   5F                                             MOV         E,A
1D2F   79                                             MOV         A,C
1D30   C3         1D03                                JMP         NXTCHR
1D33                          ERROR:
1D33   06         02                                  MVI         B,2
1D35   C3         1D40                                JMP         ERRDNE
1D38                          ESCPE:
1D38   06         01                                  MVI         B,1
1D3A   C3         1D40                                JMP         ERRDNE
1D3D                          DONE:
1D3D   06         00                                  MVI         B,0
1D3F   4F                                             MOV         C,A
1D40                          ERRDNE:
1D40   D1                                             POP         D
1D41   F1                                             POP         PSW
1D42   C9                                             RET
                                          ;***************************************************
                                          ; CALL  HEX2 = 11DC
                                          ; PRE:  HEX VALUE TO DISPLAY IN A
                                          ; POST: NONE
                                          ;---------------------------------------------------
                                          ;***************************************************
                                          ; PRE:  ADDRES OF EPROM IN HL
                                          ;       # BYTES IN DE
                                          ;       TYPE OF EPROM IN A
                                          ;
                                          ;
                                          ; POST: STACK IS LOADED: TYPE    TOS
                                          ;                        # BYTES
                                          ;                        EPROM ADDRES
                                          ;                        RAM ADDRES    =EPROM ADDRESS + C000H
                                          ;---------------------------------------------------
1D43                          LDSTK:
1D43   44                                             MOV         B,H
1D44   4D                                             MOV         C,L
1D45   E1                                             POP         H
1D46   C5                                             PUSH        B
1D47   3A         FFFE                                LDA         ETYP
1D4A   FE         00                                  CPI         0
1D4C   CA         1D56                                JZ          ADOS64
1D4F   78                                             MOV         A,B
1D50   D6         40                                  SUI         OFFSET
1D52   47                                             MOV         B,A
1D53   C3         1D5A                                JMP         OSDUN
1D56                          ADOS64:
1D56   78                                             MOV         A,B
1D57   D6         C0                                  SUI         OFST64
1D59   47                                             MOV         B,A
1D5A                          OSDUN:
1D5A   3A         FFFE                                LDA         ETYP
1D5D   FE         00                                  CPI         0
1D5F   C2         1D63                                JNZ         SKPTF
1D62   3C                                             INR         A
1D63                          SKPTF:
1D63   C5                                             PUSH        B
1D64   D5                                             PUSH        D
1D65   4F                                             MOV         C,A
1D66   06         00                                  MVI         B,0
1D68   C5                                             PUSH        B
1D69   E5                                             PUSH        H
1D6A   C9                                             RET

                                          ;***************************************************
                                          ; PRE:  NONE
                                          ; POST: NONE
                                          ;---------------------------------------------------
1D6B                          CLRSCR:
1D6B   D5                                             PUSH        D
1D6C   1E         0D                                  MVI         E,0DH
1D6E   CD         10BE                                CALL        CONOUT
1D71   16         18                                  MVI         D,24
1D73                          CLRCNT:
1D73   1E         0A                                  MVI         E,0AH
1D75   CD         10BE                                CALL        CONOUT
1D78   15                                             DCR         D
1D79   C2         1D73                                JNZ         CLRCNT
1D7C   D1                                             POP         D
1D7D   C9                                             RET

                                          ;---------------------------------------------------

1D7E                          MES_MAIN:
1D7E   20         20 20 20 20 20                      DB          '       PRIMER EPROM PROGRAMMER'
1D9C   0D                                             DB          0DH
1D9D   0A                                             DB          0AH
1D9E   20         20 20 20 20 20                      DB          '               EMAC INC.'
1DB6   0D                                             DB          0DH
1DB7   0A                                             DB          0AH
1DB8   0D                                             DB          0DH
1DB9   0A                                             DB          0AH
1DBA   0D                                             DB          0DH
1DBB   0A                                             DB          0AH

1DBC   20         20 20 20 20 20                      DB          '       A  Auto (erase chk, burn, verify)'
1DE4   0D                                             DB          0DH
1DE5   0A                                             DB          0AH

1DE6   20         20 20 20 20 20                      DB          '       B  Burn EPROM'
```

```
1DFA   0D                                             DB          0DH
1DFB   0A                                             DB          0AH

1DFC   20         20 20 20 20 20                      DB          '       C  Clear memory'
1E12   0D                                             DB          0DH
1E13   0A                                             DB          0AH

1E14   20         20 20 20 20 20                      DB          '       D  Dump memory'
1E29   0D                                             DB          0DH
1E2A   0A                                             DB          0AH

1E2B   20         20 20 20 20 20                      DB          '       E  Erase check'
1E40   0D                                             DB          0DH
1E41   0A                                             DB          0AH

1E42   20         20 20 20 20 20                      DB          '       L  Load memory'
1E57   0D                                             DB          0DH
1E58   0A                                             DB          0AH

1E59   20         20 20 20 20 20                      DB          '       M  Modify memory'
1E70   0D                                             DB          0DH
1E71   0A                                             DB          0AH

1E72   20         20 20 20 20 20                      DB          '       R  Read EPROM'
1E86   0D                                             DB          0DH
1E87   0A                                             DB          0AH

1E88   20         20 20 20 20 20                      DB          '       S  Send file'
1E9B   0D                                             DB          0DH
1E9C   0A                                             DB          0AH

1E9D   20         20 20 20 20 20                      DB          '       T  Type select  < TYPE '
1EBB   24                                             DB          '$'

1EBC                          MES_RSTMAN:
1EBC   0D                                             DB          0DH
1EBD   0A                                             DB          0AH

1EBE   20         20 20 20 20 20                      DB          '       V  Verify EPROM'
1ED4   0D                                             DB          0DH
1ED5   0A                                             DB          0AH

1ED6   20         20 20 20 20 20                      DB          '       Z  Zap EPROM'
1EE9   0D                                             DB          0DH
1EEA   0A                                             DB          0AH

1EEB   20         20 20 20 20 20                      DB          '       ?  Help'


1EF9                          MES_SLCT:
1EF9   0D                                             DB          0DH
1EFA   0A                                             DB          0AH
1EFB   0A                                             DB          0AH
1EFC   0A                                             DB          0AH
1EFD   0A                                             DB          0AH
1EFE   24                                             DB          '$'

1EFF                          MES_STRTAD:
1EFF   20         73 74 61 72 74                      DB          ' starting address '
1F11   24                                             DB          '$'
1F12                          MES_BYTES:
1F12   20         20 6E 75 6D 62                      DB          '  number of bytes '
1F24   24                                             DB          '$'

                                          ;-------------------------------------------------------------
1F25                          MES_LSTYPE:
1F25   0D                                             DB          0DH
1F26   0A                                             DB          0AH
1F27   20         20 20 20 20 20                      DB          '       '
1F2E                          TXT_0TYPE:
1F2E   30         20 28 36 34 4B                      DB          '0 (64K X 8 HI, Vpp = 12.5)>'
1F49   0D                                             DB          0DH
1F4A   0A                                             DB          0AH
1F4B   20         20 20 20 20 20                      DB          '       '
1F52                          TXT_1TYPE:
1F52   31         20 28 36 34 4B                      DB          '1 (64K X 8 LO, Vpp = 12.5)>'
1F6D   0D                                             DB          0DH
1F6E   0A                                             DB          0AH
1F6F   20         20 20 20 20 20                      DB          '       '
1F76                          TXT_2TYPE:
1F76   32         20 28 33 32 4B                      DB          '2 (32K X 8, Vpp = 12.5)>  '
1F90   0D                                             DB          0DH
1F91   0A                                             DB          0AH
1F92   20         20 20 20 20 20                      DB          '       '
1F99                          TXT_3TYPE:
1F99   33         20 28 31 36 4B                      DB          '3 (16K X 8, Vpp = 12.5)>  '
1FB3   0D                                             DB          0DH
1FB4   0A                                             DB          0AH
1FB5   20         20 20 20 20 20                      DB          '       '
1FBC                          TXT_4TYPE:
1FBC   34         20 28 31 36 4B                      DB          '4 (16K X 8, Vpp = 21.0)>  '
1FD6   0D                                             DB          0DH
1FD7   0A                                             DB          0AH
1FD8   20         20 20 20 20 20                      DB          '       '
1FDF                          TXT_5TYPE:
1FDF   35         20 28 20 38 4B                      DB          '5 ( 8K X 8, Vpp = 12.5)>  '
1FF9   0D                                             DB          0DH
1FFA   0A                                             DB          0AH
1FFB   20         20 20 20 20 20                      DB          '       '
2002                          TXT_6TYPE:
2002   36         20 28 20 38 4B                      DB          '6 ( 8K X 8, Vpp = 21.0)>  '
201C   0D                                             DB          0DH
201D   0A                                             DB          0AH
201E   24                                             DB          '$'
                                          ;---------------------------------------------------------------
201F                          MES_MORE:
201F   20         45 53 43 20 74                      DB          ' ESC to exit, RET to continue. '
203E   24                                             DB          '$'
                                          ;---------------------------------------------------------------
203F                          MES_EGOOD:
203F   20         20 20 20 20 20                      DB          '       NO ERRORS DETECTED '
2059   24                                             DB          '$'
205A                          MES_EERR:
205A   20         20 20 20 20 20                      DB          '       EPROM ERROR. '
206E   24                                             DB          '$'

                                          ;----------------------------------------------------------------
206F                          MES_NBLNK:
206F   20         20 20 20 20 20                      DB          '       DEVICE NOT ERASED '
2088   24                                             DB          '$'
2089                          MES_BLNK:
2089   20         20 20 20 20 20                      DB          '       DEVICE ERASED '
209E   24                                             DB          '$'
209F                          MES_UPLDCMPLT:
209F   20         20 20 20 20 20                      DB          '       UPLOAD COMPLETE. '
20B7   24                                             DB          '$'
20B8                          MES_NONHEX:
20B8   20         20 20 20 20 20                      DB          '       NON HEX CHARACTER INCOUNTERED. '
20DE   24                                             DB          '$'
20DF                          MES_ESCERR:
20DF   20         20 20 20 20 20                      DB          '       ESCAPE CHARACTER ENOUNTERED. '
2103   24                                             DB          '$'
2104                          MES_CHKSUM:
2104   20         20 20 20 20 20                      DB          '       CHECKSUM ERROR. '
211B   24                                             DB          '$'
211C                          MES_RDYUPLD:
211C   20         20 20 20 20 20                      DB          '       READY FOR UPLOAD. '
2135   24                                             DB          '$'

2136                          MES_PRSENTR:
2136   20         20 20 20 20 20                      DB          '       PRESS A KEY WHEN READY. '
2155   24                                             DB          '$'

2156                          MES_MUSTTYP2:
2156   20         20 20 20 20 20                      DB          '       MUST SELECT TYPE 2 EPROM. '
2177   24                                             DB          '$'

2178   C7                                 LED_B:  DB  0C7H        ;"B"
2179   C1                                             DB          0C1H     ;"U"
```

```
217A   05                                             DB          005H     ;"R"
217B   45                                             DB          045H     ;"N"







                                          ;++++++++++++++++++++++++++++
                                          ; END OF STUFF COPIED BY ZAP
                                          ;++++++++++++++++++++++++++++
                                                      ORG         2F01H
2F01   C3         3260                    moscode: jmp            init_mon




                                          ;========================================================================
                                          ; SELF TEST CODE
                                          ;========================================================================

3FFE=                         chksum      equ         3ffeh       ; address for checksum
2F04   55         41 52 54 20 74          commsg:  defb           'UART test'
2F0D   0A         0D 3E 24    prmpt:   defb           10,13,'>$'

2F11                          SLFTST:
                                                      ; set decimal pts on displays 1 to 4
2F11   06         06                                  mvi         b,6      ; start at left display
2F13   78                                 slft1:      mov         a,b      ; b= 80h-85h
2F14   F6         80                                  ori         80h
2F16   3D                                             dcr         a        ; make 1 less
2F17   D3         41                                  out         dspcmd   ; select display
2F19   3E         08                                  mvi         a,8      ; make a period
2F1B   D3         40                                  out         dspout   ; output bit pattern to the display
2F1D   05                                             dcr         b
2F1E   C2         2F13                                jnz         slft1

                                                      ;
                                                      ; Checksum the EPROM
                                                      ;
2F21   2A         3FFE                                lhld        chksum   ; read check sum from last 2 bytes
2F24   CD         30B6                                call        checksm  ; do a check sum from 1ffe down to 0
2F27   7D                                             mov         a,l
2F28   B4                                             ora         h
2F29   CA         2F37                                jz          etst2    ; if chksum=0 then skip error msg
2F2C   3E         0B                                  mvi         a,11
2F2E   CD         00CC                                call        regprnt  ; display bE for bad EPROM
2F31   CD         016C                                call        beep
2F34   CD         0133                                call        rdkey    ; wait here so displays aren't trashed
2F37                          etst2:

                                                      ;
                                                      ; Check the RAM
                                                      ;

2F37   3E         08                                  mvi         a,8
2F39   CD         00CC                                call        regprnt  ; print "rd" on the right displays

                                                      ; This checks out the 32k RAM which may be in slot 1
                                                      ; 1st check for pos. "A" mem map.  If no ram at 8000 or 4000
                                                      ; then check 8155 RAM at FF00
2F3C   21         4000                                lxi         h,4000h
2F3F   01         8000                                lxi         b,8000h  ; bc is the number of bytes to check
2F42   7E                                 rmchk2:     mov         a,m
2F43   57                                             mov         d,a      ; preserve original A
2F44   3C                                             inr         a        ; inc A
2F45   34                                             inr         m        ; inc mem
2F46   BE                                             cmp         m        ; They should be same if RAM
2F47   72                                             mov         m,d      ; restore original data, in case of RAM
2F48   CA         2F5B                                jz       rmchk1      ; jmp if RAM at this address
2F4B   7C                                             mov         a,h
2F4C   FE         80                                  cpi         80h
2F4E   CA         2F56                                jz       rmchk3      ; if 8000 has been checked, check ff00
2F51   26         80                                  mvi         h,80h
2F53   C3         2F42                                jmp         rmchk2   ; check 8000

2F56   26         FF                      rmchk3:     mvi         h,0ffh
2F58   01         00FF                                lxi         b,0ffh   ; check 8155 ram

2F5B   56                                 rmchk1:     mov         d,m      ; save the original data in D
2F5C   AF                                             xra         a
2F5D   77                                             mov         m,a      ; 1st write 0
2F5E   BE                                             cmp         m
2F5F   C2         2F7A                                jnz         ramerr   ; if not =, then error
2F62   3D                                             dcr         a
2F63   77                                             mov         m,a      ; next write FF
2F64   BE                                             cmp         m
2F65   C2         2F7A                                jnz         ramerr
2F68   3E         5A                                  mvi         a,5ah
2F6A   77                                             mov         m,a      ; last write 5A
2F6B   BE                                             cmp         m
2F6C   C2         2F7A                                jnz         ramerr   ; if not =, then error
2F6F   72                                             mov         m,d      ; restore original data
2F70   23                                             inx         h
2F71   0B                                             dcx         b        ; dec # of bytes to check
2F72   78                                             mov         a,b
2F73   B1                                             ora         c
2F74   C2         2F5B                                jnz         rmchk1
2F77   C3         2F88                                jmp         uartch   ; no RAM ERRORS CHECK UART

2F7A   CD         011D                    ramerr:     call        daddr    ; display the address of error
2F7D   3E         09                                  mvi         a,9
2F7F   CD         00CC                                call        regprnt  ; show "br" in right two  displays
2F82   CD         016C                                call        beep
2F85   CD         0133                    ramerr1:    call        rdkey    ; wait here so displays aren't trashed

                                                      ;
                                                      ; Now do checks of I/O devices
                                                      ;
                                                      ; See if there is a UART
2F88   AF                                 uartch:     xra         a
2F89   32         FFF7                                sta         uartflg  ; assume there is a uart

2F8C   3E         23                                  mvi         a,23h    ; enable tx only
2F8E   D3         81                                  out         sercom
2F90   DB         80                                  in          serdta   ; discard char that might be waiting
2F92   DB         81                                  in          sercom   ; RxRDY must be low now
2F94   E6         02                                  ani         2        ; see if RxRDY is low
2F96   C2         302F                                jnz         nouart   ; if high, no UART

2F99   3E         27                                  mvi         a,27h
2F9B   D3         81                                  out         sercom   ; enable Tx and Rx

2F9D   CD         3098                                call        randchk  ; pause then check for variation
2FA0   C2         302F                                jnz         nouart

                                                      ; See if transmit works
2FA3   3E         0D                                  mvi         a,0dh    ; A = CR
2FA5   D3         80                                  out         serdta
2FA7   D3         80                                  out         serdta   ; fill the double buffer
2FA9   DB         81                                  in          sercom
2FAB   E6         01                                  ani         1        ; isolate transmit ready bit
2FAD   C2         302F                                jnz         nouart   ; if txrdy, no uart (it should be full)
2FB0   21         FFFF                                lxi         h,0ffffh
2FB3   CD         0181                                call        dlay     ; give time to empty
2FB6   2B                                             dcx         h
2FB7   CD         0181                                call        dlay

2FBA   DB         81                                  in       sercom
2FBC   E6         01                                  ani         1        ; isolate transmit ready bit
2FBE   CA         302F                                jz       nouart      ; if not txrdy, no uart

2FC1   3E         26                                  mvi         a,26h    ; disable transmit
2FC3   D3         81                                  out         sercom
2FC5   3E         20                                  mvi         a,' '
2FC7   D3         80                                  out         serdta   ; send UART a space
2FC9   D3         80                                  out         serdta
2FCB   CD         3098                                call        randchk  ; pause and check for variation
```

```
2FCE   C2         302F                                jnz         nouart   ; if fluctuation then no uart

2FD1   3E         27                                  mvi         a,27h    ; enable transmit again
2FD3   D3         81                                  out         sercom
2FD5   DB         81                                  in       sercom      ; if you read this real quick, it shouldn't be empty
2FD7   E6         04                                  ani         100b     ; isolate txempty bit
2FD9   C2         302F                                jnz         nouart   ;if txempty=true then error, because tx is disabled

                                                      ; yes Virginia, there is a UART.

                                                      ; check for local loopback
2FDC   1E         0D                                  mvi         e,cr
2FDE   CD         10BE                                call        conout                  ; send CR to console
2FE1   21         FFFF                                lxi         h,0ffffh
2FE4   CD         0181                                call        dlay                    ; wait for char to finish TXing
2FE7   DB         81                                  in       sercom                     ; get serial port status
2FE9   E6         02                                  ani         2                       ; isolate receive ready bit
2FEB   CA         3026                                jz          looplx                  ; if no key, exit
2FEE   DB         80                                  in          serdta                  ; get key
2FF0   FE         0D                                  cpi         cr
2FF2   C2         3026                                jnz         looplx                  ; if not CR, exit
                                                      ; TX and RX are connected for local loopback. Send 00 to FF
2FF5   0E         00                                  mvi         c,0                     ; start with 0
2FF7   21         FFFF                    loopl0:     lxi         h,0ffffh                ; timeout delay
2FFA   79                                             mov         a,c
2FFB   D3         80                                  out         serdta                  ; send c
2FFD   DB         81                      loopl2:     in       sercom                     ; get serial port status
2FFF   E6         02                                  ani         2                       ; isolate receive ready bit
3001   C2         300D                                jnz         loopl3                  ; if key, exit timeout loop
3004   2B                                             dcx         h                       ; dec timeout count
3005   7C                                             mov         a,h
3006   B5                                             ora         l                       ; see if HL=0
3007   C2         2FFD                                jnz         loopl2                  ; check stat again
300A   C3         3039                                jmp         badser                  ; if timeout, error

300D   DB         80                      loopl3:     in          serdta                  ; get key read from UART
300F   B9                                             cmp         c                       ; see if same as transmitted
3010   C2         3039                                jnz         badser                  ; if not same, error
3013   06         84                                  mvi         b,rgtdsp+4              ; put uart data on left pair
3015   CD         010B                                call        disbyt

3018   0C                                             inr         c
3019   C2         2FF7                                jnz         loopl0                  ; send next char if not 0
                                                      ; UART tests ok, so disable further tests
301C   06         85                                  mvi         b,rgtdsp+5              ; select left pair
301E   3E         0E                                  mvi         a,14                    ; show L.L. (local loopback)
3020   CD         00CE                                call        regprn1
3023   C3         3044                                jmp         disuart                 ; disable UART tests

3026                          looplx:
3026   11         2F04                                lxi         d,commsg
3029   CD         10CD                                call        pstrng                  ; print "UART test"
302C   C3         3049                                jmp         iochk

302F   06         85                      nouart:     mvi         b,rgtdsp+5              ; select left pair
3031   3E         0D                                  mvi         a,13                    ; show N.U.
3033   CD         00CE                                call        regprn1
3036   C3         3044                                jmp         disuart

3039   3E         0C                      badser:     mvi         a,12                    ; bad serial port error
303B   CD         00CC                                call        regprnt                 ; display "b.s." on left 2 displays
303E   CD         016C                                call        beep
3041   CD         0133                                call        rdkey                   ; wait here so displays aren't trashed

3044   3E         01                      disuart:    mvi         a,1                     ; disable UART test
3046   32         FFF7                                sta         uartflg                 ; uart flag > 0 if no uart

                                                      ; This is the main loop
3049   DB         12                      iochk:      in       dip                        ; echo dip to leds
304B   D3         11                                  out         leds
                                                      ; check A/D
304D   CD         1191                                call        adcin                   ; get A/D input
3050   7D                                             mov         a,l
3051   B7                                             ora         a
3052   3E         40                                  mvi         a,01000000b             ; set sod off bit pattern
3054   CA         3059                                jz       iochka                     ; if l=0 then shut off sod
3057   F6         80                                  ori         80h                     ; set sod on  bit pattern
3059   30                                 iochka:     sim
305A   65                                             mov         h,l
305B   CD         015C                                call        sdiv                    ; send value to 8155 timer for sound
305E   7D                                             mov         a,l
305F   CD         0109                                call        ddata                   ; display the a/d input on right pair

3062   CD         0129                    iochk1:     call        plkpad                  ; check keypad
3065   FE         FF                                  cpi         0ffh                    ; see if no key pressed
3067   CA         306F                                jz       iochk2                     ; if no key ready, don't output data
306A   06         82                                  mvi         b,rgtdsp+2              ; choose middle display
306C   CD         010B                                call        disbyt


306F   3A         FFF7                    iochk2:     lda         uartflg                 ; if there is no uart, skip the following
3072   B7                                             ora         a
3073   C2         3095                                jnz         iochk9

3076   DB         81                                  in       sercom                     ; get serial port status
3078   E6         02                                  ani         2                       ; isolate receive ready bit
307A   CA         3095                                jz       iochk9                     ; if no key ready, continue
307D   DB         80                                  in       serdta                     ; get the character
307F   4F                                             mov         c,a                     ; save key in c
3080   06         84                                  mvi         b,rgtdsp+4              ; put uart data on left pair
3082   CD         010B                                call        disbyt

3085   DB         81                                  in        sercom                    ; now send data back to UART
3087   E6         01                                  ani         1                       ; isolate transmit ready bit
3089   CA         3095                                jz       iochk9                     ; if transmit not ready, don't output
308C   79                                             mov         a,c                     ; put char in a
308D   D3         80                                  out         serdta                  ; output to terminal
308F   11         2F0D                                lxi         d,prmpt
3092   CD         10CD                                call        pstrng                  ; display prompt

3095   C3         3049                    iochk9:     jmp         iochk                   ; do it all again

                                          ; This checks for changes in the UART status, indicating no UART
                                          ;
3098   21         FFFF                    randchk:    lxi         h,0ffffh
309B   CD         0181                                call        dlay
309E   DB         81                                  in       sercom      ; read the initial value of sercom
30A0   E6         06                                  ani         110b     ; txempty and rxrdy lines
30A2   4F                                             mov         c,a      ; original value is in c
30A3   06         0A                                  mvi         b,10     ; number of times to check
30A5   21         0200                    rndchk1:    lxi         h,200h
30A8   CD         0181                                call        dlay     ; pause to allow change
30AB   DB         81                                  in       sercom
30AD   E6         06                                  ani         110b     ; check txempty and rxrdy lines again
30AF   B9                                             cmp         c
30B0   C0                                             rnz                  ; if not equal, then return with B > 0
30B1   05                                             dcr         b
30B2   C2         30A5                                jnz         rndchk1
                                                      ; return with z=true after 10 loops without a mistake
30B5   C9                                             ret


30B6   01         3FFE                    checksm:    lxi         b,chksum ; point to end of mem
30B9   0B                                 etst1:      dcx         b        ; start at byte before
30BA   16         00                                  mvi         d,0      ; clear upper byte of de
30BC   0A                                             ldax        b        ; a=byte from (bc)
30BD   5F                                             mov         e,a
30BE   19                                             dad         d        ; hl=hl+de
30BF   78                                             mov         a,b
30C0   B1                                             ora         c        ; see if bc=0
30C1   C2         30B9                                jnz         etst1
30C4   C9                                             ret

                                          ;=========================================================================
                                          ;=========================================================================
                                          ; PROGRAM ENTRY MODE
                                          ; This allows you to change the contents at the address displayed in the
                                          ; left 4 digits.  The contents are displayed in the right two digits.
                                          ; From this mode you can choose any of the direct commands or functions.
                                          ;=========================================================================
30C5                          EMODE:
30C5   2A         FFF5                                LHLD        PCREG    ; HL=PC
30C8   CD         011D                                CALL        DADDR    ; PRINT PC ON LEFT 4 DISPLAYS
30CB   7E                                             MOV         A,M      ; LOAD DATA FROM PC
30CC   CD         0109                                CALL        DDATA    ; PRINT DATA AT PC ON RIGHT TWO DISPLAYS
30CF   6E                                             MOV         L,M      ; L = DATA AT PC

30D0   CD         0133                    NXTKEY:     CALL        RDKEY    ; GET A KEY
30D3   FE         17                      KEYCHK:     CPI         ENTER
30D5   C2         30E4                                JNZ         CHKDEC   ; IF NOT ENTER, CHECK DECREMENT KEY
                                                      ; ENTER WAS PRESSED STORE DATA IN L AT PC AND INC PC
30D8   7D                                             MOV         A,L      ; L IS THE DATA SHOWN IN THE RIGHT 2 DISPLAYS
30D9   2A         FFF5                                LHLD        PCREG
30DC   77                                             MOV         M,A      ; STORE DATA AT PC
30DD   23                                             INX         H        ; POINT TO NEXT ADDRESS
30DE   22         FFF5                                SHLD        PCREG    ; SAVE NEW PC
30E1   C3         30C5                                JMP         EMODE

30E4   FE         16                      CHKDEC:     CPI         DECPC
30E6   C2         30F3                                JNZ         CHKSTP   ; IF NOT DEC, CHECK STEP
30E9   2A         FFF5                                LHLD        PCREG
30EC   2B                                             DCX         H        ; DEC PCREG
30ED   22         FFF5                                SHLD        PCREG
30F0   C3         30C5                                JMP         EMODE

30F3   FE         14                      CHKSTP:     CPI         STEP
30F5   CA         31B7                                JZ       SINGSTP     ; IF NOT STEP, CHECK FOR HEX DIGIT

30F8   FE         10                      KEYCHK1:    CPI         10H
30FA   D2         3107                                JNC         CHKCMD   ; IF A<10 THEN CY=1, IT IS A HEX DIGIT
                                                      ; A DIGIT WAS TYPED IN THE PROGRAM ENTRY MODE, DISPLAY IT ON THE LEFT 2 DISPLAYS
30FD   CD         008A                                CALL        DIGIT2
3100   7D                                             MOV         A,L
3101   CD         0109                                CALL        DDATA    ; DISPLAY NEW DATA
3104   C3         30D0                                JMP         NXTKEY

                                                      ; IS IT A COMMAND KEY?
3107   FE         15                      CHKCMD:     CPI         FUNC
3109   C2         30C5                                JNZ         EMODE    ; IF NOT FUNC THERE IS NO OTHER

                                          ;FUNCTION MODE
                                                      ; FUNC. PRESSED
310C   CD         00C2                                CALL        FUNPRNT  ; PRINT FUNC.
310F   CD         0133                                CALL        RDKEY    ; GET THE NEXT KEY

                                                      ; THESE VALUES ARE USED TO SELECT THE REGISTER
                                                      ;  AF = 0, BC=1, DE=2, HL=3, SP=4,PC=5,BRK = 6, SC = 7
3112   FE         14                                  CPI         STEP     ; STEP/RUN KEY
3114   CA         3204                                JZ       RUN
                                                      ; IF NOT RUN CHECK A/F
3117   FE         0A                                  CPI         0AH
3119   C2         3121                    EMD0:       JNZ         EMD1     ; IF NOT A/F CHECK B/C
                                                      ; CHANGE AF
311C   3E         00                                  MVI         A,0      ; SELECT AF
311E   C3         31A0                                JMP         SHCHNG   ; SHOW VALUE OF REGISTER AND CHANGE IT

3121   FE         0B                      EMD1:       CPI         0BH
3123   C2         312B                                JNZ         EMD2     ; IF NOT B/C CHECK D/E
                                                      ; CHANGE BC
3126   3E         01                                  MVI         A,1      ; SELECT BC
3128   C3         31A0                                JMP         SHCHNG   ; SHOW VALUE OF REGISTER AND CHANGE IT

312B   FE         0C                      EMD2:       CPI         0CH
312D   C2         3135                                JNZ         EMD3     ; IF NOT D/E CHECK H/L
                                                      ; CHANGE DE
3130   3E         02                                  MVI         A,2      ; SELECT DE
3132   C3         31A0                                JMP         SHCHNG   ; SHOW VALUE OF REGISTER AND CHANGE IT

3135   FE         0D                      EMD3:       CPI         0DH
3137   C2         313F                                JNZ         EMD4     ; IF NOT H/L CHECK SP
                                                      ; CHANGE HL
313A   3E         03                                  MVI         A,3      ; SELECT HL
313C   C3         31A0                                JMP         SHCHNG   ; SHOW VALUE OF REGISTER AND CHANGE IT

313F   FE         0E                      EMD4:       CPI         0EH
3141   C2         3149                                JNZ         EMD5     ; IF NOT SP CHECK PC
                                                      ; CHANGE SP
3144   3E         04                                  MVI         A,4      ; SELECT SP
3146   C3         31A0                                JMP         SHCHNG   ; SHOW VALUE OF REGISTER AND CHANGE IT

3149   FE         0F                      EMD5:       CPI         0FH
314B   C2         3153                                JNZ         EMD6     ; IF NOT PC CHECK BP
                                                      ; CHANGE PC
314E   3E         05                                  MVI         A,5      ; SELECT PC
3150   C3         31A0                                JMP         SHCHNG   ; SHOW VALUE OF REGISTER AND CHANGE IT

3153   FE         08                      EMD6:       CPI         8
3155   C2         315D                                JNZ         EMD7     ; IF NOT BP CHECK STACK CONTENTS
                                                      ; CHANGE BP
3158   3E         06                                  MVI         A,6
315A   C3         31A0                                JMP         SHCHNG

315D   FE         09                      EMD7:       CPI         9
315F   C2         316D                                JNZ         EMD8     ; IF NOT SC CHECK SELF TEST
3162   3E         07                                  MVI         A,7      ; SELECT SC
3164   CD         00CC                                CALL        REGPRNT  ;  PRINT IT
                                          ; POINT TO BYTE ON TOP OF USER STACK.  THIS BYTE AND THE ONE BELOW WILL BE
                                          ; REMOVED WHENEVER THERE IS A POP.
3167   2A         FFF3                                LHLD        SPREG
316A   C3         31AF                                JMP         SHCHNG1

316D   FE         01                      EMD8:       CPI         1
316F   CA         2F11                                JZ       SLFTST      ; FUNC. 1 = SELF TESTER

3172   FE         03                                  CPI         3
3174   CA         3300                                JZ          CHEXCON  ; FUNC. 3 = RECEIVE HEX FILE FROM SERIAL PORT

3177   FE         04                                  CPI         4
3179   CA         1814                                JZ          EPRMPRO  ; FUNC. 4 = MENU DRIVEN EPROM PROGRAMMER

317C   FE         02                                  CPI         2
317E   C2         30C5                                JNZ         EMODE                   ; GO BACK TO ENTRY MODE IF NOT FUNC.2

3181   2A         FFEB                                LHLD        AFREG
3184   E5                                             PUSH        H
3185   F1                                             POP         PSW                     ; AF = AFREG
3186   2A         FFED                                LHLD        BCREG
3189   44                                             MOV         B,H
318A   4D                                             MOV         C,L                     ; BC = BCREG
318B   2A         FFEF                                LHLD        DEREG
318E   EB                                             XCHG                                ; DE = DEREG
318F   2A         FFF3                                LHLD        SPREG
3192   F9                                             SPHL                                ; SP = SPREG
3193   2A         FFF5                                LHLD        PCREG
3196   E5                                             PUSH        H                       ; PUT RET ADDRESS ON STACK FOR MON
3197   2A         FFF1                                LHLD        HLREG                   ; HL = HLREG
319A   CD         1000                                CALL        MSERVICES               ; FUNC. 2 = CALL 1000
319D   C3         32A4                                JMP         MON                     ; PRETEND WE SINGLE STEPPED

                                          ; SHOW AND CHANGE REG PAIR WHOSE NUMBER IS IN A.
                                          ;  AF = 0, BC=1, DE=2, HL=3, SP=4,PC=5,BRK = 6, SC = 7

31A0   57                                 SHCHNG:     MOV         D,A      ; D=REG PAIR
31A1   CD         00CC                                CALL        REGPRNT  ; SHOW THE REGISTER NAME
31A4   7A                                             MOV         A,D      ; RESTORE REG PAIR
31A5   21         FFEB                                LXI         H,AFREG
31A8   07                                             RLC         ; A=A*2
31A9   85                                             ADD         L
31AA   6F                                             MOV         L,A
31AB   3E         00                                  MVI         A,0
31AD   8C                                             ADC         H
31AE   67                                             MOV         H,A      ; HL=HL+2*A

31AF   E5                                 SHCHNG1:    PUSH        H        ; HL= POINTER LOW BYTE OF REGISTER IN MEMORY

31B0   7E                                             MOV         A,M
31B1   23                                             INX         H
31B2   66                                             MOV         H,M
```

```
31B3   6F                                       MOV         L,A      ; HL IS NOW DATA AT HL
31B4   C3         32DE                          JMP         DSPHL



                                   ; SINGSTP: performs a software single step.
                                   ; store ei before the instr then jump to EI
                                   ; it will interrupt after the command is executed
                                   ; ALSO check for calls to rom and put ff after them instead of ei before
31B7   2A         FFF5             singstp:     lhld        pcreg
31BA   7E                                       mov         a,m      ; get opcode from next byte to execute
31BB   FE         FF                            cpi         0ffh     ; is the next instruction a RST 7?
31BD   CA         30C5                          jz       emode       ; if it is, don't execute

31C0   20                                       rim
31C1   E6         0D                            ani         1101b    ; clear 6.5 bit
31C3   F6         08                            ori         1000b    ; set mask set enable bit
31C5   30                                       sim                  ; enable 6.5 interrupt (the 6.5 pin is tied high)
31C6   21         32A4                          lxi         h,mon    ; vector the 6.5 interrupt to MON
31C9   22         FFE7                          shld        vec6hlf

31CC   3E         01                            mvi         a,1      ; set single step flag
31CE   32         FFFA                          sta         sstep    ; store flag

31D1   2A         FFF5                          lhld        pcreg    ; get pcreg again
31D4   7E                                       mov         a,m      ; get opcode from next byte to execute
31D5   FE         CD                            cpi         0cdh     ; is it a CALL?
31D7   C2         31F0                          jnz         nocall
                                                ; see if it is a call to rom
31DA   23                                       inx         h
31DB   23                                       inx         h        ; point to high byte of address
31DC   7E                                       mov         a,m      ; A = high byte
31DD   FE         40                            cpi         maxrom
31DF   D2         31F0                          jnc         nocall   ; if > maxrom, this is not a rom call
                                                ; this is a rom call
31E2   23                                       inx         h        ; point to address after the call
31E3   7E                                       mov         a,m
31E4   32         FFFD                          sta         sstemp   ; save the original value
31E7   22         FFFB                          shld        ssadd    ; save the address
31EA   3E         FF                            mvi         a,0ffh
31EC   77                                       mov         m,a      ; store rst 7 at hl
31ED   C3         321D                          jmp         run1     ; run from here

31F0   2A         FFF5             nocall:      lhld        pcreg
31F3   2B                                       dcx         h        ; point to the byte before next instruction
31F4   7E                                       mov         a,m      ; get data
31F5   32         FFFD                          sta         sstemp   ; save it
31F8   3E         FB                            mvi         a,0fbh   ; A= ei instr.
31FA   77                                       mov         m,a      ; store the ei
31FB   22         FFFB                          shld        ssadd    ; save address
31FE   22         FFF5                          shld        pcreg    ; pcreg points to the ei
3201   C3         321D                          jmp         run1     ; run from here

                                   ;
                                   ; RUN : Replaces the data at the breakpoint with a FF,  restores
                                   ; the registers that were stored at MON:
                                   ; and runs from PC
3204   2A         FFF5             run:     lhld          pcreg    ; see if there is a breakpoint
3207   CD         3237                          call        chl2bp   ; where we will start executing
320A   CA         30C5                          jz       emode       ; if BPREG = PCREG then
                                                                     ;  don't execute it
320D   7E                                       mov         a,m      ; a = next byte to execute
320E   FE         FF                            cpi         0ffh     ; is it RST 7?
3210   CA         30C5                          jz       emode       ; don't execute it

3213   2A         FFF7                          lhld        bpreg
3216   7E                                       mov         a,m      ; a=data from program address (hl)
3217   32         FFF9                          sta         bptemp   ; preserve opcode from (hl)
321A   3E         FF                            mvi         a,0ffh
321C   77                                       mov         m,a      ; store a rst 7 at break point
                                                ;restore original values of registers
321D   2A         FFEB             run1:        lhld        afreg
3220   E5                                       push        h
3221   F1                                       pop         psw      ; af is restored
3222   2A         FFED                          lhld        bcreg
3225   44                                       mov         b,h
3226   4D                                       mov         c,l      ; bc is restored
3227   2A         FFEF                          lhld        dereg
322A   EB                                       xchg                 ; de is restored
322B   2A         FFF3                          lhld        spreg
322E   F9                                       sphl                 ; sp is restored
322F   2A         FFF5                          lhld        pcreg
3232   E5                                       push        h        ; push pc so we can return to this address
3233   2A         FFF1                          lhld        hlreg    ; hl is restored
3236   C9                                       ret                  ; return to pc

                                   ; CHL2BP = RETURN Z=1 IF HL = BPREG
3237   3A         FFF7             chl2bp:      lda         bpreg
323A   BD                                       cmp         l
323B   C0                                       rnz                  ; if L<>low byte of bpreg return z=0
                                                ; L = low byte at bpreg
323C   3A         FFF8                          lda         bpreg+1
323F   BC                                       cmp         h        ; if H=hi byte of bpreg ret z=1
                                                                     ; else return z=0
3240   C9                                       ret




                                   ;
                                   ; This is  the entry point for a breakpoint whether it was a single step
                                   ; through a rom call or not.
3241   22         FFF1             bpentry:     shld        hlreg    ; save hl
3244   E1                                       pop         h        ; get ret address
3245   2B                                       dcx         h        ; point to the rst 7
3246   22         FFF5                          shld        pcreg    ; save pc

                                                ; clear breakpoint address and replace original data
3249   F5                                       push        psw      ; this will be POPed into hl at mon1:
324A   CD         3237                          call        chl2bp
                                   ; if there was a hand placed FF or a single step over a call to ROM and
                                   ; it wasn't at the breakpoint address then don't clear the breakpoint
324D   C2         32AC                          jnz         mon1

3250   3A         FFF9                          lda         bptemp   ; a= data that was replaced by ff
3253   2A         FFF7                          lhld        bpreg    ; point to the break point
3256   77                                       mov         m,a      ; restore it to the break point
3257   21         0000                          lxi         h,0
325A   22         FFF7                          shld        bpreg    ; clear breakpoint address
325D   C3         32AC                          jmp         mon1

                                                ; initialize monitor
                                                ; in case reset was pressed after a breakpoint was selected
                                                ; or during single step, restore value at breakpoint
3260                         init_mon:
3260   2A         FFF7                          lhld        bpreg
3263   3A         FFF9                          lda         bptemp
3266   77                                       mov         m,a
3267   2A         FFFB                          lhld        ssadd
326A   3A         FFFD                          lda         sstemp
326D   77                                       mov         m,a

326E   AF                                       xra         a
326F   32         FFFA                          sta         sstep    ; clear single step flag
3272   21         0000                          lxi         h,0
3275   22         FFF7                          shld        bpreg    ; set bp in rom so it is ineffective
3278   22         FFFB                          shld        ssadd    ; set single step in rom  also
                                                ; clear user registers
327B   22         FFEB                          shld        afreg
327E   22         FFED                          shld        bcreg
3281   22         FFEF                          shld        dereg
3284   22         FFF1                          shld        hlreg

3287   31         FFD6                          lxi         sp,userstk
328A   E5                                       push        h        ; push a zero
328B   39                                       dad         sp       ; hl= sp
328C   22         FFF3                          shld        spreg    ; store in spreg

328F   21         0000                          lxi         h,0
```

```
3292   31         FFEB                                lxi         sp,monstk+6
3295   E5                                             push        h        ; set 5.5 vector to 0
3296   E5                                             push        h        ; set 6.5 vector to 0
3297   E5                                             push        h        ; set 7.5 vector to 0
                                          ;
                                          ; start user program at begram+1 so single step will work on the
                                          ; first instruction (this makes room for an EI).
3298   21         FF01                                lxi         h,begram+1
329B   22         FFF5                                shld        pcreg

329E   CD         016C                                call        beep
32A1   C3         30C5                                jmp         emode    ; go to entry mode

                                          ; SAVE REGS IN RAM
32A4   22         FFF1                    mon:        SHLD        HLREG    ; hl is saved
32A7   E1                                             POP         H        ; HL=RET ADDRESS
32A8   22         FFF5                                shld        pcreg    ; pc is saved
32AB   F5                                             push        psw

32AC   E1                                 mon1:       pop         h        ; hl = af
32AD   22         FFEB                                shld        afreg    ; af is saved
32B0   EB                                             XCHG        ; HL=DE
32B1   22         FFEF                                SHLD        DEREG    ; de is saved
32B4   60                                             MOV         H,B
32B5   69                                             MOV         L,C      ; HL=BC
32B6   22         FFED                                SHLD        BCREG    ; bc is saved
32B9   21         0000                                LXI         H,0
32BC   39                                             DAD         SP       ; HL=SP
32BD   22         FFF3                                SHLD        SPREG    ; sp is saved

32C0   31         FFE5                                lxi         sp,monstk               ; point to monitor stack

                                                      ; check for single step flag
32C3   3A         FFFA                                lda         sstep
32C6   B7                                             ora         a
32C7   CA         30C5                                jz       emode       ; if not single step go to entry mode
32CA   20                                             rim
32CB   E6         0F                                  ani         1111b    ; we only want interrupt mask status
32CD   F6         0A                                  ori         1010b    ; a 1 disables the interrupt
32CF   30                                             sim                  ; disable 6.5 interrupt
32D0   AF                                             xra         a
32D1   32         FFFA                                sta         sstep    ; clear single step flag
32D4   2A         FFFB                                lhld        ssadd    ; hl= address of ei or rst 7
32D7   3A         FFFD                                lda         sstemp   ; get original data
32DA   77                                             mov         m,a      ; restore the original value
32DB   C3         30C5                                jmp         emode

32DE   CD         011D                    dsphl:      call        daddr    ; display hl
32E1   CD         0133                                call        rdkey
32E4   FE         17                                  cpi         enter
32E6   CA         32F8                                jz       entreg      ; if enter, change reg to value of hl
32E9   FE         10                                  cpi         10h
32EB   D2         32F4                                jnc         otherk   ; if >= 10 then it isn't a digit
32EE   CD         0093                                call        digit4
32F1   C3         32DE                                jmp         dsphl    ; display new hl

32F4   E1                                 otherk:     pop         h        ; clean stack
32F5   C3         30D3                                jmp         keychk   ; see what command it was and do it

32F8   EB                                 entreg:     xchg        ; de=hl
32F9   E1                                             pop         h        ; hl points to low byte of register pair
32FA   73                                             mov         m,e
32FB   23                                             inx         h
32FC   72                                             mov         m,d      ; de is saved in the selected ram register
32FD   C3         30C5                                jmp         emode

                                          ;
                                          ; RECEIVE AN INTEL HEX FILE
                                          ;
3300                          CHEXCON:
3300   11         0504                                LXI         D,0504H  ; START AT DISPLAY 5 AND OUTPUT 4 CHARS
3303   21         3323                                LXI         H,CHEXMS1
3306   CD         148A                                CALL        LEDSTR   ; SHOW 'REC.'
3309   CD         332A                                CALL        HEXCON
330C   E6         0E                                  ANI         1110B    ; MASK OFF ALL BUT ERROR BITS
330E   CA         3320                                JZ          CHEXCN1
3311   CD         0109                                CALL        DDATA    ; SHOW ACCUM IN RIGHT PAIR OF DISPLAYS
3314   11         0503                                LXI         D,0503H  ; START AT DISPLAY 5 AND OUTPUT 3 CHARS
3317   21         3327                                LXI         H,CHEXMS2
331A   CD         148A                                CALL        LEDSTR
331D   CD         0133                                CALL        RDKEY    ; PAUSE FOR KEY
3320   C3         30C5                    CHEXCN1:    JMP         EMODE

3323   05         97 9B 08    CHEXMS1:    DB          5,97H,9BH,8 ; 'rEC..'
3327   97         05 05                   CHEXMS2:    DB          97H,5,5                 ; 'Err.'
                                          ;
                                          ;THIS ROUTINE LOADS A INTEL HEX FILE INTO THE PRIMER'S MEMORY.
                                          ;THE TOP OF THE STACK SHOULD HAVE THE ADDRESS WHERE THE FIRST LINE OF THE HEX
                                          ;FILE WILL BE STORED.  THIS IS POPed INTO DE THEN DE IS CHANGED TO  A DIS-
                                          ;PLACEMENT VALUE BY SUBTRACTING THE FIRST HEX FILE LOADING ADDRESS FROM DE.
                                          ;THE DISPLACEMENT OF DE IS ADDED TO THE LOAD ADDRESSES OF EACH LINE OF HEX.
                                          ;IF THE LAST HEX LINE'S DATA LENGTH IS 0 THEN HL WILL CONTAIN THE OPTIONAL
                                          ;STARTING ADDRESS AS GIVEN BY THE LAST HEX LINE.

                                          ;AFTER THE ROUTINE RETURNS, THE BITS IN E INDICATE THE FOLLOWING ERRORS, IF SET
                                          ;BIT#    = ERROR
                                          ;1       = CHECKSUM ERROR
                                          ;2       = NON HEX CHAR ENCOUNTERED
                                          ;3       = ESC CHARACTER ENCOUNTERED.

                                          ;THE BITS IN E INDICATE THE FOLLOWING CONDITIONS
                                          ;BIT#    = CONDITION
                                          ;0       = THIS IS RECORD TYPE 1 (AN END RECORD)
                                          ;6       = IF SET, DE CONTAINS THE STARTING ADDRESS
                                          ;        = IF RESET, DE CONTAINS THE DISPLACEMENT
                                          ;7       = THIS IS THE LAST HEX LINE.  HL CONTAINS THE START ADDRESS

                                          ;AFTER AN ERROR OCCURS, HL CONTAINS THE VALUE IT HAD AFTER THE ERROR OCCURED

332A                          HEXCON:
332A   11         0000                                LXI         D,0

332D                          HEX1CON:
332D   F3                                          DI ;DISABLE INTERUPTS

332E   3E         00                      STRT1:      MVI         A,0      ;===TEST WAS 40H           ;SET BIT 6,SO DE CONTAINS THE DISP
3330   F5                                             PUSH        PSW      ;SAVE ERROR FLAGS ON STACK


3331                          GETCOLON:
3331   CD         33F6                                CALL        GETCHAR
3334   FE         1B                                  CPI         1BH      ;===TEST
3336   CA         33B1                                JZ       ESC1        ;IF ESC,QUIT
3339   FE         3A                                  CPI         ':'      ;IS IT A COLON?
333B   C2         3331                                JNZ         GETCOLON ;IF NOT, GET THE NEXT CHAR

333E   0E         00                                  MVI         C,0      ;CLEAR RUNNING SUM

                                                      ;GET THE RECORD LENGTH
3340   CD         33B8                                CALL        HEX2BIN
3343   CA         33B1                                JZ       ESC1        ;IF ESC, QUIT

3346   47                                             MOV         B,A      ;B = RECORD LENGTH

3347   81                                             ADD         C        ;ADD TO RUNNING SUM
3348   4F                                             MOV         C,A      ;STORE IN C


                                                      ;GET THE LOAD ADDRESS AND STORE IN HL
3349   CD         33B8                                CALL        HEX2BIN
334C   CA         33B1                                JZ       ESC1        ;IF ESC, QUIT
334F   67                                             MOV         H,A      ;SAVE THE START ADDRESS HI

3350   81                                             ADD         C        ;ADD TO RUNNING SUM
3351   4F                                             MOV         C,A      ;STORE IN C

3352   CD         33B8                                CALL        HEX2BIN
3355   CA         33B1                                JZ       ESC1        ;IF ESC, QUIT
3358   6F                                             MOV         L,A      ;SAVE THE START ADDRESS LO
```

```

3359   81                                             ADD         C        ;ADD TO RUNNING SUM
335A   4F                                             MOV         C,A      ;STORE IN C

                                                      ;SEE IF DE HAS BEEN CHANGED FROM STORAGE ADDRESS TO DISPLACEMENT
335B   F1                                             POP         PSW      ;GET DE DISPLACEMENT FLAG
335C   F5                                             PUSH        PSW
335D   E6         40                                  ANI         40H      ;SEE IF BIT 6 IS SET
335F   CA         336C                                JZ          DISPLACE ;IF NOT SET, ADD DISPLACEMENT

3362   F1                                             POP         PSW      ;GET ERRORS FROM STACK
3363   EE         40                                  XRI         40H      ;RESET DISP FLAG
3365   F5                                             PUSH        PSW      ;SAVE DISP FLAG

                                                      ;DE=DE-HL TO GET THE OFFSET THAT WILL BE ADDED TO THE
                                                      ;POINTER TO STORE THE DATA.
3366   7B                                             MOV         A,E
3367   95                                             SUB         L
3368   5F                                             MOV         E,A
3369   7A                                             MOV         A,D
336A   9C                                             SBB         H
336B   57                                             MOV         D,A      ; DE=DE-HL ?Y

336C                          DISPLACE:
336C   19                                             DAD         D        ;ADD OFFSET TO START ADDRESS

                                                      ;GET THE RECORD TYPE
336D   CD         33B8                                CALL        HEX2BIN
3370   CA         33B1                                JZ       ESC1        ;IF ESC, QUIT
3373   FE         01                                  CPI         1        ;IF A=1, IT IS AN END RECORD
3375   C2         337E                                JNZ         SKP1

3378   F1                                             POP         PSW      ;GET ERRORS FROM STACK
3379   F6         01                                  ORI         1        ;SET BIT 0, INDICATE END RECORD
337B   F5                                             PUSH        PSW

337C   3E         01                                  MVI         A,1      ;RESTORE A FOR CHECKSUM ADDING

337E                          SKP1:
337E   81                                             ADD         C        ;ADD TO RUNNING SUM
337F   4F                                             MOV         C,A      ;STORE IN C

3380   AF                                             XRA         A
3381   B8                                             CMP         B        ;IF B=0 THERE IS NO DATA
3382   C2         338C                                JNZ         NEXTRD   ;IF B<>0 THEN READ THE DATA

3385   F1                                             POP         PSW      ;GET ERRORS FROM STACK
3386   F6         80                                  ORI         80H      ;SET BIT 7 OF E
3388   F5                                             PUSH        PSW

3389   C3         339A                                JMP         CHECKSUM ;GET THE CHECKSUM

                                                      ;READ AND CONVERT THE HEX FILE DATA

338C                          NEXTRD:
338C   CD         33B8                                CALL        HEX2BIN
338F   CA         33B1                                JZ       ESC1        ;IF ESC, QUIT
3392   77                                             MOV         M,A      ;STORE AT HL
3393   81                                             ADD         C
3394   4F                                             MOV         C,A      ;KEEP SUM IN C

3395   23                                             INX         H        ;POINT TO THE NEXT STORAGE
3396   05                                             DCR         B        ;DECREMENT RECORD COUNTER
3397   C2         338C                                JNZ         NEXTRD   ;READ THE NEXT BYTE


                                                      ;GET THE CHECKSUM BYTE AND ADD IT TO THE RUNNING SUM
339A                          CHECKSUM:
339A   CD         33B8                                CALL        HEX2BIN
339D   CA         33B1                                JZ       ESC1        ;IF ESC, QUIT
33A0   81                                             ADD         C
33A1   CA         33A8                                JZ       CHEKERR     ;IF RESULT IS 0, DATA IS CORRECT

33A4   F1                                             POP         PSW      ;GET ERRORS FROM STACK
33A5   F6         02                                  ORI         2        ;SET BIT 1= CHECKSUM ERROR
33A7   F5                                             PUSH        PSW

33A8                          CHEKERR:
33A8   F1                                             POP         PSW      ;GET ERRORS FROM STACK
33A9   B7                                             ORA         A        ;SEE IF A=0 (NO ERRORS)
33AA   F5                                             PUSH        PSW
33AB   CA         3331                                JZ       GETCOLON    ;IF NO ERRORS, GET NEXT HEX LINE
33AE   C3         33B5                                JMP         NOESCEXIT               ;NO ESC ENCOUNTERED

33B1                          ESC1:
33B1   F1                                             POP         PSW      ;GET ERRORS FROM STACK
33B2   F6         08                                  ORI         8        ;SET BIT 3= ESC KEY FOUND
33B4   F5                                             PUSH        PSW

33B5                          NOESCEXIT:
33B5   FB                                             EI          ;ENABLE INTERUPTS
33B6   F1                                          POP            PSW      ;GET ERRORS FROM STACK
33B7   C9                                             RET


                                                      ; ASCII TO HEX CONVERSION
                                                      ; THIS SUBROUTINE CHANGES ASCII CHARS
                                                      ; FROM GETCHAR TO BINARY STORED IN THE ACCUMULATOR
                                                      ; ZF=1 IF ESC KEY ENCOUNTERED AND 0 OTHERWISE

33B8                          HEX2BIN:
33B8   C5                                             PUSH        B
33B9   0E         00                                  MVI         C,0      ;CLEAR C
33BB   06         02                                  MVI         B,2      ;# OF CHARS TO CONVERT

33BD                          NEXTCHR:
33BD   CD         33F6                                CALL        GETCHAR  ;PUT CHAR IN A
33C0   CA         33F4                                JZ       ESCEXIT     ;EXIT IF ESC CHAR ENCOUNTERED

33C3   FE         30                                  CPI         '0'      ;SEE IF GREATER THAN 0
33C5   DA         33EB                                JC       HEXERR      ;IF LESS THAN 0 THEN GIVE ERROR CODE
33C8   FE         3A                                  CPI         ':'      ; ':' IS THE CHAR AFTER '9'
33CA   D2         33D2                                JNC         LETTERS  ;IF NOT 0..9, SEE IF A..F

33CD   E6         0F                                  ANI         0FH      ;MASK OFF THE 1ST NIBBLE
33CF   C3         33DE                                JMP         SKIP     ;SKIP OVER LETTER CONVERSION

33D2   FE         41                      LETTERS:    CPI         'A'      ;SEE IF GREATER THAN A;
33D4   DA         33EB                                JC       HEXERR      ;IF < A AND > 0 THEN ERROR
33D7   FE         47                                  CPI         'G'      ;SEE IF A..F
33D9   D2         33EB                                JNC         HEXERR   ;IF NOT < G GIVE AN ERROR

33DC   D6         37                                  SUI         37H      ;CONVERT TO BINARY ('A'=101 - 91 = 10)

33DE   B1                                 SKIP:       ORA         C        ;C=HI NIBBLE MASKED INTO A
33DF   05                                             DCR         B        ;ITERATION COUNTER
33E0   CA         33F3                                JZ       FINISH      ;IF B = 0  THEN FINISHED

33E3   07                                             RLC         ;SHIFT TO THE HI NIBBLE
33E4   07                                             RLC
33E5   07                                             RLC
33E6   07                                             RLC

33E7   4F                                             MOV         C,A      ;PRESERVE 1ST NIBBLE

33E8   C3         33BD                                JMP         NEXTCHR  ;DO THE NEXT CONVERSION

33EB                          HEXERR:
33EB   C1                                             POP         B        ;REMOVE BC FROM STACK
33EC   D1                                             POP         D        ;REMOVE RETURN ADDRESS FROM STACK
33ED   F1                                             POP         PSW      ;GET ERRORS FROM STACK
33EE   F6         04                                  ORI         4        ;SET BIT 2=NON HEX ERROR
33F0   F5                                             PUSH        PSW      ;RESTORE ERRORS TO STACK
33F1   D5                                             PUSH        D        ;PUT RETURN ADDRESS ON STACK
33F2   C5                                             PUSH        B        ;PUT BC BACK


33F3   04                                 FINISH:     INR         B        ;CLEAR ZF TO INDICATE NO ESC FOUND
33F4   C1                                 ESCEXIT:    POP         B        ;RESTORE BC PAIR
```

```
33F5   C9                                          RET

                                          ;
                                          ; Get a char from the serial port and return it in Accumulator.
                                          ;
33F6   DB         81                      GETCHAR: in       sercom         ; get serial port status
33F8   E6         02                                  ani         2        ; isolate receive ready bit
33FA   CA         33F6                                jz       getchar     ; loop until bit set
33FD   DB         80                                  in       serdta      ; get the character
33FF   C9                                             ret


                                                      org         chksum
3FFE   7357                               dw          07357h      ; 2's comp of sum of data before this

                                                      org         monstk
                                          ;
                                          ; SYSTEM VARIABLES

FFE5                          vec5hlf: defs           2           ; vector for 5.5 interrupt
FFE7                          vec6hlf: defs           2           ; vector for 6.5 interrupt
FFE9                          vec7hlf: defs           2           ; vector for 7.5 interrupt
FFEB                          afreg:   defs           2           ; af
FFED                          bcreg:   defs           2           ; bc
FFEF                          dereg:   defs           2           ; de
FFF1                          hlreg:   defs           2           ; hl
FFF3                          spreg:   defs           2           ; sp
FFF5                          CLKDUM:     ; dummy register to store data in RTC
FFF5                          pcreg:   defs           2           ; pc
FFF7                          uartflg:    ; used during self test to tell that UART exists
FFF7                          bpreg:   defs           2           ; break point
FFF9                          bptemp:  defs           1           ; byte that was replaced by FF
FFFA                          SSTEP:   defs           1           ; SINGLE STEP FLAG
FFFB                          ssadd:   defs           2           ; address of the ei or ff instruction
FFFD                          sstemp:  defs           1           ; holds value replaced by  ei or rst 7 instructions
FFFE                          ETYP:    defs           1           ; EPROM type for EPROM programmer

```