APPENDIX C

SYSTEM 80/20 MONITOR LISTING

```
TITLE '80/20 MONITOR, V 1.2, 12 JLY 77'
;
;
;*****************************************************************
;*****************************************************************
;
;                       80/20  MONITOR
;                       (MON82)
;                       VERSION 1.2
;                       12 JLY 1977
;
;*****************************************************************
;*****************************************************************
;
;    (C) 1976 INTEL CORPORATION. ALL RIGHTS RESERVED. NO PART OF THIS
;    PROGRAM OR PUBLICATION MAY BE REPRODUCED, TRANSMITTED, TRANSCRIBED,
;    STORED IN A RETRIEVAL SYSTEM, OR TRANSLATED INTO ANY LANGUAGE OR
;    COMPUTER LANGUAGE, IN ANY FORM OR BY ANY MEANS, ELECTRONIC,
;    MECHANICAL, MAGNETIC, OPTICAL, CHEMICAL, MANUAL OR OTHERWISE,
;    WITHOUT THE PRIOR WRITTEN PERMISSION OF INTEL CORPORATION,
;    3065 BOWERS AVENUE, SANTA CLARA, CALIFORNIA 95051.
;
;*****************************************************************
;*****************************************************************
;
;
; ABSTRACT
; ========
;
; THIS PROGRAM RUNS ON THE SBC 80/20 BOARD AND IS DESIGNED TO PROVIDE
; THE USER WITH A MINIMAL MONITOR.  BY USING THIS PROGRAM,
; THE USER CAN EXAMINE AND CHANGE MEMORY OR CPU REGISTERS, LOAD
; A PROGRAM (IN ABSOLUTE HEX) INTO RAM, AND EXECUTE INSTRUCTIONS
; ALREADY IN MEMORY.  THE MONITOR ALSO PROVIDES THE USER WITH
; ROUTINES FOR PERFORMING CONSOLE I/O AND PAPER TAPE I/O.
;
;
; PROGRAM ORGANIZATION
; ======= ============
;
; THE LISTING IS ORGANIZED IN THE FOLLOWING WAY. THE FIRST ROUTINE
; IS THE COMMAND RECOGNIZER, WHICH IS THE HIGHEST LEVEL
; ROUTINE IN THE PROGRAM. NEXT, ARE THE ROUTINES TO IMPLEMENT
; THE VARIOUS COMMANDS, FOLLOWED BY THE INTERRUPT HANDLERS,
; AND FINALLY THE UTILITY ROUTINES WHICH ACTUALLY DO THE DIRTY WORK.
;
; WITHIN EACH SECTION, THE ROUTINES ARE ORGANIZED IN ALPHABETICAL
; ORDER, BY ENTRY POINT OF THE ROUTINE.
;
; THE 80/20 MONITOR CAN RESID[illegible]N TWO 8708 PROMS,
```

```
                ; BOTH OF WHICH ARE REQUIRED FOR MONITOR OPERATIONS.
                ;
                ; THIS PROGRAM EXPECTS TO RUN IN THE FIRST 2K OF ADDRESS SPACE.
                ; IF, FOR SOME REASON, THE PROGRAM IS RE-ORG'ED, CARE SHOULD
                ; BE TAKEN TO MAKE SURE THAT THE TRANSFER INSTRUCTIONS FOR RST 1
                ; ARE ADJUSTED APPROPRIATELY.
                ;
                ; THE PROGRAM ALSO EXPECTS THAT RAM LOCATIONS 3F80H TO 3FEDH,
                ; INCLUSIVE, ARE RESERVED FOR THE PROGRAM'S OWN USE.  THESE
                ; LOCATIONS MAY BE ALTERED, HOWEVER, BY CHANGING THE EQU'ED
                ; SYMBOL "DATA" AS DESIRED.
                ;
                ;
0000                    ORG     0H
                ;
                ;*****************************************************************
                ;
                ;
                ;                       MONITOR EQUATES
                ;
                ;
                ;*****************************************************************
                ;
                ;
001B            BRCHR   EQU     1BH     ; CODE FOR BREAK CHARACTER (ESCAPE)
03FA            BRTAB   EQU     3FAH    ; LOCATION OF START OF BRANCH TABLE IN ROM
0027            CMD     EQU     027H    ; COMMAND INSTRUCTION FOR USART INITIALIZATION
00ED            CNCTL   EQU     0EDH    ; CONSOLE (USART) CONTROL PORT
00EC            CNIN    EQU     0ECH    ; CONSOLE INPUT PORT
00EC            CNOUT   EQU     0ECH    ; CONSOLE OUTPUT PORT
00ED            CONST   EQU     0EDH    ; CONSOLE STATUS INPUT PORT
000D            CR      EQU     0DH     ; CODE FOR CARRIAGE RETURN
4000            DATA    EQU     16*1024 ; END OF MONITOR RAM USAGE
001B            ESC     EQU     1BH     ; CODE FOR ESCAPE CHARACTER
000F            HCHAR   EQU     0FH     ; MASK TO SELECT LOWER HEX CHAR FROM BYTE
00FF            INVRT   EQU     0FFH    ; MASK TO INVERT HALF BYTE FLAG
000A            LF      EQU     0AH     ; CODE FOR LINE FEED
                ;LSGNON EQU     ---     ; LENGTH OF SIGNON MESSAGE - DEFINED LATER
004E            MODE    EQU     04EH    ; MODE SET FOR USART INITIALIZATION
                ;MSTAK  EQU     ---     ; START OF MONITOR STACK - DEFINED LATER
                ;NCMDS  EQU     ---     ; NUMBER OF VALID COMMANDS
000F            NEWLN   EQU     0FH     ; MASK FOR CHECKING MEMORY ADDR DISPLAY
007F            PRTYO   EQU     07FH    ; MASK TO CLEAR PARITY BIT FROM CONSOLE CHAR
3F00            REGS    EQU     DATA-48 ; START OF REGISTER SAVE AREA
0002            RDR     EQU     2       ; MASK TO TEST RECEIVER STATUS
                ;RTABS  EQU     ---     ; SIZE OF ENTRY IN RTAB TABLE
001B            TERM    EQU     1BH     ; CODE FOR ICMD TERMINATING CHARACTER (ESCAPE)
0001            TRDY    EQU     1       ; MASK TO TEST TRANSMITTER STATUS.
00FF            UPPER   EQU     0FFH    ; DENOTES UPPER HALF OF BYTE IN ICMD
3F80            USAREA  EQU     DATA-128        ; START OF USER STACK AREA
```

```
0004            TXDE    EQU     04H     ; USART TRANSMITTER BUFFER EMPTY
0027            TTYADV  EQU     27H     ; TTY READER ADVANCE COMMAND
0025            TTYSTP  EQU     25H     ; TTY READER STOP COMMAND
008B            ONEMS   EQU     139     ; 1 MILLISECOND CONSTANT
0000            IMASK   EQU     0       ; INTERRUPT MASK VALUE
00DB            MSKPT   EQU     0DBH    ; INT. MASK PORT
00DA            ICCP    EQU     0DAH    ; INT. CONTROLLER COMMAND PORT
3FE0            JTBL    EQU     DATA-32 ; START OF JUMP TABLE IN RAM
00F6            ICW1    EQU     (JTBL AND 0E0H) + 16H ; INTERRUPT CMD WORD 1
003F            ICW2    EQU     JTBL SHR 8 ; INTERRUPT COMMAND WORD 2
00DB            OCW3    EQU     0DBH    ; INTERRUPT OPERATION COMMAND WORD 3
0020            EOIC    EQU     020H    ; END OF INTERRUPT CMD WORD
00DF            TMCP    EQU     0DFH    ; INTERVAL TIMER COMMAND PORT
00DC            CTR0    EQU     0DCH    ; COUNTER 0 PORT
00DD            CTR1    EQU     0DDH    ; COUNTER 1 PORT
00DE            CTR2    EQU     0DEH    ; COUNTER 2 PORT
0000            MSVC0   EQU     0       ; MOST SIG. VAL. FOR CTR 0
0020            LSVC0   EQU     20H     ; LEAST SIG. VAL. FOR CTR 0
0030            COM0    EQU     030H    ; CTR 0 TO MODE 0; SINGLE STEP CMD WORD
0007            B9600   EQU     007D    ; RATE FACTOR FOR 9600 BAUD
0263            B0110   EQU     611D    ; RATE FACTOR FOR 110 BAUD
0055            CHARR   EQU     055H    ; CODE FOR BAUD RATE RECOGNITION CHAR 'U'
0086            C2M3    EQU     086H    ; CTR 2 TO MODE 3 COMMAND WORD
0040            RSTUST  EQU     040H    ; COMMAND INSTRUCTION TO RESET USART
                ;
                ;*****************************************************************
                ;
                ;
                ;                       MONITOR MACROS
                ;
                ;
                ;*****************************************************************
                ;
                ;
                TRUE    MACRO   WHERE   ; BRANCH IF FUNCTION RETURNS TRUE (SUCCESS)
                        JC      WHERE
                        ENDM
                ;
                FALSE   MACRO   WHERE   ; BRANCH IF FUNCTION RETURNS FALSE (FAILURE)
                        JNC     WHERE
                        ENDM
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ;                       USART INITIALIZATION CODE
                ;
                ;
                ;*****************************************************************
```

```
                ;
                ;
                ;   THE USART IS ASSUMED TO COME UP IN THE RESET POSITION (THIS
                ;   FUNCTION IS TAKEN CARE OF BY THE HARDWARE).  THE USART WILL
                ;   BE INITIALIZED IN THE SAME WAY FOR EITHER A TTY OR CRT
                ;   INTERFACE EXCEPT FOR THE NUMBER OF STOP BITS USED. TWO STOP
                ;   BITS WILL BE USED WHEN THE 110 BAUD RATE HAS BEEN SELECTED
                ;   AND ONE STOP BIT WILL BE USED ON ALL OTHER BAUD RATES.
                ;   THE FOLLOWING PARAMETERS ARE USED:
                ;
                ;       MODE INSTRUCTION
                ;       ==== ===========
                ;
                ;       2 STOP BITS FOR 110 BAUD
                ;       1 STOP BIT FOR ALL OTHER BAUD RATES
                ;       PARITY DISABLED
                ;       8 BIT CHARACTERS
                ;       BAUD RATE FACTOR OF 16
                ;
                ;       COMMAND INSTRUCTION
                ;       ======= ===========
                ;
                ;       NO HUNT MODE
                ;       NOT(RTS) FORCED TO 0
                ;       RECEIVE ENABLED
                ;       TRANSMIT ENABLED
                ;
0000 3E4E               MVI     A,MODE      ;4E
0002 D3E9               OUT     CNCTL   ; OUTPUT MODE SET TO USART
0004 C3DA03             JMP     INUST   ; BRANCH TO COMPLETE USART INITIALIZATION
0007 00                 NOP             ; FILLER
                ;
                ;*****************************************************************
                ;
                ;
                ;                       RESTART ENTRY POINT
                ;
                ;
                ;*****************************************************************
                ;
                ;
                GO:
0008 F3                 DI              ; DISABLE INTERRUPTS ON MONITOR ENTRANCE
0009 CD4805             CALL    REGSV   ; SAVE ALL USER REGISTERS
000C C3B802             JMP     ADROUT
                ;
                ;*************************************************************
000A C3DA03     ;
                ; BRANCH TABLE FOR USER ACCESSIBLE ROUTINES
                ;
```

```
                        ;
000F C30703             JMP     CO      ; CONSOLE OUT
0012 C3F402             JMP     CI      ; CONSOLE IN
0015 C37805             JMP     RI      ; READER IN
0018 C30905             JMP     PO      ; PUNCH OUT
                        ;
                        ;*****************************************************************
                        ;
                        ;
                        CPYRT:
001B 28432920           DB      '(C) 1976 INTEL CORP'
001F 31393736
0023 20494E54
0027 454C2043
002B 4F5250
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ;*****************************************************
                        ;
                        ;                       PRINT SIGNON MESSAGE
                        ;
                        ;*****************************************************
                        ;
                        SOMSG:
002E 213706             LXI     H,SGNON ; GET ADDRESS OF SIGNON MESSAGE
0031 0619               MVI     B,LSGNON        ; COUNTER FOR CHARACTERS IN MESSAGE
                        MSGL:
0033 4E                 MOV     C,M     ; FETCH NEXT CHAR TO C REG
0034 CD0703             CALL    CO      ; SEND IT TO THE CONSOLE
0037 23                 INX     H       ; POINT TO NEXT CHARACTER
0038 05                 DCR     B       ; DECREMENT BYTE COUNTER
0039 C23300             JNZ     MSGL    ; RETURN FOR NEXT CHARACTER
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ;
                        ;                   COMMAND RECOGNIZING ROUTINE
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ; FUNCTION: GETCM
                        ; INPUTS: NONE
```

```
                ; OUTPUTS: NONE
                ; CALLS: GETCH,ECHO,ERROR
                ; DESTROYS: A,B,C,H,L,F/F'S
                ; DESCRIPTION: GETCM RECEIVES AN INPUT CHARACTER FROM THE USER
                ;              AND ATTEMPTS TO LOCATE THIS CHARACTER IN ITS COMMAND
                ;              CHARACTER TABLE.  IF SUCCESSFUL, THE ROUTINE
                ;              CORRESPONDING TO THIS CHARACTER IS SELECTED FROM
                ;              A TABLE OF COMMAND ROUTINE ADDRESSES, AND CONTROL
                ;              IS TRANSFERRED TO THIS ROUTINE.  IF THE CHARACTER
                ;              DOES NOT MATCH ANY ENTRIES, CONTROL IS PASSED TO
                ;              THE ERROR HANDLER.
                ;
                GETCM:
003C 31003F             LXI     SP,MSTAK ; ALWAYS WANT TO RESET STACK PTR TO MONITOR
                                         ; /STARTING VALUE SO ROUTINES NEEDING CLEAN UP
003F 0E2E               MVI     C,'.'    ; PROMPT CHARACTER TO C
0041 CD2103             CALL    ECHO     ; SEND PROMPT CHARACTER TO USER TERMINAL
0044 CD4803             CALL    GETCH    ; GET COMMAND CHARACTER TO A
0047 CD2103             CALL    ECHO     ; ECHO CHARACTER TO USER
004A 79                 MOV     A,C      ; PUT COMMAND CHARACTER INTO ACCUMULATOR
004B 010900             LXI     B,NCMDS  ; C CONTAINS LOOP AND INDEX COUNT
004E 216406             LXI     H,CTAB   ; HL POINTS INTO COMMAND TABLE
                GTC05:
0051 BE                 CMP     M        ; COMPARE TABLE ENTRY AND CHARACTER
0052 CA5900             JZ      GTC10    ; BRANCH IF EQUAL - COMMAND RECOGNIZED
0055 23                 INX     H        ; ELSE, INCREMENT TABLE POINTER
0056 0D                 DCR     C        ; DECREMENT LOOP COUNT
0057 C25100             JNZ     GTC05    ; BRANCH IF NOT AT TABLE END
005A C33A03             JMP     ERROR    ; ELSE, COMMAND CHARACTER IS ILLEGAL
                GTC10:
005D 215006             LXI     H,CADR   ; IF GOOD COMMAND, LOAD ADDRESS OF TABLE
                                         ; /OF COMMAND ROUTINE ADDRESSES
0060 09                 DAD     B        ; ADD WHAT IS LEFT OF LOOP COUNT
0061 09                 DAD     B        ; ADD AGAIN - EACH ENTRY IN CADR IS 2 BYTES LONG
0062 7E                 MOV     A,M      ; GET LSP OF ADDRESS OF TABLE ENTRY TO A
0063 23                 INX     H        ; POINT TO NEXT BYTE IN TABLE
0064 66                 MOV     H,M      ; GET MSP OF ADDRESS OF TABLE ENTRY TO H
0065 6F                 MOV     L,A      ; PUT LSP OF ADDRESS OF TABLE ENTRY INTO L
0066 E9                 PCHL             ; NEXT INSTRUCTION COMES FROM COMMAND ROUTINE
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ;            COMMAND IMPLEMENTING ROUTINES
                ;
                ;
                ;*****************************************************************
                ;
                ;
```

```
                ; FUNCTION: DCMD
                ; INPUTS: NONE
                ; OUTPUTS: NONE
                ; CALLS: ECHO,NMOUT,HILO,GETCM,CROUT,GETNM
                ; DESTROYS: A,B,C,D,E,H,L,F/F'S
                ; DESCRIPTION: DCMD IMPLEMENTS THE DISPLAY MEMORY (D) COMMAND
                ;
                DCMD:
0067 0E02               MVI.    C,2     ; GET TWO NUMBERS FROM INPUT STREAM
0069 CD8303             CALL    GETNM
006C D1                 POP     D       ; ENDING ADDRESS TO DE
006D E1                 POP     H       ; STARTING ADDRESS TO HL
                DCM05:
006E CD1203             CALL    CROUT   ; ECHO CARRIAGE RETURN/LINE FEED
0071 CDAF02             CALL    ADRD    ; DISPLAY ADDRESS
                DCM10:
0074 0E20               MVI     C,' '
0076 CD2103             CALL    ECHO    ; USE BLANK AS SEPARATOR
0079 7E                 MOV     A,M     ; GET CONTENTS OF NEXT MEMORY LOCATION
007A CD6704             CALL    NMOUT   ; DISPLAY CONTENTS
007D CDC602             CALL    BREAK   ; SEE IF USER WANTS OUT
                +       TRUE    EXIT    ; IF SO, BRANCH TO EXIT
0080 DA3F03     +       JC      0033FH

0083 CDCE03             CALL    HILO    ; SEE IF ADDRESS OF DISPLAYED LOCATION IS
                                        ; /GREATER THAN OR EQUAL TO ENDING ADDRESS
                +       TRUE    EXIT    ; EXIT IF NO MORE TO DISPLAY
0086 DA3F03     +       JC      0033FH

0089 23                 INX     H       ; IF MORE TO GO, POINT TO NEXT LOC TO DISPLAY
008A 7D                 MOV     A,L     ; GET LOW ORDER BITS OF NEW ADDRESS
008B E60F               ANI     NEWLN   ; SEE IF LAST HEX DIGIT OF ADDRESS DENOTES
                                        ; /START OF NEW LINE
008D C27400             JNZ     DCM10   ; NO - NOT AT END OF LINE
0090 C36E00             JMP     DCM05   ; YES - START NEW LINE WITH ADDRESS
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION: GCMD
                ; INPUTS: NONE
                ; OUTPUTS: NONE
                ; CALLS: ERROR,GETHX,RSTTF
                ; DESTROYS: A,B,C,D,E,H,L,F/F'S
                ; DESCRIPTION: GCMD IMPLEMENTS THE BEGIN EXECUTION (G) COMMAND.
                ;
                GCMD:
0093 CD4F03             CALL    GETHX   ; GET ADDRESS (IF PRESENT) FROM INPUT STREAM
                +       FALSE   GCM05   ; BRANCH IF NO NUMBER PRESENT
```

```
0096 D2A800  +          JNC     000A8H

0099 7A                 MOV     A,D     ; ELSE, GET TERMINATOR
009A FE0D               CPI     CR      ; SEE IF CARRIAGE RETURN
009C C23A03             JNZ     ERROR   ; ERROR IF NOT PROPERLY TERMINATED
009F 21D83F             LXI     H,PSAVE ; WANT NUMBER TO REPLACE SAVE PGM COUNTER
00A2 71                 MOV     M,C
00A3 23                 INX     H
00A4 70                 MOV     M,B
00A5 C3AE00             JMP     GCM10
                GCM05:
00A8 7A                 MOV     A,D     ; IF NO STARTING ADDRESS, MAKE SURE THAT
00A9 FE0D               CPI     CR      ; /CARRIAGE RETURN TERMINATED COMMAND
00AB C23A03             JNZ     ERROR   ; ERROR IF NOT
                GCM10:
00AE AF                 XRA     A       ; RESET SINGLE STEP FLAG FOR GO CMD
00AF C3B305             JMP     RSTTF   ; RESTORE REGISTERS AND BEGIN EXECUTION

                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION: ICMD
                ; INPUTS: NONE
                ; OUTPUTS: NONE
                ; CALLS: ERROR,ECHO,GETCH,VALDL,VALDG,CNVBN,STHLF,GETNM,CROUT
                ; DESTROYS: A,B,C,D,E,H,L,F/F'S
                ; DESCRIPTION: ICMD IMPLEMENTS THE INSERT CODE INTO MEMORY (I) COMMAND.
                ;
                ICMD:
0082 0E01               MVI     C,1
0084 CD8303             CALL    GETNM   ; GET SINGLE NUMBER FROM INPUT STREAM
0087 3EFF               MVI     A,UPPER
0089 32DC3F             STA     TEMP    ; TEMP WILL HOLD THE UPPER/LOWER HALF BYTE FLAG
008C D1                 POP     D       ; ADDRESS OF START TO DE
                ICM05:
008D CD4803             CALL    GETCH   ; GET A CHARACTER FROM INPUT STREAM
0090 CD2103             CALL    ECHO    ; ECHO IT
0093 79                 MOV     A,C     ; PUT CHARACTER BACK INTO A
0094 FE1B               CPI     TERM    ; SEE IF CHARACTER IS A TERMINATING CHARACTER
0096 CAF200             JZ      ICM25   ; IF SO, ALL DONE ENTERING CHARACTERS
0099 CD2406             CALL    VALDL   ; ELSE, SEE IF VALID DELIMITER
                +       TRUE    ICM05   ; IF SO SIMPLY IGNORE THIS CHARACTER
009C DABD00  +          JC      000BDH

009F CD0906             CALL    VALDG   ; ELSE, CHECK TO SEE IF VALID HEX DIGIT
                +       FALSE   ICM20   ; IF NOT, BRANCH TO HANDLE ERROR CONDITION
00A2 D2EC00  +          JNC     000ECH
```

```
00D5 CDFE02             CALL    CNVBN   ; CONVERT DIGIT TO BINARY
00D8 4F                 MOV     C,A     ; MOVE RESULT TO C
00D9 CDEA05             CALL    STHLF   ; STORE IN APPROPRIATE HALF WORD
00DC 3ADC3F             LDA     TEMP    ; GET HALF BYTE FLAG
00DF B7                 ORA     A       ; SET F/F'S
00E0 C2E400             JNZ     ICM10   ; BRANCH IF FLAG SET FOR UPPER
00E3 13                 INX     D       ; IF LOWER, INC ADDRESS OF BYTE TO STORE IN
                ICM10:
00E4 EEFF               XRI     INVRT   ; TOGGLE STATE OF FLAG
00E6 32DC3F             STA     TEMP    ; PUT NEW VALUE OF FLAG BACK
00E9 C3B800             JMP     ICM05   ; PROCESS NEXT DIGIT
                ICM20:
00EC CDDF05             CALL    STHF0   ; ILLEGAL CHARACTER
00EF C33A03             JMP     ERROR   ; MAKE SURE ENTIRE BYTE FILLED THEN ERROR
                ICM25:
00F2 CDDF05             CALL    STHF0   ; HERE FOR ESCAPE CHARACTER - INPUT IS DONE
00F5 C33F03             JMP     EXIT
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION: MCMD
                ; INPUTS: NONE
                ; OUTPUTS: NONE
                ; CALLS: GETCM,HILO,GETNM
                ; DESTROYS: A,B,C,D,E,H,L,F/F'S
                ; DESCRIPTION: MCMD IMPLEMENTS THE MOVE DATA IN MEMORY (M) COMMAND.
                ;
                MCMD:
00F8 0E03               MVI     C,3
00FA CD2303             CALL    GETNM   ; GET 3 NUMBERS FROM INPUT STREAM
00FD C1                 POP     B       ; DESTINATION ADDRESS TO BC
00FE E1                 POP     H       ; ENDING ADDRESS TO HL
00FF D1                 POP     D       ; STARTING ADDRESS TO DE
                MCM05:
0100 E5                 PUSH    H       ; SAVE ENDING ADDRESS
0101 62                 MOV     H,D
0102 6B                 MOV     L,E     ; SOURCE ADDRESS TO HL
0103 7E                 MOV     A,M     ; GET SOURCE BYTE
0104 60                 MOV     H,B
0105 69                 MOV     L,C     ; DESTINATION ADDRESS TO HL
0106 77                 MOV     M,A     ; MOVE BYTE TO DESTINATION
0107 03                 INX     B       ; INCREMENT DESTINATION ADDRESS
0108 78                 MOV     A,B
0109 B1                 ORA     C       ; TEST FOR DESTINATION ADDRESS OVERFLOW
010A CA3C00             JZ      GETCM   ; IF SO, CAN TERMINATE COMMAND
010D 13                 INX     D       ; INCREMENT SOURCE ADDRESS
010E E1                 POP     H       ; ELSE, GET BACK ENDING ADDRESS
010F CDC803             CALL    HILO    ; SEE IF ENDING ADDR>=SOURCE ADDR
```

```
                +       FALSE   GETCH   ; IF NOT, COMMAND IS DONE
0112 D23C00     +       JNC     0003CH
0115 C30001             JMP     MCM05   ; MOVE ANOTHER BYTE
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION MCMD
                ; INPUTS: NONE
                ; OUTPUTS: NONE
                ; CALLS: CROUT
                ; DESTROYS: A
                ; DESCRIPTION: MCMD IMPLEMENTS THE SINGLE STEP (M) COMMAND
                ;
0118 CD1203     MCMD:   CALL    CROUT   ; ECHO CR/LF
0118 3EFF               MVI     A,0FFH  ; SET SINGLE STEP FLAG
011D C3B305             JMP     RSTTF   ; RESTORE REGISTERS AND EXECUTE NEXT INST.
                ;
                ;
                ;*****************************************************************
                ;
                ; FUNCTION RCMD
                ; INPUTS: NONE
                ; OUTPUTS: NONE
                ; CALLS: GETCH,ECHO,CO,RICH,BYTE
                ; DESTROYS: A,B,C,D,E,H,L,F/F'S
                ; DESCRIPTION: RCMD IMPLEMENTS THE READ HEXADECIMAL TAPE (R)
                ;               COMMAND.
                ;
                RCMD:
0120 CD4803             CALL    GETCH   ; GET CARRIAGE RETURN CHARACTER
0123 CD2103             CALL    ECHO    ; ECHO IT
0126 79                 MOV     A,C     ; MOVE IT TO A REGISTER
0127 FE0D               CPI     CR      ; SEE IF CARRIAGE RETURN
0129 C23A03             JNZ     ERROR   ; ERROR IF NOT PROPERLY TERMINATED
                RCM05:
012C CDAA05             CALL    RICH    ; READ CHARACTER FROM TAPE
012F FE3A               CPI     ':'     ; SEE IF RECORD MARK
0131 C22C01             JNZ     RCM05   ; TRY AGAIN IF NOT MARK
0134 AF                 XRA     A       ; ZERO A REGISTER
0135 57                 MOV     D,A     ; INITIALIZE D FOR HOLDING THE CHECKSUM
0136 CDD902             CALL    BYTE    ; READ TWO CHARACTERS FROM TAPE
0139 CA3C00             JZ      GETCH   ; IF ZERO RECORD LENGTH, ALL DONE
013C 5F                 MOV     E,A     ; OTHERWISE, PUT THE RECORD LENGTH IN E
013D CDD902             CALL    BYTE    ; GET MSB OF LOAD ADDRESS
0140 67                 MOV     H,A     ; MOVE TO H
0141 CDD902             CALL    BYTE    ; GET LSB OF LOAD ADDRESS
```

```
0144 6F                 MOV     L,A     ; MOVE TO L
0145 CDD902             CALL    BYTE    ; GET RECORD TYPE
0148 48                 MOV     C,E     ; MOVE RECORD LENGTH TO C
                RCM10:
0149 CDD902             CALL    BYTE    ; READ DATA FROM TAPE
014C 77                 MOV     M,A     ; PUT DATA INTO MEMORY
014D 23                 INX     H       ; INCREMENT HL FOR NEXT LOCATION
014E 1D                 DCR     E       ; DECREMENT RECORD LENGTH
014F C24901             JNZ     RCM10   ; LOOP UNTIL DONE
0152 CDD902             CALL    BYTE    ; READ CHECKSUM FROM TAPE
0155 C23A03             JNZ     ERROR   ; CHECKSUM ERROR IF NOT ZERO
0158 C32C01             JMP     RCM05   ; GET ANOTHER RECORD
                )
                )
                )
                ;*****************************************************************
                ;
                ;
                ; FUNCTION: SCMD
                ; INPUTS: NONE
                ; OUTPUTS: NONE
                ; CALLS: GETHX,GETCM,NMOUT,ECHO
                ; DESTROYS: A,B,C,D,E,H,L,F/F'S
                ; DESCRIPTION: SCMD IMPLEMENTS THE SUBSTITUTE INTO MEMORY (S) COMMAND.
                ;
                SCMD:
015B CD4F03             CALL    GETHX   ; GET A NUMBER, IF PRESENT, FROM INPUT
015E C5                 PUSH    B
015F E1                 POP     H       ; GET NUMBER TO HL - DENOTES MEMORY LOCATION
                SCM05:
0160 7A                 MOV     A,D     ; GET TERMINATOR
0161 FE20               CPI     ' '     ; SEE IF SPACE
0163 CA6801             JZ      SCM10   ; YES - CONTINUE PROCESSING
0166 FE2C               CPI     ','     ; ELSE, SEE IF COMMA
0168 C23C00             JNZ     GETCM   ; NO - TERMINATE COMMAND
                SCM10:
016B 7E                 MOV     A,M     ; GET CONTENTS OF SPECIFIED LOCATION TO A
016C CD8704             CALL    NMOUT   ; DISPLAY CONTENTS ON CONSOLE
016F 0E2D               MVI     C,'-'
0171 CD2103             CALL    ECHO    ; USE DASH FOR SEPARATOR
0174 CD4F03             CALL    GETHX   ; GET NEW VALUE FOR MEMORY LOCATION, IF ANY
                +       FALSE   SCM15   ; IF NO VALUE PRESENT, BRANCH
0177 D27B01     +       JNC     00178H

017A 71                 MOV     M,C     ; ELSE, STORE LOWER 8 BITS OF NUMBER ENTERED
                SCM15:
017B 23                 INX     H       ; INCREMENT ADDRESS OF MEMORY LOCATION TO VIEW
017C C36001             JMP     SCM05
                )
                )
```

```
                ;*****************************************************************
                ;
                ;
                ; FUNCTION WCMD
                ; INPUTS: NONE
                ; OUTPUTS: NONE
                ; CALLS: GETNM,LEAD,PO,PBYTE,PADR,PEOL,PEOF
                ; DESTROYS: A,B,C,D,E,H,L,F/F'S
                ; DESCRIPTION: WCMD IMPLEMENTS THE WRITE HEXADECIMAL TAPE (W)
                ;              COMMAND.
                ;
                WCMD:
017F 0E02               MVI     C,2
0181 CD8303             CALL    GETNM   ; GET 2 NUMBERS FROM INPUT STREAM
0184 CD7804             CALL    LEAD    ; PUNCH 60 NULL CHARACTERS FOR TAPE LEADER
0187 D1                 POP     D       ; ENDING ADDRESS TO DE
0188 E1                 POP     H       ; STARTING ADDRESS TO HL
                WCM05:
0189 7D                 MOV     A,L     ; MOVE L TO A
018A C610               ADI     16      ; INCREMENT THE LSB OF STARTING ADDRESS BY 16
018C 4F                 MOV     C,A     ; MOVE RESULT TO C
018D 7C                 MOV     A,H     ; MOVE H TO A
018E CE00               ACI     0       ; ADD CARRY IN FROM PREVIOUS OPERATION
0190 47                 MOV     B,A     ; SAVE RESULT IN B
0191 7B                 MOV     A,E     ; NOW MOVE LSB OF ENDING ADDRESS TO A
0192 91                 SUB     C       ; SUBTRACT LSB OF STARTING ADDRESS
0193 4F                 MOV     C,A     ; SAVE IN C
0194 7A                 MOV     A,D     ; NOW GET MSB OF ENDING ADDRESS IN A
0195 98                 SBB     B       ; SUBTRACT MSB OF STARTING ADDRESS
0196 DA9E01             JC      WCM10   ; BRANCH IF THE RECORD LENGTH IS NOT 16
0199 3E10               MVI     A,16    ; OTHERWISE SET A TO RECORD LENGTH OF 16
019B C3A101             JMP     WCM15   ; NOW BRANCH TO PUNCH THE RECORD
                WCM10:
019E 79                 MOV     A,C     ; THIS IS THE LAST RECORD
019F C611               ADI     17      ; SO SET A TO REMAINING DATA LENGTH
                WCM15:
01A1 B7                 ORA     A       ; CHECK FOR RECORD LENGTH OF ZERO
01A2 CACE01             JZ      WCM25   ; IF IT IS, ALL DONE
01A5 D5                 PUSH    D       ; OTHERWISE, SAVE ENDING ADDRESS
01A6 5F                 MOV     E,A     ; PUT RECORD LENGTH IN E
01A7 1600               MVI     D,0     ; INITIALIZE D FOR HOLDING CHECKSUM
01A9 0E3A               MVI     C,':'
01AB CD0905             CALL    PO      ; PUNCH RECORD MARK CHARACTER
01AE 7B                 MOV     A,E     ; PUT RECORD LENGTH IN A
01AF CDC904             CALL    PBYTE   ; PUNCH RECORD LENGTH
01B2 CDC004             CALL    PADR    ; PUNCH STARTING ADDRESS
01B5 AF                 XRA     A       ; ZERO A
01B6 CDC904             CALL    PBYTE   ; PUNCH RECORD TYPE
                WCM20:
01B9 7E                 MOV     A,M     ; GET DATA TO BE PUNCHED FROM MEMORY
```

```
016A CDC904          CALL    PBYTE   ; PUNCH IT
016D 23              INX     H       ; INCREMENT MEMORY ADDRESS
016E 1D              DCR     E       ; DECREMENT LENGTH COUNT
016F C28901          JNZ     WCM20   ; LOOP UNTIL ALL DATA PUNCHED
01C2 AF              XRA     A
01C3 92              SUB     D       ; PUNCH CHECKSUM
01C4 CDC904          CALL    PBYTE
01C7 D1              POP     D       ; RESTORE ENDING ADDRESS
01C8 CDFE04          CALL    PEOL    ; PUNCH CARRIAGE RETURN AND LINE FEED
01CB C38901          JMP     WCM05
                     WCM25:
01CE CDE004          CALL    PEOF    ; PUNCH END OF FILE RECORD
01D1 C33F03          JMP     EXIT    ; ALL DONE
                     ;
                     ;*****************************************************
                     ;
                     ;
                     ; FUNCTION: XCMD
                     ; INPUTS: NONE
                     ; OUTPUTS: NONE
                     ; CALLS: GETCH,ECHO,REGDS,GETCM,ERROR,RGADR,NMOUT,CROUT,GETHX
                     ; DESTROYS: A,B,C,D,E,H,L,F/F'S
                     ; DESCRIPTION: XCMD IMPLEMENTS THE REGISTER EXAMINE AND CHANGE (X)
                     ;              COMMAND.
                     ;
                     XCMD:
01D4 CD4803          CALL    GETCH   ; GET REGISTER IDENTIFIER
01D7 CD2103          CALL    ECHO    ; ECHO IT
01DA 79              MOV     A,C
01DB FE0D            CPI     CR
01DD C2E601          JNZ     XCM05   ; BRANCH IF NOT CARRIAGE RETURN
01E0 CD1705          CALL    REGDS   ; ELSE, DISPLAY REGISTER CONTENTS
01E3 C33CC0          JMP     GETCM   ; THEN TERMINATE COMMAND
                     XCM05:
01E6 4F              MOV     C,A     ; GET REGISTER IDENTIFIER TO C
01E7 CD6105          CALL    RGADR   ; CONVERT IDENTIFIER INTO RTAB TABLE ADDR
01EA C5              PUSH    B
01EB E1              POP     H       ; PUT POINTER TO REGISTER ENTRY INTO HL
01EC 0E20            MVI     C,' '
01EE CD2103          CALL    ECHO    ; ECHO SPACE TO USER
01F1 79              MOV     A,C
01F2 320C3F          STA     TEMP    ; PUT SPACE INTO TEMP AS DELINITER
                     XCM10:
01F5 3A0C3F          LDA     TEMP    ; GET TERMINATOR
01F8 FE20            CPI     ' '     ; SEE IF A BLANK
01FA CA0202          JZ      XCM15   ; YES - GO CHECK POINTER INTO TABLE
01FD FE2C            CPI     ','     ; NO - SEE IF COMMA
01FF C23C00          JNZ     GETCM   ; NO - MUST BE CARRIAGE RETURN TO END COMMAND
                     XCM15:
```

```
0202 7E             MOV     A,M
0203 B7             ORA     A       ; SET F/F'S
0204 CA3F03         JZ      EXIT    ; BRANCH IF AT END OF TABLE
0207 E5             PUSH    H       ; PUT POINTER ON STACK
0208 5E             MOV     E,M
0209 163F           MVI     D,REGS SHR 8    ; FETCH ADDRESS OF SAVE LOCATION FROM
0209 23             INX     H               ; /TABLE
020C 46             MOV     B,M     ; FETCH LENGTH FLAG FROM TABLE
020D D5             PUSH    D       ; SAVE ADDRESS OF SAVE LOCATION
020E D5             PUSH    D
020F E1             POP     H       ; MOVE ADDRESS TO HL
0210 C5             PUSH    B       ; SAVE LENGTH FLAG
0211 7E             MOV     A,M     ; GET 8 BITS OF REGISTER FROM SAVE LOCATION
0212 CD8704         CALL    HMOUT   ; DISPLAY IT
0215 F1             POP     PSW     ; GET BACK LENGTH FLAG
0216 F5             PUSH    PSW     ; SAVE IT AGAIN
0217 B7             ORA     A       ; SET F/F'S
0218 CA2002         JZ      XCM20   ; IF 8 BIT REGISTER, NOTHING MORE TO DISPLAY
0218 2B             DCX     H       ; ELSE, FOR 16 BIT REGISTER, GET LOWER 8 BITS
021C 7E             MOV     A,M
021D CD8704         CALL    HMOUT   ; DISPLAY THEM
                XCM20:
0220 0E2D           MVI     C,'-'
0222 CD2103         CALL    ECHO    ; USE DASH AS SEPARATOR
0225 CD4F03         CALL    GETHX   ; SEE IF THERE IS A VALUE TO PUT INTO REGISTER
                +       FALSE   XCM30   ; NO - GO CHECK FOR NEXT REGISTER
0228 D24002     +       JNC     00240H

0228 7A             MOV     A,D
022C 32DC3F         STA     TEMP    ; ELSE, SAVE THE TERMINATOR FOR NOW
022F F1             POP     PSW     ; GET BACK LENGTH FLAG
0230 E1             POP     H       ; PUT ADDRESS OF SAVE LOCATION INTO HL
0231 B7             ORA     A       ; SET F/F'S
0232 CA3702         JZ      XCM25   ; IF 8 BIT REGISTER, BRANCH
0235 70             MOV     M,B     ; SAVE UPPER 8 BITS
0236 2B             DCX     H       ; POINT TO SAVE LOCATION FOR LOWER 8 BITS
                XCM25:
0237 71             MOV     M,C     ; STORE ALL OF 8 BIT OR LOWER 1/2 OF 16 BIT REG
                XCM27:
0238 110300         LXI     D,RTABS ; SIZE OF ENTRY IN RTAB TABLE
0238 E1             POP     H       ; POINTER INTO REGISTER TABLE RTAB
023C 19             DAD     D       ; ADD ENTRY SIZE TO POINTER
023D C3F501         JMP     XCM10   ; DO NEXT REGISTER
                XCM30:
0240 7A             MOV     A,D     ; GET TERMINATOR
0241 32DC3F         STA     TEMP    ; SAVE IN MEMORY
0244 D1             POP     D       ; CLEAR STACK OF LENGTH FLAG AND ADDRESS
0245 D1             POP     D       ; /OF SAVE LOCATION
0246 C33802         JMP     XCM27   ; GO INCREMENT REGISTER TABLE POINTER
                ;
```

```
                        ;
                        ;*****************************************************************
                        ;
                        ;
                        ;                   INTERRUPT SERVICE ROUTINES
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ;
                        ;*****************************************************************************
                        ;
                        ;
                        ; FUNCTION INTIN
                        ; INPUTS: NONE
                        ; OUTPUTS: NONE
                        ; CALLS: REGSV,ECHO,NMOUT,REGDS
                        ; DESTROYS: A,B,C
                        ; DESCRIPTION: INTIN HANDLES ALL INTERRUPTS NOT HANDLED BY THE USER.
                        ;              IT PRINTS THE INTERRUPT NUMBER, NEXT INSTRUCTION, AND
                        ;              REGISTER VALUES.
                        ;
                        ;
                        INTIN:
0249 CD4805                     CALL    REGSV   ; SAVE ALL USERS REGISTERS
024C 0E49                       MVI     C,'I'
024E CD2103                     CALL    ECHO    ; OUTPUT INTERRUPT MESSAGE 'I=#'
0251 0E3D                       MVI     C,'='
0253 CD2103                     CALL    ECHO
0256 3E0B                       MVI     A,OCW3  ; READ INTERRUPT 'IN SERVICE' REGISTER
0258 D3DA                       OUT     ICCP
025A DBDA                       IN      ICCP
025C 0608                       MVI     B,8     ; SET UP TO FIND INTERRUPT NUMBER
025E 0E00                       MVI     C,0
0260 1F                 FINTN:  RAR             ; ROTATE TO CHECK INTERRUPT 'IS' BIT
0261 DA6902                     JC      FNDI    ; EXIT IF # FOUND
0264 0C                         INR     C
0265 05                         DCR     B       ; TRY AGAIN
0266 C26002                     JNZ     FINTN
0269 79                 FNDI:   MOV     A,C     ; MOVE FOR OUTPUT CONVERSION
026A CD8704                     CALL    NMOUT   ; PRINT INTERRUPT #
026D 0E20                       MVI     C,' '   ; USE SPACE AS DELIMITER
026F CD2103                     CALL    ECHO    ; PRINT IT
0272 CD9A04                     CALL    NXTIN   ; OUTPUT NEXT INSTRUCTION
0275 CD1705                     CALL    REGDS   ; OUTPUT REGISTERS
0278 3E20                       MVI     A,EOIC  ; END OF INTERRUPT
027A D3DA                       OUT     ICCP
027C C33C00                     JMP     GETCM   ; GO GET USER COMMAND
                        ;
                        ;
```

```
                        ;*****************************************************************
                        ;
                        ;
                        ; FUNCTION STEPIN
                        ; INPUTS: NONE
                        ; OUTPUTS: NONE
                        ; CALLS: REGSV,REGDS,NXTIN
                        ; DESTROYS: A,F/F'S
                        ; DESCRIPTION: STEPIN OUTPUTS DATA AFTER SINGLE STEP TIMER INTERRUPT
                        ;
                        ;
                        STEPIN:
027F CD4805                     CALL    REGSV   ; SAVE ALL REGISTERS ON ENTRY
0282 3E20                       MVI     A,EOIC  ; END OF INTERRUPT
0284 D3DA                       OUT     ICCP
0286 3AD93F                     LDA     PSAVE+1 ; TEST FOR SINGLE STEP INTO BREAKPOINT
0289 A7                         ANA     A
028A C2A602                     JNZ     STPOK   ; PC HIGH=0 FOR BREAKPOINT ADDRESS
028D 3AD83F                     LDA     PSAVE
0290 FE0F                       CPI     0FH     ; PC LOW<F FOR BREAKPOINT ADDRESS
0292 D2A602                     JNC     STPOK   ; CONTINUE IF NO USER BREAKPOINT
0295 2ADA3F                     LHLD    SSAVE   ; GET USER STACK POINTER
0298 5E                         MOV     E,M     ; RESTORE ADDRESS OF USER BREAKPOINT
0299 23                         INX     H
029A 56                         MOV     D,M
029B 23                         INX     H
029C 22DA3F                     SHLD    SSAVE   ; UPDATE USER STACK POINTER
029F EB                         XCHG            ; GET BREAKPOINT ADDRESS INTO HL
02A0 22D83F                     SHLD    PSAVE   ; UPDATE USER P COUNTER
02A3 C3B802                     JMP     ADROUT  ; PRINT BREAKPOINT ENTRY
02A6 CD1705             STPOK:  CALL    REGDS   ; OUTPUT REGISTERS
02A9 CD9A04                     CALL    NXTIN   ; OUTPUT 3 BYTES FOR NEXT INSTRUCTION
02AC C33C00                     JMP     GETCM
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ;
                        ;                       UTILITY ROUTINES
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ;
                        ; FUNCTION ADRD
                        ; INPUTS: HL - ADDRESS TO BE DISPLAYED
                        ; OUTPUTS: NONE
```

```
                ; CALLS: NMOUT
                ; DESTROYS: A
                ; DESCRIPTION: ADRD OUTPUTS TO THE CONSOLE THE ADDRESS
                ;              CONTAINED IN THE H,L REGISTERS.
                ;
                ADRD:
02AF 7C                 MOV     A,H     ; DISPLAY FIRST HALF OF ADDRESS
02B0 CD8704             CALL    NMOUT
02B3 7D                 MOV     A,L     ; DISPLAY SECOND HALF OF ADDRESS
02B4 CD8704             CALL    NMOUT
02B7 C9                 RET             ; RETURN TO CALLING ROUTINE
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION ADROUT
                ; INPUTS: USER REGISTERS ON THE STACK
                ; OUTPUTS: NOTHING
                ; CALLS: ECHO,ADRD
                ; DESTROYS: A,B,C,D,E,H,L,F/F'S
                ; DESCRIPTION: ADROUT OUTPUTS THE USER P COUNTER TO THE CONSOLE
                ;              AFTER AN RST 1 INSTRUCTION.
                ;
                ADROUT:
02B8 0E23               MVI     C,'#'
02BA CD2103             CALL    ECHO    ; OUTPUT '#'
02BD 2AD83F             LHLD    PSAVE   ; LOAD USER P COUNTER
02C0 CDAF02             CALL    ADRD    ; DISPLAY ADDRESS
02C3 C33F03             JMP     EXIT    ; GET NEW COMMAND
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION: BREAK
                ; INPUTS: NONE
                ; OUTPUTS: CARRY - 1 IF ESCAPE CHARACTER INPUT
                ;                - 0 IF ANY OTHER CHARACTER OR NO CHARACTER PENDING
                ; CALLS: NOTHING
                ; DESTROYS: A,F/F'S
                ; DESCRIPTION: BREAK IS USED TO SENSE AN ESCAPE CHARACTER FROM
                ;              THE USER.  IF NO CHARACTER IS PENDING, OR IF THE
                ;              PENDING CHARACTER IS NOT THE ESCAPE, THEN A FAILURE
                ;              RETURN (CARRY=0) IS TAKEN.  IN THIS CASE, THE
                ;              PENDING CHARACTER (IF ANY) IS LOST.  IF THE PENDING
                ;              CHARACTER IS AN ESCAPE CHARACTER, BREAK TAKES A SUCCESS
                ;              RETURN (CARRY=1).
                ;
                BREAK:
```

```
02C6 DBED               IN      CONST   ; GET CONSOLE STATUS
02C8 E602               ANI     RBR     ; SEE IF CHARACTER PENDING
02CA CA4503             JZ      FRET    ; NO - TAKE FAILURE RETURN
02CD DBEC               IN      CHIN    ; YES - PICK UP CHARACTER
02CF E67F               ANI     PRTYO   ; STRIP OFF PARITY BIT
02D1 FE1B               CPI     BRCHR   ; SEE IF BREAK CHARACTER
02D3 CADD05             JZ      SRET    ; YES - SUCCESS RETURN
02D6 C34503             JMP     FRET    ; NO - FAILURE RETURN - CHARACTER LOST
                        )
                        )
                        ;*****************************************************************
                        ;
                        ;
                        ; FUNCTION BYTE
                        ; INPUTS: D - CURRENT VALUE OF CHECKSUM
                        ; OUTPUTS: A - HEXADECIMAL CHARACTER
                        ;          D - UPDATED VALUE OF CHECKSUM
                        ; CALLS: RICH,CNVBN
                        ; DESTROYS: A,B,C,D,F/F'S
                        ; DESCRIPTION: BYTE READS 2 ASCII CHARACTERS FROM THE [illegible]
                        ;               AND CONVERTS THE CHARACTERS TO ONE HEXADECIMAL CHARACTER.
                        ;               THE A REGISTER CONTAINS THE FINAL CHARACTER AND THE
                        ;               D REGISTER CONTAINS THE UPDATED VALUE OF
                        ;               THE CHECKSUM.
                        ;
                        BYTE:
02D9 C5                 PUSH    B       ; SAVE BC
02DA CDAA05             CALL    RICH    ; READ ASCII CHARACTER FROM TAPE
02DD 4F                 MOV     C,A
02DE CDFE02             CALL    CNVBN   ; CONVERT CHARACTER TO HEXADECIMAL
02E1 07                 RLC             ; POSITION VALUE INTO UPPER 4 BITS
02E2 07                 RLC
02E3 07                 RLC
02E4 07                 RLC
02E5 47                 MOV     B,A     ; SAVE RESULTS IN B
02E6 CDAA05             CALL    RICH    ; GET ANOTHER CHARACTER FROM TAPE
02E9 4F                 MOV     C,A
02EA CDFE02             CALL    CNVBN   ; CONVERT IT
02ED B0                 ORA     B       ; OR IN THE UPPER 4 BITS
02EE 4F                 MOV     C,A     ; SAVE
02EF 82                 ADD     D       ; INCREMENT CHECKSUM
02F0 57                 MOV     D,A
02F1 79                 MOV     A,C     ; RESTORE HEX DATA TO A REGISTER
02F2 C1                 POP     B       ; RESTORE BC
02F3 C9                 RET
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ;
```

```
                ; FUNCTION: CI
                ; INPUTS: NONE
                ; OUTPUTS: A - CHARACTER FROM CONSOLE
                ; CALLS: NOTHING
                ; DESTROYS: A,F/F'S
                ; DESCRIPTION: CI WAITS UNTIL A CHARACTER HAS BEEN ENTERED AT THE
                ;              CONSOLE AND THEN RETURNS THE CHARACTER, VIA THE A
                ;              REGISTER, TO THE CALLING ROUTINE.  THIS ROUTINE
                ;              IS CALLED BY THE USER VIA A JUMP TABLE IN RAM.
                ;
                CI:
02F4 DBED               IN      CONST   ; GET STATUS OF CONSOLE
02F6 E602               ANI     RBR     ; CHECK FOR RECEIVER BUFFER READY
02F8 CAF402             JZ      CI      ; NOT YET - WAIT
02FB DBEC               IN      CNIN    ; READY SO GET CHARACTER
02FD C9                 RET
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION: CNVBN
                ; INPUTS: C - ASCII CHARACTER '0'-'9' OR 'A'-'F'
                ; OUTPUTS: A - 0 TO F HEX
                ; CALLS: NOTHING
                ; DESTROYS: A,F/F'S
                ; DESCRIPTION: CNVBN CONVERTS THE ASCII REPRESENTATION OF A HEX
                ;              CHARACTER INTO ITS CORRESPONDING BINARY VALUE.  CNVBN
                ;              DOES NOT CHECK THE VALIDITY OF ITS INPUT.
                ;
                CNVBN:
02FE 79                 MOV     A,C
02FF D630               SUI     '0'     ; SUBTRACT CODE FOR '0' FROM ARGUMENT
0301 FE0A               CPI     10      ; WANT TO TEST FOR RESULT OF 0 TO 9
0303 F8                 RM              ; IF SO, THEN ALL DONE
0304 D607               SUI     7       ; ELSE, RESULT BETWEEN 17 AND 23 DECIMAL
0306 C9                 RET             ; SO RETURN AFTER SUBTRACTING BIAS OF 7
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION: CO
                ; INPUTS: C - CHARACTER TO OUTPUT TO CONSOLE
                ; OUTPUTS: C - CHARACTER OUTPUT TO CONSOLE
                ; CALLS: NOTHING
                ; DESTROYS: A,F/F'S
                ; DESCRIPTION: CO WAITS UNTIL THE CONSOLE IS READY TO ACCEPT A CHARACTER
                ;              AND THEN SENDS THE INPUT ARGUMENT TO THE CONSOLE.
                ;
```

```
                CO:
0307 DBED               IN      CONST   ; GET STATUS OF CONSOLE
0309 E601               ANI     TRDY    ; SEE IF TRANSMITTER READY
030B CA0703             JZ      CO      ; NO - WAIT
030E 79                 MOV     A,C     ; ELSE, MOVE CHARACTER TO A REGISTER FOR OUTPUT
030F D3EC               OUT     CHOUT   ; SEND TO CONSOLE
0311 C9                 RET
                ;
                ;*****************************************************************
                ;
                ; FUNCTION CROUT
                ; INPUTS: NONE
                ; OUTPUTS: NONE
                ; CALLS: ECHO
                ; DESTROYS: A,B,C,F/F'S
                ; DESCRIPTION: CROUT SENDS A CARRIAGE RETURN (AND HENCE A LINE
                ;               FEED) TO THE CONSOLE.
                ;
                CROUT:
0312 0E0D               MVI     C,CR
0314 CD2103             CALL    ECHO    ; OUTPUT CARRIAGE RETURN TO USER TERMINAL
0317 C9                 RET
                ;
                ;*****************************************************************
                ;
                ; FUNCTION DELAY
                ; INPUTS: NONE
                ; OUTPUTS: NONE
                ; CALLS: NOTHING
                ; DESTROYS: F/F'S
                ; DESCRIPTION: DELAY PROVIDES A PROGRAMMED DELAY OF 1 MILLISECOND
                ;
                DELAY:
0318 C5                 PUSH    B       ; SAVE BC REGISTERS
0319 068B               MVI     B,ONEMS ; LOAD 1 MILLISECOND CONSTANT
                DEL1:
031B 05                 DCR     B       ; DECREMENT COUNTER
031C C21B03             JNZ     DEL1    ; JUMP IF NOT DONE
031F C1                 POP     B       ; RESTORE BC REGISTERS
0320 C9                 RET             ; RETURN TO CALLING ROUTINE
                ;
                ;*****************************************************************
                ;
                ; FUNCTION: ECHO
```

```
                ; INPUTS: C - CHARACTER TO ECHO TO TERMINAL
                ; OUTPUTS: C - CHARACTER ECHOED TO TERMINAL
                ; CALLS: CO
                ; DESTROYS: A,B,F/F'S
                ; DESCRIPTION: ECHO TAKES A SINGLE CHARACTER AS INPUT AND, VIA
                ;               THE MONITOR, SENDS THAT CHARACTER TO THE USER
                ;               TERMINAL.  A CARRIAGE RETURN IS ECHOED AS A CARRIAGE
                ;               RETURN LINE FEED, AND AN ESCAPE CHARACTER IS ECHOED AS $.
                ;
                ECHO:
0321 41                 MOV     B,C     ; SAVE ARGUMENT
0322 3E1B               MVI     A,ESC
0324 B8                 CMP     B       ; SEE IF ECHOING AN ESCAPE CHARACTER
0325 C22A03             JNZ     ECHO5   ; NO - BRANCH
0328 0E24               MVI     C,'$'   ; YES - ECHO AS $
                ECHO5:
032A CD0703             CALL    CO      ; DO OUTPUT THROUGH MONITOR
032D 3E0D               MVI     A,CR
032F B8                 CMP     B       ; SEE IF CHARACTER ECHOED WAS A CARRIAGE RETURN
0330 C23803             JNZ     ECH10   ; NO - NO NEED TO TAKE SPECIAL ACTION
0333 0E0A               MVI     C,LF    ; YES - WANT TO ECHO LINE FEED, TOO
0335 CD0703             CALL    CO
                ECH10:
0338 48                 MOV     C,B     ; RESTORE ARGUMENT
0339 C9                 RET
                ;
                ;*****************************************************************
                ;
                ; FUNCTION: ERROR
                ; INPUTS: NONE
                ; OUTPUTS: NONE
                ; CALLS: ECHO,CROUT,GETCM
                ; DESTROYS: A,B,C,F/F'S
                ; DESCRIPTION: ERROR PRINTS THE ERROR CHARACTER (CURRENTLY A NUMBER SIGN)
                ;               ON THE CONSOLE, FOLLOWED BY A CARRIAGE RETURN-LINE FEED,
                ;               AND THEN RETURNS CONTROL TO THE COMMAND RECOGNIZER.
                ;
                ERROR:
033A 0E23               MVI     C,'#'
033C CD2103             CALL    ECHO    ; SEND # TO CONSOLE
                EXIT:
033F CD1203             CALL    CROUT   ; SKIP TO BEGINNING OF NEXT LINE
0342 C33C00             JMP     GETCM   ; TRY AGAIN FOR ANOTHER COMMAND
                ;
                ;*****************************************************************
                ;
```

```
                ; FUNCTION: FRET
                ; INPUTS: NONE
                ; OUTPUTS: CARRY - ALWAYS 0
                ; CALLS: NOTHING
                ; DESTROYS: CARRY
                ; DESCRIPTION: FRET IS JUMPED TO BY ANY ROUTINE THAT WISHES TO
                ;               INDICATE FAILURE ON RETURN.  FRET SETS THE CARRY
                ;               FALSE, DENOTING FAILURE, AND THEN RETURNS TO THE
                ;               CALLER OF THE ROUTINE INVOKING FRET.
                ;
                FRET:
0345 37                 STC             ; FIRST SET CARRY TRUE
0346 3F                 CMC             ; THEN COMPLEMENT IT TO MAKE IT FALSE
0347 C9                 RET             ; RETURN APPROPRIATELY
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION: GETCH
                ; INPUTS: NONE
                ; OUTPUTS: C - NEXT CHARACTER IN INPUT STREAM
                ; CALLS: CI
                ; DESTROYS: A,C,F/F'S
                ; DESCRIPTION: GETCH RETURNS THE NEXT CHARACTER IN THE INPUT STREAM
                ;               TO THE CALLING PROGRAM.
                ;
                GETCH:
0348 CDF402             CALL    CI      ; GET CHARACTER FROM TERMINAL
0348 E67F               ANI     PRTYO   ; TURN OFF PARITY BIT IN CASE SET BY CONSOLE
034D 4F                 MOV     C,A     ; PUT VALUE IN C REGISTER FOR RETURN
034E C9                 RET
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION: GETHX
                ; INPUTS: NONE
                ; OUTPUTS: BC - 16 BIT INTEGER
                ;          D - CHARACTER WHICH TERMINATED THE INTEGER
                ;          CARRY - 1 IF FIRST CHARACTER NOT DELIMITER
                ;                - 0 IF FIRST CHARACTER IS DELIMITER
                ; CALLS: GETCH,ECHO,VALDL,VALDG,CNVBN,ERROR
                ; DESTROYS: A,B,C,D,E,F/F'S
                ; DESCRIPTION: GETHX ACCEPTS A STRING OF HEX DIGITS FROM THE INPUT
                ;               STREAM AND RETURNS THEIR VALUE AS A 16 BIT BINARY
                ;               INTEGER.  IF MORE THAN 4 HEX DIGITS ARE ENTERED,
                ;               ONLY THE LAST 4 ARE USED.  THE NUMBER TERMINATES WHEN
                ;               A VALID DELIMITER IS ENCOUNTERED.  THE DELIMITER IS
```

```
                ;               ALSO RETURNED AS AN OUTPUT OF THE FUNCTION.  ILLEGAL
                ;               CHARACTERS (NOT HEX DIGITS OR DELIMITERS) CAUSE AN
                ;               ERROR INDICATION.  IF THE FIRST (VALID) CHARACTER
                ;               ENCOUNTERED IN THE INPUT STREAM IS NOT A DELIMITER,
                ;               GETHX WILL RETURN WITH THE CARRY BIT SET TO 1;
                ;               OTHERWISE, THE CARRY BIT IS SET TO 0 AND THE CONTENTS
                ;               OF BC ARE UNDEFINED.
                ;
                GETHX:
034F E5                 PUSH    H       ; SAVE HL
0350 210000             LXI     H,0     ; INITIALIZE RESULT
0353 1E00               MVI     E,0     ; INITIALIZE DIGIT FLAG TO FALSE
                GHX05:
0355 CD4803             CALL    GETCH   ; GET A CHARACTER
0358 CD2103             CALL    ECHO    ; ECHO THE CHARACTER
035B CD2406             CALL    VALDL   ; SEE IF DELIMITER
                +       FALSE   GHX10   ; NO - BRANCH
035E D26D03     +       JNC     0036DH

0361 51                 MOV     D,C     ; YES - ALL DONE, BUT WANT TO RETURN DELIMITER
0362 E5                 PUSH    H
0363 C1                 POP     B       ; MOVE RESULT TO BC
0364 E1                 POP     H       ; RESTORE HL
0365 7B                 MOV     A,E     ; GET FLAG
0366 B7                 ORA     A       ; SET F/F'S
0367 C2DC05             JNZ     SRET    ; IF FLAG NON-0, A NUMBER HAS BEEN FOUND
036A CA4503             JZ      FRET    ; ELSE, DELIMITER WAS FIRST CHARACTER
                GHX10:
036D CD0906             CALL    VALDG   ; IF NOT DELIMITER, SEE IF DIGIT
                +       FALSE   ERROR   ; ERROR IF NOT A VALID DIGIT, EITHER
0370 D23A03     +       JNC     0033AH

0373 CDFE02             CALL    CNVBN   ; CONVERT DIGIT TO ITS BINARY VALUE
0376 1EFF               MVI     E,0FFH  ; SET DIGIT FLAG NON-0
0378 29                 DAD     H       ; *2
0379 29                 DAD     H       ; *4
037A 29                 DAD     H       ; *8
037B 29                 DAD     H       ; *16
037C 0600               MVI     B,0     ; CLEAR UPPER 8 BITS OF BC PAIR
037E 4F                 MOV     C,A     ; BINARY VALUE OF CHARACTER INTO C
037F 09                 DAD     B       ; ADD THIS VALUE TO PARTIAL RESULT
0380 C35503             JMP     GHX05   ; GET NEXT CHARACTER
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION: GETNM
                ; INPUTS: C - COUNT OF NUMBERS TO FIND IN INPUT STREAM
                ; OUTPUTS: TOP OF STACK - NUMBERS FOUND IN REVERSE ORDER (LAST ON TOP
```

```
                ;                       OF STACK)
                ; CALLS: GETHX,HILO,ERROR
                ; DESTROYS: A,B,C,D,E,H,L,F/F'S
                ; DESCRIPTION: GETHM FINDS A SPECIFIED COUNT OF NUMBERS, BETWEEN 1
                ;               AND 3, INCLUSIVE,  IN THE INPUT
                ;               STREAM AND RETURNS THEIR VALUES ON THE STACK.  IF 2
                ;               OR MORE NUMBERS ARE REQUESTED, THEN THE FIRST MUST BE
                ;               LESS THAN OR EQUAL TO THE SECOND, OR THE FIRST AND
                ;               SECOND NUMBERS WILL BE SET EQUAL.  THE LAST NUMBER
                ;               REQUESTED MUST BE TERMINATED BY A CARRIAGE RETURN
                ;               OR AN ERROR INDICATION WILL RESULT
                ;
                GETHM:
0383 2E03               MVI     L,3     ; PUT MAXIMUM ARGUMENT COUNT INTO L
0385 79                 MOV     A,C     ; GET THE ACTUAL ARGUMENT COUNT
0386 E603               ANI     3       ; FORCE TO MAXIMUM OF 3
0388 C8                 RZ              ; IF 0, DON'T BOTHER TO DO ANYTHING
0389 67                 MOV     H,A     ; ELSE, PUT ACTUAL COUNT INTO H
                GNM05:
038A CD4F03             CALL    GETHX   ; GET A NUMBER FROM INPUT STREAM
                +       FALSE   ERROR   ; ERROR IF NOT THERE - TOO FEW NUMBERS
038D D23A03     +       JNC     0033AH

0390 C5                 PUSH    B       ; ELSE, SAVE NUMBER ON STACK
0391 2D                 DCR     L       ; DECREMENT MAXIMUM ARGUMENT COUNT
0392 25                 DCR     H       ; DECREMENT ACTUAL ARGUMENT COUNT
0393 CA9F03             JZ      GNM10   ; BRANCH IF NO MORE NUMBERS WANTED
0396 7A                 MOV     A,D     ; ELSE, GET NUMBER TERMINATOR TO A
0397 FE0D               CPI     CR      ; SEE IF CARRIAGE RETURN
0399 CA3A03             JZ      ERROR   ; ERROR IF SO - TOO FEW NUMBERS
039C C38A03             JMP     GNM05   ; ELSE, PROCESS NEXT NUMBER
                GNM10:
039F 7A                 MOV     A,D     ; WHEN COUNT 0, CHECK LAST TERMINATOR
03A0 FE0D               CPI     CR
03A2 C23A03             JNZ     ERROR   ; ERROR IF NOT CARRIAGE RETURN
03A5 01FFFF             LXI     B,0FFFFH        ; HL GETS LARGEST NUMBER
03A8 7D                 MOV     A,L     ; GET WHAT'S LEFT OF MAXIMUM ARG COUNT
03A9 B7                 ORA     A       ; CHECK FOR 0
03AA CAB203             JZ      GNM20   ; IF YES, 3 NUMBERS WERE INPUT
                GNM15:
03AD C5                 PUSH    B       ; IF NOT, FILL REMAINING ARGUMENTS WITH 0FFFFH
03AE 2D                 DCR     L
03AF C2AD03             JNZ     GNM15
                GNM20:
03B2 C1                 POP     B       ; GET THE 3 ARGUMENTS OUT
03B3 D1                 POP     D
03B4 E1                 POP     H
03B5 CDC803             CALL    HILO    ; SEE IF FIRST >= SECOND
                +       FALSE   GNM25   ; NO - BRANCH
03B8 D2BD03     +       JNC     003BDH
```

```
038B 54                 MOV     D,H
038C 5D                 MOV     E,L     ; YES - MAKE SECOND EQUAL TO THE FIRST
                GNM25:
038D E3                 XTHL            ; PUT FIRST ON STACK - GET RETURN ADDR
038E D5                 PUSH    D       ; PUT SECOND ON STACK
038F C5                 PUSH    B       ; PUT THIRD ON STACK
03C0 E5                 PUSH    H       ; PUT RETURN ADDRESS ON STACK
                GNM30:
03C1 3D                 DCR     A       ; DECREMENT RESIDUAL COUNT
03C2 F8                 RM              ; IF NEGATIVE, PROPER RESULTS ON STACK
03C3 E1                 POP     H       ; ELSE, GET RETURN ADDR
03C4 E3                 XTHL            ; REPLACE TOP RESULT WITH RETURN ADDR
03C5 C3C103             JMP     GNM30   ; TRY AGAIN
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION: HILO
                ; INPUTS: DE - 16 BIT INTEGER
                ;         HL - 16 BIT INTEGER
                ; OUTPUTS: CARRY - 0 IF HL<DE
                ;                - 1 IF HL>=DE
                ; CALLS: NOTHING
                ; DESTROYS: A,F/F'S
                ; DESCRIPTION: HILO COMPARES THE 2 16 BIT INTEGERS IN HL AND DE.  THE
                ;              INTEGERS ARE TREATED AS UNSIGNED NUMBERS.  THE CARRY
                ;              BIT IS SET ACCORDING TO THE RESULT OF THE COMPARISON.
                ;
                HILO:
03C8 C5                 PUSH    B       ; SAVE BC
03C9 47                 MOV     B,A     ; SAVE A REGISTER
03CA 23                 INX     H       ; INCREMENT HL BY 1
03CB 7C                 MOV     A,H     ; WANT TO TEST FOR 0 RESULT AFTER
03CC B5                 ORA     L       ; /INCREMENTING
03CD 2B                 DCX     H       ; RESTORE HL
03CE 37                 STC             ; SET CARRY
03CF CAD703             JZ      HILO5   ; IF SO, CARRY IS SET PROPERLY
03D2 7D                 MOV     A,L     ; IF NOT, MOVE L TO A
03D3 93                 SUB     E       ; SUBTRACT E
03D4 7C                 MOV     A,H     ; MOVE H TO A
03D5 9A                 SBB     D       ; SUBTRACT D WITH BORROW
03D6 3F                 CMC             ; COMPLIMENT CARRY FOR CORRECT CARRY BIT VALUE
                HILO5:
03D7 78                 MOV     A,B     ; RESTORE A
03D8 C1                 POP     B       ; RESTORE BC
03D9 C9                 RET             ; EXIT
                ;
                ;
```

```
                ;*****************************************************************
                ;
                ;
                ; FUNCTION INUST
                ; INPUTS: NONE
                ; OUTPUTS: NOTHING
                ; CALLS: NOTHING
                ; DESTROYS: A,H,L,SP
                ; DESCRIPTION: INUST OUTPUTS TO THE USART THE COMMAND WORD
                ;              AND INITIALIZES THE STACK POINTER.
                ;              ALSO THE INTERVAL TIMER AND THE INTERRUPT
                ;              CONTROLLER ARE INITIALIZED.
                ;
                INUST:
03DA 3E27               MVI     A,CMD
03DC D3ED               OUT     CNCTL   ; OUTPUT COMMAND WORD TO USART
03DE 21903F             LXI     H,MSTAK-64      ; LOAD POINTER TO STACK
03E1 22DA3F             SHLD    SSAVE   ; INITIALIZE USER STACK POINTER
03E4 31D03F             LXI     SP,MSTAK        ; INITIALIZE MONITOR STACK
03E7 3E30               MVI     A,COM0  ; INITIALIZE SINGLE STEP TIMER MODE
03E9 D3DF               OUT     TMCP
03EB 3EB6               MVI     A,C2M3  ; INITIALIZE COUNTER #2 FOR BAUD RATE
03ED D3DF               OUT     TMCP    ; OUTPUT COMMAND WORD TO INTERVAL TIMER
03EF 0608       BRSEL:  MVI     B,08D   ; LOAD '# OF RATES' COUNTER
03F1 210700             LXI     H,B9600 ; LOAD HIGHEST BAUD RATE FACTOR
03F4 3E37       BRS05:  MVI     A,37H   ; RESET USART STATUS ERRORS AND
03F6 D3ED               OUT     CNCTL   ; SET "DTR"
03F8 7D         BRS06:  MOV     A,L     ; LEAST SIGNIFICANT WORD FOR CTR2
03F9 D3DE               OUT     CTR2    ; OUTPUT WORD TO CTR 2
03FB 7C                 MOV     A,H     ; MOST SIGNIFICANT WORD FOR CTR2
03FC D3DE               OUT     CTR2    ; OUTPUT WORD TO CTR2
03FE 11E803             LXI     D,1000  ; SETUP 1 SECOND TIMEOUT
0401 CD1803     BRS07:  CALL    DELAY   ; 1 MS DELAY
0404 1B                 DCX     D       ; DECREMENT TIMER
0405 DBED               IN      CONST   ; INPUT USART STATUS
0407 E602               ANI     RBR     ; CHECK FOR RECEIVER BUFFER READY
0409 C21404             JNZ     BRS08   ; NOT YET - WAIT 1 MS AND CHECK AGAIN
040C 7B                 MOV     A,E     ; TEST FOR ZERO-
040D B2                 ORA     D       ; AFTER DECREMENTING
040E C20104             JNZ     BRS07   ; CONTINUE TO CHECK STATUS FOR 1 SEC
0411 C3EF03             JMP     BRSEL   ; AFTER 1 SEC REINITIALIZE BAUD RATE SEARCH
0414 DBEC       BRS08:  IN      CNIN    ; READY SO GET CHARACTER
0416 E67F               ANI     7FH     ; MASK OFF PARITY BIT
0418 4F                 MOV     C,A     ; SAVE CHAR.
0419 DBED               IN      CNCTL   ; GET USART STATUS
041B E630               ANI     30H     ; MASK ERROR BITS
041D C22904             JNZ     BRS10   ; IF ERROR , GET NEXT CHAR.
0420 79                 MOV     A,C     ; CHAR TO ACC.
0421 FE55               CPI     CHARR   ; COMPARE FOR CORRECT CHAR.
0423 C22904             JNZ     BRS10   ; IF BAD CHAR. GET NEXT ONE
```

```
0426 C35604            JMP     IICR    ; GO TO INTERRUPT INITIALIZATION
0429 0E78      BRS10:  MVI     C,120   ; SETUP 120 MS TIMER
042B CD1803    BRS15:  CALL    DELAY   ; 1 MS DELAY
042E 0D                DCR     C       ; DECREMENT TIMER
042F C22B04            JNZ     BRS15   ; JUMP IF TIMER NOT EXPIRED
0432 DBEC              IN      CHIN    ; CLEAR USART INPUT BUFFER
0434 05                DCR     B       ; DECREMENT RATE COUNTER
0435 78                MOV     A,B     ; MOVE RATE CTR TO ACC.
0436 FE01              CPI     01      ; IS RATE CTR =1?
0438 CA4404            JZ      BRS20   ; GO TO 110 BAUD SELECT IF LSB =1
043B FE00              CPI     00      ; IS RATE CTR =0?
043D CAEF03            JZ      BRSEL   ; IF ZERO , START OVER
0440 29                DAD     H       ; HALVE THE BAUD RATE
0441 C3F403            JMP     BRS05   ; TRY AGAIN
0444 3E40      BRS20:  MVI     A,RSTUST ; USART RESET VALUE
0446 D3ED              OUT     CNCTL   ; RESET USART TO ACCEPT NEW MODE INST.
0448 3ECE              MVI     A,(MODE OR 80H)  ; TWO STOP BITS MODE INSTRUCTION
044A D3ED              OUT     CNCTL   ; LOAD NEW MODE INSTRUCTION
044C 3E35              MVI     A,35H   ; RESET USART STATUS ERRORS AND-
044E D3ED              OUT     CNCTL   ; TURN OFF DTR
0450 216302            LXI     H,B0110 ; LOAD 110 BAUD RATE FACTOR
0453 C3F603            JMP     BRS06   ; TRY AGAIN
                ;
                ;
0456 3EF6      IICR:   MVI     A,ICW1  ; INITIALIZE INTERRUPT CONTROLLER
0458 D30A              OUT     ICCP    ; OUTPUT COMMAND WORD #1
045A 3E3F              MVI     A,ICW2  ;
045C D30B              OUT     ICCP+1  ; OUTPUT COMMAND WORD #2
045E 3E00              MVI     A,IMASK ; INTERRUPT MASK VALUE
0460 D30B              OUT     MSKPT   ; OUTPUT MASK WORD TO CONTROLLER
0462 219006            LXI     H,JPTB  ; LOAD START OF PROM JUMP TABLE
0465 11E03F            LXI     D,JTBL  ; LOAD START OF RAM JUMP TABLE
0468 061F              MVI     B,31D   ; LENGTH OF TABLE IN "B"
046A 7E        MTBL:   MOV     A,M     ; MOVE PROM JUMP TABLE TO RAM
046B 12                STAX    D       ;                    "
046C 23                INX     H       ;                    "
046D 13                INX     D       ;                    "
046E 05                DCR     B       ;                    "
046F C26A04            JNZ     MTBL    ;                    "
0472 21E03F            LXI     H,USAREA        ; INITIALIZE USER STACK POINTER
0475 22DA3F            SHLD    SSAVE
0478 C32E00            JMP     SOMSG   ; GO TO PRINT SIGNON MESSAGE
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION LEAD
                ; INPUTS: NONE
                ; OUTPUTS: NONE
```

```
                ; CALLS: PO
                ; DESTROYS: B,C,F/F'S
                ; DESCRIPTION: LEAD OUTPUTS 60 NULL CHARACTERS TO PAPER TAPE TO FORM A
                ;              LEADER.
                ;
                LEAD:
047B 063C               MVI     B,60    ; LOAD B WITH A COUNT OF 60
                LE05:
047D 0E00               MVI.    C,0
047F CD0905             CALL    PO      ; PUNCH NULL CHARACTER
0482 05                 DCR     B       ; DECREMENT COUNT
0483 C27D04             JNZ     LE05    ; DO IT AGAIN IF NOT DONE
0486 C9                 RET
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION: NMOUT
                ; INPUTS: A - 8 BIT INTEGER
                ; OUTPUTS: NONE
                ; CALLS: ECHO,PRVAL
                ; DESTROYS: A,B,C,F/F'S
                ; DESCRIPTION: NMOUT CONVERTS THE 8 BIT, UNSIGNED INTEGER IN THE
                ;              A REGISTER INTO 2 ASCII CHARACTERS.  THE ASCII CHARACTERS
                ;              ARE THE ONES REPRESENTING THE 8 BITS.  THESE TWO
                ;              CHARACTERS ARE SENT TO THE CONSOLE AT THE CURRENT PRINT
                ;              POSITION OF THE CONSOLE.
                ;
                NMOUT:
0487 F5                 PUSH    PSW     ; SAVE ARGUMENT
0488 0F                 RRC
0489 0F                 RRC
048A 0F                 RRC
048B 0F                 RRC             ; GET UPPER 4 BITS TO LOW 4 BIT POSITIONS
048C CD0D05             CALL    PRVAL   ;CONVERT LOWER 4 BITS TO ASCII
048F CD2103             CALL    ECHO    ; SEND TO TERMINAL
0492 F1                 POP     PSW     ; GET BACK ARGUMENT
0493 CD0D05             CALL    PRVAL
0496 CD2103             CALL    ECHO
0499 C9                 RET
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION HXTIN
                ; INPUTS: NONE
                ; OUTPUTS: NONE
                ; CALLS: ECHO,NMOUT,CROUT
```

```
                ; DESTROYS: A,F/F'S,C,D,H,L
                ; DESCRIPTION: NXTIN PRINTS 3 BYTES OF NEXT INSTRUCTION ON THE CONSOLE
                ;
                ;
                NXTIN:
049A 0E4E               MVI     C,'N'   ; OUTPUT 'NI='
049C CD2103             CALL    ECHO
049F 0E49               MVI     C,'I'
04A1 CD2103             CALL    ECHO
04A4 0E3D               MVI     C,'='
04A6 CD2103             CALL    ECHO
04A9 1603               MVI     D,3     ; OUTPUT 3 BYTES
04AB 2AD83F             LHLD    PSAVE   ; GET LAST PC
04AE 7E         NXTBT:  MOV     A,M
04AF CD8704             CALL    HMOUT   ; OUTPUT BYTE
04B2 0E20               MVI     C,' '   ; USE SPACE FOR DELIMITER
04B4 CD2103             CALL    ECHO
04B7 15                 DCR     D       ; DECREMENT COUNT
04B8 23                 INX     H       ; INCREMENT PC ADDRESS
04B9 C2AE04             JNZ     NXTBT   ; DO NEXT BYTE
04BC CD1203             CALL    CROUT
04BF C9                 RET             ; RETURN
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION PADR
                ; INPUTS: HL - ADDRESS TO BE PUNCHED
                ; OUTPUTS: NONE
                ; CALLS: PBYTE
                ; DESTROYS: A
                ; DESCRIPTION: PADR PUNCHES ON THE TELETYPEWRITER THE ADDRESS
                ;               CONTAINED IN THE H,L REGISTERS.
                ;
                PADR:
04C0 7C                 MOV     A,H     ; PUNCH FIRST HALF OF ADDRESS
04C1 CDC904             CALL    PBYTE
04C4 7D                 MOV     A,L     ; PUNCH SECOND HALF OF ADDRESS
04C5 CDC904             CALL    PBYTE
04C8 C9                 RET             ; RETURN TO CALLING ROUTINE
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION PBYTE
                ; INPUTS: A - CHARACTER TO BE PUNCHED
                ;         D - CURRENT VALUE OF CHECKSUM
                ; OUTPUTS: D - UPDATED VALUE OF CHECKSUM
```

```
                ; CALLS: PRVAL,PO
                ; DESTROYS: A,F/F'S
                ; DESCRIPTION: PBYTE CONVERTS THE HEXADECIMAL VALUE IN THE A REGISTER
                ;              INTO TWO ASCII CHARACTERS AND PUNCHES THESE CHARACTERS
                ;              ON PAPER TAPE. THE CHECKSUM CONTAINED IN D IS UPDATED.
                ;
                PBYTE:
04C9 F5                 PUSH    PSW     ; SAVE A,F/F'S
04CA 0F                 RRC             ; POSITION UPPER 4 BITS INTO LOWER 4 BITS
04CB 0F                 RRC
04CC 0F                 RRC
04CD 0F                 RRC
04CE CD0D05             CALL    PRVAL   ; CONVERT UPPER 4 BITS JUST ROTATED TO ASCII
04D1 CD0905             CALL    PO      ; PUNCH CHARACTER
04D4 F1                 POP     PSW     ; RESTORE A,F/F'S
04D5 F5                 PUSH    PSW     ; SAVE A AGAIN
04D6 CD0D05             CALL    PRVAL   ; CONVERT LOWER 4 BITS TO ASCII CHARACTER
04D9 CD0905             CALL    PO      ; PUNCH CHARACTER
04DC F1                 POP     PSW     ; RESTORE A
04DD 82                 ADD     D       ; ADD VALUE TO CHECKSUM
04DE 57                 MOV     D,A     ; UPDATE D REGISTER WITH NEW CHECKSUM
04DF C9                 RET             ; RETURN TO CALLING ROUTINE
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION PEOF
                ; INPUTS: NONE
                ; OUTPUTS: NONE
                ; CALLS: PO,PBYTE,PADR,LEAD
                ; DESTROYS: A,C,D,H,L,F/F'S
                ; DESCRIPTION: PEOF PUNCHES THE END OF FILE RECORD CONSISTING OF A RECORD
                ;              MARK, A LOAD ADDRESS OF 0, THE RECORD TYPE, AND THE
                ;              RECORD CHECKSUM.
                ;
                PEOF:
04E0 0E3A               MVI     C,':'
04E2 CD0905             CALL    PO      ; PUNCH RECORD MARK
04E5 AF                 XRA     A       ; ZERO CHECKSUM
04E6 57                 MOV     D,A     ; SAVE IN D REGISTER
04E7 CDC904             CALL    PBYTE   ; PUNCH RECORD LENGTH
04EA 210000             LXI     H,0     ; LOAD HL WITH ZERO ADDRESS
04ED CDC004             CALL    PADR    ; PUNCH IT
04F0 3E01               MVI     A,1     ; LOAD A WITH RECORD TYPE
04F2 CDC904             CALL    PBYTE   ; PUNCH IT
04F5 AF                 XRA     A       ; ZERO A
04F6 92                 SUB     D       ; COMPUTE CHECKSUM
04F7 CDC904             CALL    PBYTE   ; PUNCH IT
04FA CD7804             CALL    LEAD    ; PUNCH TRAILER
```

```
04FD C9                 RET
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ;
                        ; FUNCTION PEOL
                        ; INPUTS: NONE
                        ; OUTPUTS: NONE
                        ; CALLS: PO
                        ; DESTROYS: C
                        ; DESCRIPTION: PEOL PUNCHES A CARRIAGE RETURN AND LINE FEED ONTO
                        ;               PAPER TAPE.
                        ;
                        PEOL:
04FE 0E0D               MVI     C,CR
0500 CD0905             CALL    PO      ; PUNCH CARRIAGE RETURN CHARACTER
0503 0E0A               MVI     C,LF
0505 CD0905             CALL    PO      ; PUNCH LINE FEED CHARACTER
0508 C9                 RET
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ;
                        ; FUNCTION PO
                        ; INPUTS: C - CHARACTER TO BE PUNCHED
                        ; OUTPUTS: NONE
                        ; CALLS: CO
                        ; DESTROYS: NOTHING
                        ; DESCRIPTION: PO PUNCHES THE CHARACTER SUPPLIED IN THE C REGISTER TO
                        ;               THE USER TELETYPEWRITER.
                        ;
                        PO:
0509 CD0703             CALL    CO      ; CALL CONSOLE OUT TO PERFORM CHARACTER OUTPUT
050C C9                 RET
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ;
                        ; FUNCTION:  PRVAL
                        ; INPUTS: A - INTEGER, RANGE 0 TO F
                        ; OUTPUTS: A - ASCII CHARACTER
                        ; CALLS: NOTHING
                        ; DESTROYS: NOTHING
                        ; DESCRIPTION: PRVAL CONVERTS A NUMBER IN THE RANGE 0 TO F HEX TO
                        ;               THE CORRESPONDING ASCII CHARACTER, 0-9,A-F.  PRVAL
                        ;               DOES NOT CHECK THE VALIDITY OF ITS INPUT ARGUMENT.
                        ;
```

```
                PRVAL:
0500 E60F               ANI     HCHAR   ; MASK OUT UPPER 4 BITS - WANT 1 HEX CHAR
050F C690               ADI     90H     ; SET UP A SO THAT A-F CAUSE A CARRY
0511 27                 DAA             ; ADJUST CONTENTS OF A REGISTER
0512 CE40               ACI     40H     ; ADD IN CARRY AND ADJUST UPPER 4 BITS
0514 27                 DAA             ; ADJUST CONTENTS OF A REGISTER AGAIN
0515 4F                 MOV     C,A     ; MOVE ASCII CHARACTER TO C
0516 C9                 RET             ; ALL DONE
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION: REGDS
                ; INPUTS: NONE
                ; OUTPUTS: NONE
                ; CALLS: ECHO,HMOUT,ERROR,CROUT
                ; DESTROYS: A,B,C,D,E,H,L,F/F'S
                ; DESCRIPTION: REGDS DISPLAYS THE CONTENTS OF THE REGISTER SAVE
                ;               LOCATIONS, IN FORMATTED FORM, ON THE CONSOLE.  THE
                ;               DISPLAY IS DRIVEN FROM A TABLE, RTAB, WHICH CONTAINS
                ;               THE REGISTER'S PRINT SYMBOL, SAVE LOCATION ADDRESS,
                ;               AND LENGTH (8 OR 16 BITS).
                ;
                REGDS:
0517 216D06             LXI     H,RTAB  ; LOAD HL WITH ADDRESS OF START OF TABLE
                REG05:
051A 4E                 MOV     C,M     ; GET PRINT SYMBOL OF REGISTER
051B 79                 MOV     A,C
051C B7                 ORA     A       ; TEST FOR 0 - END OF TABLE
051D C22405             JNZ     REG10   ; IF NOT END, BRANCH
0520 CD1203             CALL    CROUT   ; ELSE, CARRIAGE RETURN/LINE FEED TO END
0523 C9                 RET             ; /DISPLAY
                REG10:
0524 CD2103             CALL    ECHO    ; ECHO CHARACTER
0527 0E3D               MVI     C,'='
0529 CD2103             CALL    ECHO    ; OUTPUT EQUALS SIGN, I.E. A=
052C 23                 INX     H       ; POINT TO START OF SAVE LOCATION ADDRESS
052D 5E                 MOV     E,M     ; GET LSP OF SAVE LOCATION ADDRESS TO E
052E 163F               MVI     D,REGS SHR 8    ; PUT MSP OF SAVE LOC ADDRESS INTO D
0530 23                 INX     H       ; POINT TO LENGTH FLAG
0531 1A                 LDAX    D       ; GET CONTENTS OF SAVE ADDRESS
0532 CD8704             CALL    HMOUT   ; DISPLAY ON CONSOLE
0535 7E                 MOV     A,M     ; GET LENGTH FLAG
0536 B7                 ORA     A       ; SET SIGN F/F
0537 CA3F05             JZ      REG15   ; IF 0, REGISTER IS 8 BITS
053A 1B                 DCX     D       ; ELSE, 16 BIT REGISTER SO MORE TO DISPLAY
053B 1A                 LDAX    D       ; GET LOWER 8 BITS
053C CD8704             CALL    HMOUT   ; DISPLAY THEM
                REG15:
```

```
053F 0E20               MVI     C,' '
0541 CD2103             CALL    ECHO    ; OUTPUT BLANK CHARACTER
0544 23                 INX     H       ; POINT TO START OF NEXT TABLE ENTRY
0545 C31A05             JMP     REG05   ; DO NEXT REGISTER
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ;
                        ; FUNCTION REGSV
                        ; INPUTS: NONE
                        ; OUTPUTS: NONE
                        ; CALLS: NONE
                        ; DESTROYS: H,SP
                        ; DESCRIPTION: REGSV SAVES THE USER REGISTERS ON INTERRUPT.
                        ;
                        REGSV:
0548 22D63F             SHLD    LSAVE   ; SAVE HL REGISTERS
054B E1                 POP     H       ; GET CALLING ADDRESS
054C E3                 XTHL            ; EXCHANGE CALLER ADDR. WITH INT. PC
054D 22DB3F             SHLD    PSAVE   ; ASSUME THIS IS THE LAST PROG COUNTER
0550 F5                 PUSH    PSW     ; SAVE A,F/F'S
0551 210400             LXI     H,4     ; SET HL TO 4 TO SAVE STACK POINTER CORRECTLY
0554 39                 DAD     SP      ; GET STACK POINTER VALUE
0555 22DA3F             SHLD    SSAVE   ; SAVE USERS STACK POINTER
0558 F1                 POP     PSW     ; RESTORE A,F/F'S
0559 E1                 POP     H       ; CALLERS RETURN POINT
055A 31D63F             LXI     SP,ASAVE+1      ; NEW VALUE FOR STACK POINTER
055D F5                 PUSH    PSW     ; SAVE THE REST OF THE REGISTERS
055E C5                 PUSH    B
055F D5                 PUSH    D
0560 E9                 PCHL            ; RETURN
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ;
                        ; FUNCTION: RGADR
                        ; INPUTS: C - CHARACTER DENOTING REGISTER
                        ; OUTPUTS: BC - ADDRESS OF ENTRY IN RTAB CORRESPONDING TO REGISTER
                        ; CALLS: ERROR
                        ; DESTROYS: A,B,C,D,E,H,L,F/F'S
                        ; DESCRIPTION: RGADR TAKES A SINGLE CHARACTER AS INPUT.  THIS CHARACTER
                        ;               DENOTES A REGISTER.  RGADR SEARCHES THE TABLE RTAB
                        ;               FOR A MATCH ON THE INPUT ARGUMENT.  IF ONE OCCURS,
                        ;               RGADR RETURNS THE ADDRESS OF THE ADDRESS OF THE
                        ;               SAVE LOCATION CORRESPONDING TO THE REGISTER.  THIS
                        ;               ADDRESS POINTS INTO RTAB.  IF NO MATCH OCCURS, THEN
                        ;               THE REGISTER IDENTIFIER IS ILLEGAL AND CONTROL IS
                        ;               PASSED TO THE ERROR ROUTINE.
```

```
                ;
                RGADR:
0561 216D06             LXI     H,RTAB  ; HL GETS ADDRESS OF TABLE START
0564 110300             LXI     D,RTABS ; DE GET SIZE OF A TABLE ENTRY
                RGA05:
0567 7E                 MOV     A,M     ; GET REGISTER IDENTIFIER
0568 B7                 ORA     A       ; CHECK FOR TABLE END (IDENTIFIER IS 0)
0569 CA3A03             JZ      ERROR   ; IF AT END OF TABLE, ARGUMENT IS ILLEGAL
056C B9                 CMP     C       ; ELSE, COMPARE TABLE ENTRY AND ARGUMENT..
056D CA7405             JZ      RGA10   ; IF EQUAL, WE'VE FOUND WHAT WE'RE LOOKING FOR
0570 19                 DAD     D       ; ELSE, INCREMENT TABLE POINTER TO NEXT ENTRY
0571 C36705             JMP     RGA05   ; TRY AGAIN
                RGA10:
0574 23                 INX     H       ; IF A MATCH, INCREMENT TABLE POINTER TO
0575 44                 MOV     B,H     ; /SAVE LOCATION ADDRESS
0576 4D                 MOV     C,L     ; RETURN THIS VALUE
0577 C9                 RET
                ;
                ;
                ;*****************************************************
                ;
                ;
                ; FUNCTION RI
                ; INPUTS: NONE
                ; OUTPUTS: A - ZERO, CARRY -> 1 IF END OF FILE
                ;          A - CHARACTER, CARRY -> 0 IF VALID CHARACTER
                ; CALLS: DELAY
                ; DESTROYS: A,F/F'S
                ; DESCRIPTION: RI READS A CHARACTER FROM THE TTY TAPE READER.
                ;
                RI:
0578 C5                 PUSH    B       ; SAVE BC
                RI05:
0579 D8ED               IN      CNCTL   ; READ IN USART STATUS
057B E604               ANI     TXBE    ; CHECK FOR TRANSMITTER BUFFER EMPTY
057D CA7905             JZ      RI05    ; TRY AGAIN IF NOT EMPTY
0580 3E27               MVI     A,TTYADV        ; ADVANCE THE TAPE
0582 D3ED               OUT     CNCTL   ; OUTPUT THE ADVANCE COMMAND
0584 0628               MVI     B,40    ; INITIALIZE TIMER FOR 40 MS
                RI07:
0586 CD1803             CALL    DELAY   ; DELAY FOR 1 MILLISECOND
0589 05                 DCR     B       ; DECREMENT TIMER
058A C28605             JNZ     RI07    ; JUMP IF TIMER NOT EXPIRED
058D 3E25               MVI     A,TTYSTP    ; STOP THE READER ADVANCE
058F D3ED               OUT     CNCTL   ; OUTPUT STOP COMMAND
0591 06FA               MVI     B,250     ; INITIALIZE TIMER FOR 250 MS.
                RI10:
0593 D8ED               IN      CONST   ; INPUT READER STATUS
0595 E602               ANI     RBR     ; CHECK FOR RECEIVER BUFFER READY
0597 C2A505             JNZ     RI15    ; YES - DATA IS READY
```

```
059A CD1803          CALL    DELAY   ; DELAY 1 MS
059D 05              DCR     B       ; DECREMENT TIMER
059E C29305          JNZ     RI10    ; JUMP IF TIMER NOT EXPIRED
05A1 AF              XRA     A       ; ZERO A
05A2 37              STC             ; SET CARRY INDICATING TIMEOUT ERROR
05A3 C1              POP     B       ; RESTORE BC
05A4 C9              RET             ; RETURN TO CALLING ROUTINE
                RI15:
05A5 DBEC            IN      CHIN    ; INPUT DATA CHARACTER
05A7 B7              ORA     A       ; CLEAR CARRY
05A8 C1              POP     B       ; RESTORE BC
05A9 C9              RET             ; RETURN TO CALLING ROUTINE
                ;
                ;*****************************************************************
                ;
                ; FUNCTION RICH
                ; INPUTS: NONE
                ; OUTPUTS: A - ZERO, CARRY - 1 IF END OF FILE
                ;          A - CHARACTER, CARRY - 0 IF VALID CHARACTER
                ; CALLS: RI
                ; DESTROYS: A,F/F'S
                ; DESCRIPTION: RICH TESTS FOR A TIMEOUT ERROR CONDITION
                ;
                RICH:
05AA CD7805          CALL    RI      ; READ A CHARACTER FROM TAPE
05AD DA3A03          JC      ERROR   ; JUMP IF READER TIMEOUT ERROR
0580 E67F            ANI     PRTYO   ; REMOVE PARITY BIT
0582 C9              RET             ; RETURN TO CALLING ROUTINE
                ;
                ;*****************************************************************
                ;
                ; FUNCTION: RSTTF
                ; INPUTS: NONE
                ; OUTPUTS: NONE
                ; CALLS: NOTHING
                ; DESTROYS: A,B,C,D,E,H,L,F/F'S
                ; DESCRIPTION: RSTTF RESTORES ALL CPU REGISTER, FLIP/FLOPS, STACK
                ;              POINTER AND PROGRAM COUNTER FROM THEIR RESPECTIVE
                ;              SAVE LOCATIONS IN MEMORY.  THE ROUTINE THEN TRANSFERS
                ;              CONTROL TO THE LOCATION SPECIFIED BY THE PROGRAM
                ;              COUNTER (I.E. THE RESTORED VALUE).  THE ROUTINE
                ;              EXITS WITH THE INTERRUPTS ENABLED.
                ;
                RSTTF:
0583 F3              DI              ; DISABLE INTERRUPTS WHILE RESTORING THINGS
0584 F5              PUSH    PSW     ; SAVE A REGISTER
```

```
0585 3E30               MVI     A,COMO  ; RESET SINGLE STEP TIMER
0587 D3DF               OUT     TMCP
0589 F1                 POP     PSW     ; RESTORE A REGISTER
058A 31D03F             LXI     SP,MSTAK        ; SET MONITOR STACK POINTER TO START
                                        ; /OF STACK
058D D1                 POP     D       ; START ALSO END OF REGISTER SAVE AREA
058E C1                 POP     B
058F 2AD43F             LHLD    SSAVE   ; RESTORE USER STACK POINTER
05C2 F9                 SPHL
05C3 2AD43F             LHLD    FSAVE   ; GET A,F/F'S
05C6 E5                 PUSH    H       ; SAVE THEM
05C7 A7                 ANA     A       ; CHECK FOR SINGLE STEP
05C8 CAD305             JZ      RSTDN   ; NO, DONE
05CB 3E20               MVI     A,LSVCO ; YES, LOAD TIMER
05CD D3DC               OUT     CTRO
05CF 3E00               MVI     A,MSVCO
05D1 D3DC               OUT     CTRO
05D3 F1         RSTDN:  POP     PSW     ; RESTORE A,F/F'S
05D4 2AD83F             LHLD    PSAVE
05D7 E5                 PUSH    H       ; PUT USER RETURN ADDRESS ON USER STACK
05D8 2AD63F             LHLD    LSAVE   ; RESTORE HL REGISTERS
05DB FB                 EI              ; ENABLE INTERRUPTS NOW
05DC C9                 RET             ; JUMP TO RESTORED PC LOCATION
                ;
                ;
                ;*****************************************************************
                ;
                ; FUNCTION: SRET
                ; INPUTS: NONE
                ; OUTPUTS: CARRY = 1
                ; CALLS: NOTHING
                ; DESTROYS: CARRY
                ; DESCRIPTION: SRET IS JUMPED TO BY ROUTINES WISHING TO RETURN SUCCESS.
                ;               SRET SETS THE CARRY TRUE AND THEN RETURNS TO THE
                ;               CALLER OF THE ROUTINE INVOKING SRET.
                ;
                SRET:
05DD 37                 STC             ; SET CARRY TRUE
05DE C9                 RET             ; RETURN APPROPRIATELY
                ;
                ;
                ;*****************************************************************
                ;
                ; FUNCTION: STHFO
                ; INPUTS: DE - 16 BIT ADDRESS OF BYTE TO BE STORED INTO
                ; OUTPUTS: NONE
                ; CALLS: NOTHING
                ; DESTROYS: A,B,C,H,L,F/F'S
```

```
                ; DESCRIPTION: STHF0 CHECKS THE HALF BYTE FLAG IN TEMP TO SEE IF
                ;              IT IS SET TO LOWER.  IF SO, STHF0 STORES A 0 TO
                ;              PAD OUT THE LOWER HALF OF THE ADDRESSED BYTE;
                ;              OTHERWISE, THE ROUTINE TAKES NO ACTION.
                ;
                STHF0:
05DF 3ADC3F             LDA     TEMP    ; GET HALF BYTE FLAG
05E2 B7                 ORA     A       ; SET F/F'S
05E3 C0                 RNZ             ; IF SET TO UPPER, DON'T DO ANYTHING
05E4 0E00               MVI     C,0     ; ELSE, WANT TO STORE THE VALUE 0
05E6 CDEA05             CALL    STHLF   ; DO IT
05E9 C9                 RET
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION: STHLF
                ; INPUTS: C - 4 BIT VALUE TO BE STORED IN HALF BYTE
                ;         DE - 16 BIT ADDRESS OF BYTE TO BE STORED INTO
                ; OUTPUTS: NONE
                ; CALLS: NOTHING
                ; DESTROYS: A,B,C,H,L,F/F'S
                ; DESCRIPTION: STHLF TAKES THE 4 BIT VALUE IN C AND STORES IT IN
                ;              HALF OF THE BYTE ADDRESSED BY REGISTERS DE.  THE
                ;              HALF BYTE USED (EITHER UPPER OR LOWER) IS DENOTED
                ;              BY THE VALUE OF THE FLAG IN TEMP.  STHLF ASSUMES
                ;              THAT THIS FLAG HAS BEEN PREVIOUSLY SET
                ;              (NOMINALLY BY ICMD).
                ;
                STHLF:
05EA D5                 PUSH    D
05EB E1                 POP     H       ; MOVE ADDRESS OF BYTE INTO HL
05EC 79                 MOV     A,C     ; GET VALUE
05ED E60F               ANI     0FH     ; FORCE TO 4 BIT LENGTH
05EF 4F                 MOV     C,A     ; PUT VALUE BACK
05F0 3ADC3F             LDA     TEMP    ; GET HALF BYTE FLAG
05F3 B7                 ORA     A       ; CHECK FOR LOWER HALF
05F4 C2FD05             JNZ     STH05   ; BRANCH IF NOT
05F7 7E                 MOV     A,M     ; ELSE, GET BYTE
05F8 E6F0               ANI     0F0H    ; CLEAR LOWER 4 BITS
05FA B1                 ORA     C       ; OR IN VALUE
05FB 77                 MOV     M,A     ; PUT BYTE BACK
05FC C9                 RET
                STH05:
05FD 7E                 MOV     A,M     ; IF UPPER HALF, GET BYTE
05FE E60F               ANI     0FH     ; CLEAR UPPER 4 BITS
0600 47                 MOV     B,A     ; SAVE BYTE IN B
0601 79                 MOV     A,C     ; GET VALUE
0602 0F                 RRC
```

```
0603 0F                 RRC
0604 0F                 RRC
0605 0F                 RRC             ; ALIGH TO UPPER 4 BITS
0606 B0                 ORA     B       ; OR IN ORIGINAL LOWER 4 BITS
0607 77                 MOV     M,A     ; PUT NEW CONFIGURATION BACK
0608 C9                 RET
                )
                ;
                ;*****************************************************************
                ;
                ;
                ; FUNCTION: VALDG
                ; INPUTS: C - ASCII CHARACTER
                ; OUTPUTS: CARRY - 1 IF CHARACTER REPRESENTS VALID HEX DIGIT
                ;                - 0 OTHERWISE
                ; CALLS: NOTHING
                ; DESTROYS: A,F/F'S
                ; DESCRIPTION: VALDG RETURNS SUCCESS IF ITS INPUT ARGUMENT IS
                ;               AN ASCII CHARACTER REPRESENTING A VALID HEX DIGIT
                ;               (0-9,A-F), AND FAILURE OTHERWISE.
                ;
                VALDG:
0609 79                 MOV     A,C
060A FE30               CPI     '0'     ; TEST CHARACTER AGAINST '0'
060C FA4503             JM      FRET    ; IF ASCII CODE LESS, CANNOT BE VALID DIGIT
060F FE39               CPI     '9'     ; ELSE, SEE IF IN RANGE '0'-'9'
0611 FADD05             JM      SRET    ; CODE BETWEEN '0' AND '9'
0614 CADD05             JZ      SRET    ; CODE EQUAL '9'
0617 FE41               CPI     'A'     ; NOT A DIGIT - TRY FOR A LETTER
0619 FA4503             JM      FRET    ; NO - CODE BETWEEN '9' AND 'A'
061C FE47               CPI     'G'
061E F24503             JP      FRET    ; NO - CODE GREATER THAN 'F'
0621 C3DD05             JMP     SRET    ; OKAY - CODE IS 'A' TO 'F', INCLUSIVE
                )
                )
                ;*****************************************************************
                ;
                ;
                ; FUNCTION: VALDL
                ; INPUTS: C - CHARACTER
                ; OUTPUTS: CARRY - 1 IF INPUT ARGUMENT VALID DELIMTER
                ;                - 0 OTHERWISE
                ; CALLS: NOTHING
                ; DESTROYS: A,F/F'S
                ; DESCRIPTION: VALDL RETURNS SUCCESS IF ITS INPUT ARGUMENT IS A VALID
                ;               DELIMITER CHARACTER (SPACE, COMMA, CARRIAGE RETURN) AND
                ;               FAILURE OTHERWISE.
                ;
                VALDL:
0624 79                 MOV     A,C
```

```
0625 FE2C               CPI     ','     ; CHECK FOR COMMA
0627 CA0D05             JZ      SRET
062A FE0D               CPI     CR      ; CHECK FOR CARRIAGE RETURN
062C CA0D05             JZ      SRET
062F FE20               CPI     ' '     ; CHECK FOR SPACE
0631 CA0D05             JZ      SRET
0634 C34503             JMP     FRET    ; ERROR IF NONE OF THE ABOVE
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ;               MONITOR TABLES
                        ;
                        ;*****************************************************************
                        ;
                        ;
                        SGNON:                  ; SIGNON MESSAGE
0637 000A3830           DB      CR,LF,'80/20 MONITOR   V 1.2',CR,LF
063B 2F323020
063F 4D4F4E49
0643 544F5220
0647 20205620
064B 312E3200
064F 0A
0819                    LSGNON  EQU     $-SGNON ; LENGTH OF SIGNON MESSAGE
                        ;
                        CADR:                   ; TABLE OF ADDRESSES OF COMMAND ROUTINES
0650 0000                       DW      0       ; DUMMY
0652 1801                       DW      MCMD
0654 D401                       DW      XCMD
0656 5B01                       DW      SCMD
0658 F800                       DW      MCMD
065A 8200                       DW      ICMD
065C 9300                       DW      GCMD
065E C700                       DW      DCMD
0660 2001                       DW      RCMD
0662 7F01                       DW      WCMD
                        ;
                        CTAB:                   ; TABLE OF VALID COMMAND CHARACTERS
0664 57                         DB      'W'
0665 52                         DB      'R'
0666 44                         DB      'D'
0667 47                         DB      'G'
0668 49                         DB      'I'
0669 4D                         DB      'M'
066A 53                         DB      'S'
066B 58                         DB      'X'
066C 4E                         DB      'N'
```

```
0009            NCMDS   EQU     $-CTAB  ; NUMBER OF VALID COMMANDS
                ;
                ;
                RTAB:                   ; TABLE OF REGISTER INFORMATION
066D 41                 DB      'A'     ; REGISTER IDENTIFIER
066E D5                 DB      ASAVE AND OFFH  ; ADDRESS OF REGISTER SAVE LOCATION
066F 00                 DB      0       ; LENGTH FLAG - 0=8 BITS, 1=16 BITS
0003            RTABS   EQU     $-RTAB  ; SIZE OF AN ENTRY IN THIS TABLE
0670 42                 DB      'B'
0671 D3                 DB      BSAVE AND OFFH
0672 00                 DB      0
0673 43                 DB      'C'
0674 D2                 DB      CSAVE AND OFFH
0675 00                 DB      0
0676 44                 DB      'D'
0677 D1                 DB      DSAVE AND OFFH
0678 00                 DB      0
0679 45                 DB      'E'
067A D0                 DB      ESAVE AND OFFH
067B 00                 DB      0
067C 46                 DB      'F'
067D D4                 DB      FSAVE AND OFFH
067E 00                 DB      0
067F 48                 DB      'H'
0680 D7                 DB      HSAVE AND OFFH
0681 00                 DB      0
0682 4C                 DB      'L'
0683 D6                 DB      LSAVE AND OFFH
0684 00                 DB      0
0685 4D                 DB      'M'
0686 D7                 DB      HSAVE AND OFFH
0687 01                 DB      1
0688 50                 DB      'P'
0689 D9                 DB      PSAVE+1 AND OFFH
068A 01                 DB      1
068B 53                 DB      'S'
068C D8                 DB      SSAVE+1 AND OFFH
068D 01                 DB      1
068E 00                 DB      0       ; END OF TABLE MARKERS
068F 00                 DB      0
                ;
                ;
                ;*****************************************************************
                ;
                ;
                ;
                ; THE FOLLOWING JUMP TABLE IS TO BE USED WITH THE
                ; INTERRUPT CONTROLLER. THE CONTROLLER HAS BEEN INITIALIZED
                ; TO CALL THIS TABLE WHEN AN INTERRUPT IS ACKNOWLEDGED.
                ; THIS TABLE IS MOVED TO RAM DURING MONITOR INITIALIZATION.
```

```
                        ; THE JUMP SHOULD BE TO AN INTERRUPT SERVICE ROUTINE.
                        ;
                        ;
0690 C34902     JPTB:   JMP     INTIN   ; JUMP TO SERVICE ROUTINE FOR LEVEL 0
0693 00                 NOP             ; FILLER
0694 C34902             JMP     INTIN   ; --- FOR LEVEL 1
0697 00                 NOP             ; FILLER
0698 C37F02             JMP     STEPIN  ; --- FOR LEVEL 2
069B 00                 NOP             ; FILLER
069C C34902             JMP     INTIN   ; --- FOR LEVEL 3
069F 00                 NOP             ; FILLER
06A0 C34902             JMP     INTIN   ; --- FOR LEVEL 4
06A3 00                 NOP             ; FILLER
06A4 C34902             JMP     INTIN   ; --- FOR LEVEL 5
06A7 00                 NOP             ; FILLER
06A8 C34902             JMP     INTIN   ; --- FOR LEVEL 6
06AB 00                 NOP             ; FILLER
06AC C34902             JMP     INTIN   ; --- FOR LEVEL 7
                        ;
                        ;
                        ;*****************************************************************
                        ;
                        ;
3F00                    ORG     REGS    ; ORG TO REGISTER SAVE - STACK GOES IN HERE
                        ;
3F00            MSTAK   EQU     $       ; START OF MONITOR STACK
3F00 00         ESAVE:  DB      0       ; E REGISTER SAVE LOCATION
3F01 00         DSAVE:  DB      0       ; D REGISTER SAVE LOCATION
3F02 00         CSAVE:  DB      0       ; C REGISTER SAVE LOCATION
3F03 00         BSAVE:  DB      0       ; B REGISTER SAVE LOCATION
3F04 00         FSAVE:  DB      0       ; FLAGS SAVE LOCATION
3F05 00         ASAVE:  DB      0       ; A REGISTER SAVE LOCATION
3F06 00         LSAVE:  DB      0       ; L REGISTER SAVE LOCATION
3F07 00         HSAVE:  DB      0       ; H REGISTER SAVE LOCATION
3F08 0000       PSAVE:  DW      0       ; PGM COUNTER SAVE LOCATION
3F0A 0000       SSAVE:  DW      0       ; USER STACK POINTER SAVE LOCATION
3F0C 00         TEMP:   DB      0       ; TEMPORARY MONITOR CELL
                        ;
                        ;
0000                    END
```

```
ADRD  02AF    ADROU 02B8    ASAVE 3FD5    B0110 0263
B9600 0007    BRCHR 001B    BREAK 02C6    BRS05 03F4
BRS06 03F8    BRS07 0401    BRS08 0414    BRS10 0429
BRS15 042B    BRS20 0444    BRSEL 03EF    BRTAB 03FA
BSAVE 3FD3    BYTE  02D9    COMD  0030    C2N3  00B6
CADR  0650    CHARR 0055    CI    02F4    CND   0027
CNCTL 00ED    CNIN  00EC    CNOUT 00EC    CNVBN 02FE
CO    0307    CONST 00ED    CPYRT 001B    CR    000D
CROUT 0312    CSAVE 3FD2    CTAB  0664    CTR0  00DC
CTR1  00DD    CTR2  00DE    DATA  4000    DCM05 006E
DCM10 0074    DCND  0067    DEL1  031B    DELAY 0318
DSAVE 3FD1    ECH05 032A    ECH10 0338    ECHO  0321
EDIC  0020    ERROR 033A    ESAVE 3FD0    ESC   001B
EXIT  033F    FALSE F785    FINTN 026D    FNDI  0269
FRET  0345    FSAVE 3FD4    GCM05 0048    GCM10 00AE
GCND  0093    GETCH 034C    GETCM 003C    GETHX 034F
GETNM 0383    GHX05 0355    GHX10 036D    GNM05 036A
GNM10 039F    GNM15 03AD    GNM20 0382    GNM25 03BD
GNM30 03C1    GO    0008    GTC05 0051    GTC10 005D
HCHAR 000F    HILO5 0307    HILO  03C8    HSAVE 3FD7
ICCP  00DA    ICM05 00BD    ICM10 00E4    ICM20 00EC
ICM25 00F2    ICND  00B2    ICW1  00F6    ICW2  003F
IICR  0456    IMASK 0000    INTIN 0249    INUST 03DA
INVRT 00FF    JPTB  0690    JTBL  3FE0    LE05  047D
LGND  047B    LF    000A    LSAVE 3FD6    LSGN0 0019
LSVC0 0020    MCM05 0100    MCMD  00F8    MODE  004E
MSGL  0033    MSKPT 00DB    MSTAK 3FD0    MSVC0 0000
MTBL  046A    NCMD  0118    NCMDS 0009    NEWLN 000F
NNOUT 0487    NXTBT 04AE    NXTIN 049A    OCW3  000B
OHEMS 008B    PADR  04C0    PBYTE 04C9    PEOF  04E0
PEOL  04FE    PO    0509    PRTY0 007F    PRVAL 050D
PSAVE 3FD8    RBR   0002    RCM05 012C    RCM10 0149
RCND  0120    REG05 051A    REG10 0524    REG15 053F
REGOS 0517    REGS  3FD0    REGSV 0548    RGA05 0567
RGA10 0574    RGADR 0561    RI    0578    RIOS  0579
RI07  0586    RI10  0593    RI15  05A5    RICH  05AA
RSTDN 0503    RSTTF 05B3    RSTUS 0040    RTAB  066D
RTABS 0003    SCM05 0160    SCM10 016B    SCM15 017B
SCRD  015D    SGNON 0637    SOMSG 002E    SRET  050D
SSAVE 3FDA    STEPI 027F    STH05 05FD    STHF0 05DF
STHLF 05EA    STPQK 02A6    TEMP  3FDC    TERM  001B
TMCP  00DF    TRDY  0001    TRUE  F7A2    TTYAD 0027
TTYST 0025    TXBE  0004    UPPER 00FF    USARE 3F80
VALDG 0609    VALDL 0624    WCM05 0189    WCM10 019E
WCM15 01A1    WCM20 01B9    WCM25 01CE    WCMD  017F
XCM05 01E6    XCM10 01F5    XCM15 0202    XCM20 0220
XCM25 0237    XCM27 0238    XCM30 0240    XCMD  01D4
```