INTELLEC® SERIES II MICROCOMPUTER DEVELOPMENT SYSTEM BOOT/MONITOR LISTING

Manual Order Number: 9800605-02 Rev. B

A372/581/4K DD

PREFACE

This manual contains the source listing for version 1.3 of the ROM-resident boot/monitor program for the Intellec Series II Microcomputer Development System. The manual also includes version 2.1 of the assembler symbol cross reference associated with the boot/monitor program and the source listing for the IPB/IPC ROM-resident diagnostic.

```
 LOC  OBJ         LINE        SOURCE STATEMENT

                  
                  ;************************************************************************
                  ;*                                                                      *
                  ;*                        INTELLEC SERIES II BOOT/MONITOR               *
                  ;*                                VERSION 1.3                           *
                  ;*                                                                      *
                  ;*      COPYRIGHT (C) 1978, 1979 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,              *
                  ;*      3Ø65 BOWERS AVENUE, SANTA CLARA, CALIFORNIA 95Ø51..             *
                  ;*                                                                      *
                  ;************************************************************************
                  ; <LEGAL COMMAND> ::=   <ASSIGN I/O COMMAND>
                  ;                       <DISPLAY MEMORY COMMAND>
                  ;                       <ENDFILE COMMAND>
                  ;                       <FILL MEMORY COMMAND>
                  ;                       <PROGRAM EXECUTE COMMAND>
                  ;                       <HEXADECIMAL ARITHMETIC COMMAND>
                  ;                       <MOVE MEMORY COMMAND>
                  ;                       <LEADER COMMAND>
                  ;                       <QUERY STATUS COMMAND>
                  ;                       <READ HEXADECIMAL FILE COMMAND>
                  ;                       <SUBSTITUTE MEMORY COMMAND>
                  ;                       <WRITE HEXADECIMAL RECORD COMMAND>
                  ;                       <REGISTER MODIFY COMMAND>
                  ;                       <TRANSFER CONTROL TO DIAGNOSTIC PROGRAM COMMAND>
                  ; <ASSIGN I/O COMMAND> ::= A<LOGICAL DEVICE>=<PHYSICAL DEVICE>
                  ; <DISPLAY MEMORY COMMAND> ::= D<NUMBER>,<NUMBER>
                  ; <ENDFILE COMMAND> ::= E<NUMBER>
                  ; <FILL MEMORY COMMAND> ::= F<NUMBER>,<NUMBER>,<NUMBER>
                  ; <PROGRAM EXECUTE COMMAND> ::= G<NUMBER>,<NUMBER>,<NUMBER>
                  ; <HEXADECIMAL ARITHMETIC COMMAND> ::= H<NUMBER>,<NUMBER>
                  ; <MOVE MEMORY COMMAND> ::= M<NUMBER>,<NUMBER>,<NUMBER>
                  ; <LEADER COMMAND> ::= N
                  ; <QUERY STATUS COMMAND> ::= Q
                  ; <READ HEXADECIMAL FILE COMMAND> ::= R<NUMBER>
                  ; <SUBSTITUTE MEMORY COMMAND> ::= S<NUMBER><COMMA>...
                  ; <WRITE HEXADECIMAL RECORD COMMAND> ::= W<NUMBER>,<NUMBER>
                  ; <REGISTER MODIFY COMMAND> ::= X<REGISTER IDENTIFIER><NUMBER>...
                  ; <TRANSFER CONTROL TO DIAGNOSTIC PROGRAM COMMAND> ::= Z$
                  ; <LOGICAL DEVICE> ::= LOCAL CONSOLE!READER!LIST!PUNCH
                  ; <PHYSICAL DEVICE> ::= CRT!TTY!PTR!PTP!LPT!BATCH!1!2
                  ; <REGISTER IDENTIFIER> ::= A!B!C!D!E!F!H!I!L!M!P!S
                  ; <NUMBER> ::=  <HEX DIGIT>
                  ;               <NUMBER><HEX DIGIT>
                  ; <HEX DIGIT> ::= Ø!1!2!3!4!5!6!7!8!9!A!B!C!D!E!F
                  ;*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*
                  $       TITLE   (' INTELLEC SERIES II MONITOR, VERSION 1.3, 1 MARCH 1979 ')
 ØØØD             VER     EQU     13      ; VERSION 1.3
```

```
LOC  OBJ         LINE        SOURCE STATEMENT

0013                 VERH    EQU     13H     ; STORAGE REPRESENTATION OF VERSION
0103                 DATE    EQU     0103H   ; CREATION DATE, 01 MARCH 1979
                     ; NOTE:
                     ; THE DATE SHOWN ABOVE IS ENCODED IN A TWO BYTE FIELD IN BOTH THE BOOTSTRAP
                     ; PROM AND THE MONITOR PROM IN ORDER TO CONTROL NEW RELEASES OF THIS PROGRAM.
                     ; THE DATE CODE IS LOCATED AT ADDRESSES 0E804H AND 0E805H IN THE BOOTSTRAP
                     ; AND AT ADDRESSES 0F824H AND 0F825H IN THE MONITOR.
                     ; THE VERSION CODE IS LOCATED IN THE MONITOR ROM AT ADDRESS 0F82FH.
                     ; WHEN A NEW RELEASE IS ISSUED, PLEASE CHANGE THE DATE AND VERSION CODES.
                     ; THE COPYRIGHT NOTICE IS LOCATED IN THE MONITOR ROM BEGINNING AT 0F830H.
                     ;***************************************************************************
                     ;*                                                                         *
                     ;*                         SYMBOL DEFINITIONS                              *
                     ;*                                                                         *
                     ;***************************************************************************
                     ;
                     ; INTELLEC SERIES II SYSTEM CONSTANTS
                     ;
                     ; INTEGRATED CONSOLE I/O PORTS
                     ;
00C0                 CONI    EQU     0C0H            ; CONSOLE INPUT DATA PORT
00C0                 CONO    EQU     0C0H            ; CONSOLE OUTPUT DATA PORT
00C1                 CONS    EQU     0C1H            ; CONSOLE STATUS PORT
00C1                 CONC    EQU     0C1H            ; CONSOLE CONTROL PORT
                     ;
                     ; SYSTEM BOOTSTRAP CONSTANTS (ISSUED TO PORT CPUC)
                     ;
000D                 DISABL  EQU     0DH             ; DISABLE INTERRUPTS
0005                 ENABL   EQU     05H             ; ENABLE INTERRUPTS
0000                 DISAXP  EQU     00H             ; DISABLE AUXILIARY PROM

0008                 ENAXP   EQU     08H             ; ENABLE AUXILIARY PROM
0001                 BOVROF  EQU     01H             ; TURN OFF BUS OVERRIDE
0009                 BOVRON  EQU     09H             ; TURN ON BUS OVERRIDE
0004                 BTDGOF  EQU     04H             ; TURN OFF BOOT/DIAGNOSTIC
000C                 BTDGON  EQU     0CH             ; TURN ON BOOT/DIAGNOSTIC
0002                 MOVBOT  EQU     02H             ; MOVE BOOT TO 0E800H
                     ;
                     ; SYSTEM I/O PORTS
                     ;
00FE                 CPUS    EQU     0FEH            ; CPU STATUS PORT
00FF                 CPUC    EQU     0FFH            ; CPU CONTROL PORT (CONTROLS BOOT & AUX.PROM)
                     ;
                     ; SYSTEM INTERRUPT CONSTANTS
                     ;
0012                 ICW1    EQU     00010010B       ; INITIALIZATION COMMAND WORD 1
0000                 ICW2    EQU     00000000B       ; INITIALIZATION COMMAND WORD 2
000B                 OCW3    EQU     00001011B       ; OPERATION COMMAND WORD 3
0020                 EOI     EQU     00100000B       ; END OF INTERRUPT
                     ;
                     ; SYSTEM INTERRUPT MASKS AND VALUES
                     ;
0001                 INT0    EQU     00000001B       ; MASK FOR INTERRUPT LEVEL 0
0002                 INT1    EQU     00000010B
0004                 INT2    EQU     00000100B
```

```
 LOC  OBJ          LINE        SOURCE STATEMENT

 0008               110 INT3    EQU     00001000B
 0010               111 INT4    EQU     00010000B
 0020               112 INT5    EQU     00100000B
 0040               113 INT6    EQU     01000000B
 0080               114 INT7    EQU     10000000B
 0000               115 INTA    EQU     00000000B       ; NO INTERRUPTS ALLOWED AT ALL
                    116 ;
                    117 ; SYSTEM INTERRUPT I/O PORTS
                    118 ;
 00FD               119 SICP0   EQU     0FDH            ; INITIALIZATION COMMAND PORT 0
 00FC               120 SICP1   EQU     0FCH            ; INITIALIZATION COMMAND PORT 1
 00FD               121 SOCP0   EQU     0FDH            ; OPERATION COMMAND PORT 0
 00FC               122 SOCP1   EQU     0FCH            ; OPERATION COMMAND PORT 1
                    123 ;
                    124 ; DEDICATED PROM PROGRAMMER CONSTANTS (USED IN C,P,T COMMANDS)
                    125 ;
 0002               126 PCOMP   EQU     00000010B       ; PROGRAMMING COMPLETE
 0001               127 PGRDY   EQU     00000001B       ; PROM READY
 0020               128 PSOCK   EQU     00100000B       ; 16 PIN SOCKET SELECTED
 0010               129 PNIB    EQU     00010000B       ; SELECT UPPER NIBBLE
                    130 ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
                    131 ;
                    132 ; INTELLEC SERIES II I/O SUBSYSTEM CONSTANTS
                    133 ;
                    134 ; TTY AND CRT MODE INSTRUCTION DEFINITIONS, I.E. USART MODE CONTROL
                    135 ; WORD (FIRST CONTROL BYTE AFTER RESET)
                    136 ;
 0003               137 R64X    EQU     00000011B       ; 64 X BAUD RATE
 0002               138 R16X    EQU     00000010B       ; 16 X BAUD RATE
 0001               139 R1X     EQU     00000001B       ;  1 X BAUD RATE
 0000               140 SYNC    EQU     00000000B       ; SYNC MODE
 000C               141 CL8     EQU     00001100B       ; CHARACTER LENGTH = 8
 0008               142 CL7     EQU     00001000B       ; CHARACTER LENGTH = 7
 0004               143 CL6     EQU     00000100B       ; CHARACTER LENGTH = 6
 0000               144 CL5     EQU     00000000B       ; CHARACTER LENGTH = 5
 0010               145 PENB    EQU     00010000B       ; PARITY ENABLE
 0020               146 PEVEN   EQU     00100000B       ; EVEN PARITY
 00C0               147 ST2     EQU     11000000B       ; 2 STOP BITS
 0080               148 ST15    EQU     10000000B       ; 1.5 STOP BITS
 0040               149 ST1     EQU     01000000B       ; 1 STOP BIT
                    150 ;
                    151 ; TTY AND CRT COMMAND INSTRUCTION DEFINITIONS (USART COMMAND CONTROL WORD)
                    152 ;
 0001               153 TXEN    EQU     00000001B       ; TRANSMITTER ENABLE
 0002               154 DTR     EQU     00000010B       ; DATA TERMINAL READY
 0004               155 RXEN    EQU     00000100B       ; ENABLE RECEIVER
 0008               156 SBCH    EQU     00001000B       ; SEND BREAK CHARACTER
 0010               157 CLERR   EQU     00010000B       ; CLEAR ERROR
 0020               158 RTS     EQU     00100000B       ; SET REQUEST TO SEND OUTPUT
 0040               159 USRST   EQU     01000000B       ; USART RESET - RETURN TO MODE CONTROL CYCLE
 0080               160 ENHM    EQU     10000000B       ; ENABLE HUNT MODE
                    161 ;
                    162 ; TTY/CRT STATUS WORD BIT DEFINITIONS
                    163 ;
 0001               164 TRDY    EQU     00000001B       ; TRANSMIT READY
```

```
  LOC  OBJ          LINE        SOURCE STATEMENT

  0002               165 RRDY    EQU     00000010B       ; RECEIVE BUFFER READY
  0004               166 TXBE    EQU     00000100B       ; TRANSMIT BUFFER EMPTY
  0008               167 RPAR    EQU     00001000B       ; RECEIVE PARITY ERROR
  0010               168 ROV     EQU     00010000B       ; RECEIVE OVERRUN ERROR
  0020               169 RFR     EQU     00100000B       ; RECEIVE FRAMING ERROR
  0040               170 SYND    EQU     01000000B       ; SYNC DETECTED
  0080               171 DSR     EQU     10000000B       ; DATA SET READY INPUT
                     172 ;
                     173 ; TTY TAPE READER CONSTANTS
                     174 ;
  0028               175 RADCT   EQU     40              ; TTY TAPE READER ADVANCE TIMER COUNT
  00FA               176 RTOCT   EQU     250             ; TTY TAPE READER TIMEOUT COUNT
  0027               177 TADV    EQU     TXEN OR RXEN OR RTS OR DTR
  0025               178 COMD    EQU     TXEN OR RXEN OR RTS
                     179 ;
                     180 ; TTY I/O PORTS
                     181 ;
  00F4               182 TTYI    EQU     0F4H            ; TTY INPUT DATA PORT
  00F4               183 TTYO    EQU     0F4H            ; TTY OUTPUT DATA PORT
  00F5               184 TTYS    EQU     0F5H            ; TTY INPUT STATUS PORT
  00F5               185 TTYC    EQU     0F5H            ; TTY OUTPUT CONTROL PORT
                     186 ;
                     187 ; USER I/O PORTS
                     188 ;
  00F6               189 USCI    EQU     0F6H            ; USER INPUT DATA PORT
  00F7               190 USCS    EQU     0F7H            ; USER INPUT STATUS PORT
  00F6               191 USCO    EQU     0F6H            ; USER OUTPUT DATA PORT
  00F7               192 USCC    EQU     0F7H            ; USER OUTPUT CONTROL PORT
                     193 ;
                     194 ; INTERVAL TIMER CONSTANTS
                     195 ;
  0000               196 CTR0S   EQU     00000000B       ; COUNTER 0 SELECT
  0040               197 CTR1S   EQU     01000000B       ; COUNTER 1 SELECT
  0080               198 CTR2S   EQU     10000000B       ; COUNTER 2 SELECT
  0000               199 LCTR    EQU     00000000B       ; LATCHING COUNTER
  0020               200 RLMB    EQU     00100000B       ; READ/LOAD MSB ONLY
  0010               201 RLLB    EQU     00010000B       ; READ/LOAD LSB ONLY
  0030               202 RLLM    EQU     00110000B       ; READ/LOAD LSB,MSB
  0000               203 MODE0   EQU     00000000B       ; MODE 0
  0002               204 MODE1   EQU     00000010B       ; MODE 1
  0004               205 MODE2   EQU     00000100B       ; MODE 2
  0006               206 MODE3   EQU     00000110B       ; MODE 3
  0008               207 MODE4   EQU     00001000B       ; MODE 4
  000A               208 MODE5   EQU     00001010B       ; MODE 5
  0001               209 BCDC    EQU     00000001B       ; BCD COUNTER
  0007               210 B9600   EQU     7               ; 9600 BAUD RATE FACTOR
  0020               211 B2400   EQU     32              ; 2400 BAUD RATE FACTOR
  02BA               212 B0110   EQU     698             ;  110 BAUD RATE FACTOR
                     213 ;
                     214 ; INTERVAL TIMER (8253) I/O PORTS
                     215 ;
  00F0               216 CTR0P   EQU     0F0H            ; LOAD COUNTER 0 OUTPUT COMMAND PORT
  00F1               217 CTR1P   EQU     0F1H            ; LOAD COUNTER 1 OUTPUT COMMAND PORT
  00F2               218 CTR2P   EQU     0F2H            ; LOAD COUNTER 2 OUTPUT COMMAND PORT
  00F3               219 ITCP    EQU     0F3H            ; INTERVAL TIMER OUTPUT COMMAND PORT
```

```
 LOC  OBJ         LINE        SOURCE STATEMENT

                   220 ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
                   221 ;
                   222 ; I/O CONTROLLER SYSTEM CONSTANTS
                   223 ;
                   224 ; I/O CONTROLLER PORTS
                   225 ;
 00C0              226 IOCI    EQU     0C0H            ; I/O CONTROLLER INPUT DATA (FROM DBB) PORT
 00C0              227 IOCO    EQU     0C0H            ; I/O CONTROLLER OUTPUT DATA (TO DBB) PORT
 00C1              228 IOCS    EQU     0C1H            ; I/O CONTROLLER INPUT DBB STATUS PORT
 00C1              229 IOCC    EQU     0C1H            ; I/O CONTROLLER OUTPUT CONTROL COMMAND PORT
                   230 ;
                   231 ; CRT, KEYBOARD, AND FLOPPY DISK COMMANDS
                   232 ;
 0010              233 CRTC    EQU     10H             ; CRT OUTPUT DATA COMMAND
 0011              234 CRTS    EQU     11H             ; CRT DEVICE STATUS COMMAND
 0012              235 KEYC    EQU     12H             ; KEYBOARD INPUT DATA COMMAND
 0013              236 KSTS    EQU     13H             ; KEYBOARD DEVICE STATUS COMMAND
 0014              237 KINT    EQU     14H             ; RESERVED
 0015              238 WPBC    EQU     15H             ; FLOPPY PARAMETER BLOCK TRANSFER COMMAND
 0016              239 WPBCC   EQU     16H             ; FLOPPY PARAMETER BLOCK(CONT) TRANSFER COMMAND
 0017              240 WDBC    EQU     17H             ; FLOPPY DATA BLOCK OUTPUT COMMAND
 0018              241 WDBCC   EQU     18H             ; RESERVED
 0019              242 RDBC    EQU     19H             ; FLOPPY INPUT DATA BLOCK COMMAND
 001A              243 RDBCC   EQU     1AH             ; RESERVED
 001B              244 RRSTS   EQU     1BH             ; FLOPPY RESULT STATUS COMMAND
 001C              245 RDSTS   EQU     1CH             ; FLOPPY DEVICE STATUS COMMAND
                   246 ;
                   247 ; CRT, KEYBOARD, AND FLOPPY STATUS BITS
                   248 ;
 0001              249 KRDY    EQU     00000001B       ; KEYBOARD READY WITH DATA
 0001              250 FRDY    EQU     00000001B       ; FLOPPY READY FOR DATA
                   251 ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
                   252 ;
                   253 ; PARALLEL I/O SYSTEM CONSTANTS
                   254 ;
                   255 ; PARALLEL I/O PORTS
                   256 ;
 00F8              257 PIOI    EQU     0F8H            ; PARALLEL I/O INPUT DATA (FROM DBB) PORT
 00F8              258 PIOO    EQU     0F8H            ; PARALLEL I/O OUTPUT DATA (TO DBB) PORT
 00F9              259 PIOS    EQU     0F9H            ; PARALLEL I/O INPUT DBB STATUS PORT
 00F9              260 PIOC    EQU     0F9H            ; PARALLEL I/O OUTPUT CONTROL COMMAND PORT
                   261 ;
                   262 ; PTR, PTP, LPT AND UPP COMMANDS
                   263 ;
 0010              264 RDRC    EQU     010H            ; READER DATA TRANSFER COMMAND
 0060              265 PTRREV  EQU     01100000B       ; READER REVERSE DIRECTION 1 FRAME OPTION
 0040              266 PTRADV  EQU     01000000B       ; READER ADVANCE DIRECTION 1 FRAME OPTION
 0011              267 RSTC    EQU     011H            ; READER DEVICE STATUS COMMAND
 0012              268 PUNC    EQU     012H            ; PUNCH DATA TRANSFER COMMAND
 0013              269 PSTC    EQU     013H            ; PUNCH DEVICE STATUS COMMAND
 0014              270 LPTC    EQU     014H            ; LINE PRINTER DATA TRANSFER COMMAND
 0015              271 LSTC    EQU     015H            ; LINE PRINTER STATUS COMMAND
 0016              272 WPPC    EQU     016H            ; WRITE TO UPP COMMAND
 0017              273 RPPC    EQU     017H            ; READ FROM UPP COMMAND
 0018              274 RPSTC   EQU     018H            ; READ UPP STATUS COMMAND
 LOC  OBJ          LINE        SOURCE STATEMENT

                    275 ;
                    276 ; LPT, PTR AND PTP STATUS BITS
                    277 ;
 0001               278 LPTRY   EQU     00000001B       ; LPT READY
 0001               279 PTRDY   EQU     00000001B       ; PTR READY WITH DATA
 0001               280 PTPRY   EQU     00000001B       ; PTP READY FOR DATA
                    281 ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
                    282 ;
                    283 ; PARALLEL I/O AND I/O CONTROLLER SYSTEM COMMANDS
                    284 ;
 0000               285 PACIFY  EQU     00H             ; REINITIALIZE SYSTEM
 0001               286 ERESET  EQU     01H             ; ERROR RESET
 0002               287 SYSTAT  EQU     02H             ; SYSTEM STATUS
 0003               288 DSTAT   EQU     03H             ; DEVICE STATUS
 0004               289 SRQDAK  EQU     04H             ; DEVICE SERVICE REQUEST ACK
 0005               290 SRQACK  EQU     05H             ; SYSTEM SERVICE REQUEST ACK
 0006               291 SRQ     EQU     06H             ; SERVICE REQUEST
                    292 ;
                    293 ; PARALLEL I/O AND I/O CONTROLLER DIAGNOSTIC COMMANDS
                    294 ;
 0007               295 DECHO   EQU     07H             ; DATA ECHO TEST
 0008               296 CSMEM   EQU     08H             ; CHECKSUM MEMORY
 0009               297 TRAM    EQU     09H             ; TEST RAM
 000A               298 SINT    EQU     0AH             ; SYSTEM INTERRUPT CONTROL
                    299 ;
                    300 ;
                    301 ; PARALLEL I/O AND I/O CONTROLLER STATUS CONSTANTS
                    302 ;
 0001               303 OBF     EQU     00000001B       ; SLAVE OUTPUT BUFFER IS FULL
 0002               304 IBF     EQU     00000010B       ; SLAVE INPUT BUFFER IS FULL
 0004               305 F0      EQU     00000100B       ; FLAG 0 - SLAVE IS BUSY, MASTER IS LOCKED OUT
 0008               306 CNOTD   EQU     00001000B       ; DBB CONTAINS CONTROL INFO NOT DATA
                    307 ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
                    308 ;
                    309 ; FDCC (FLOPPY DISKETTE CHANNEL COMMAND) CONSTANTS
                    310 ;
 0004               311 OPCPL   EQU     4               ; DISK COMPLETION STATUS
 0079               312 LOWW    EQU     79H             ; LOW(IOPB)
 007A               313 HI      EQU     7AH             ; HIGH(IOPB)
 007B               314 RSTS    EQU     7BH             ; DISK RESULT STATUS INPUT PORT
 0078               315 DSTS    EQU     78H             ; DISK STATUS INPUT PORT
 3000               316 TRK0    EQU     3000H           ; FIRST ADDRESS OF DISK BOOTSTRAP
                    317 ;
                    318 ;       CONDITIONAL ASSEMBLY SWITCHES
                    319 ;
 0000               320 FALSE   EQU     0
 FFFF               321 TRUE    EQU     NOT FALSE
 00FF               322 HMSK    EQU     0FFH            ; SAFE MOVE OF 16 BITS INTO 8 BIT REGISTER
                    323 ;
                    324 ; GLOBAL CONSTANTS
                    325 ;
 0070               326 ONEMS   EQU     112             ; 1 MILLISECOND TIME CONSTANT
 00FA               327 TOUT    EQU     250             ; 250 MS. COUNTER FOR READER TIMEOUT
 000D               328 CR      EQU     0DH             ; ASCII VALUE OF CARRIAGE RETURN
 000A               329 LF      EQU     0AH             ; ASCII VALUE OF LINE FEED
 LOC  OBJ          LINE        SOURCE STATEMENT

 0003               330 ETX      EQU     03H              ; MONITOR BREAK CHARACTER (CONTROL C)
                    331 ;
                    332 ; MONITOR I/O STATUS BYTE MASKS AND VALUES
                    333 ;
 00FC               334 CMSK     EQU     11111100B        ; MASK FOR LOCAL CONSOLE I/O
 00F3               335 RMSK     EQU     11110011B        ; MASK FOR READER INPUT
 00CF               336 PMSK     EQU     11001111B        ; MASK FOR PUNCH OUTPUT
 003F               337 LMSK     EQU     00111111B        ; MASK FOR LIST OUTPUT
                    338 ;-----
 0000               339 CTTY     EQU     00000000B        ; LOCAL CONSOLE = TTY
 0001               340 CCRT     EQU     00000001B        ; LOCAL CONSOLE = CRT
 0002               341 BATCH    EQU     00000010B        ; BATCH MODE:
                    342                                   ; CONSOLE INPUT = READER, CONSOLE OUTPUT = LIST
 0003               343 CUSE     EQU     00000011B        ; USER DEFINED LOCAL CONSOLE I/O
                    344 ;-----
 0000               345 RTTY     EQU     00000000B        ; READER = TTY
 0004               346 RPTR     EQU     00000100B        ; READER = PTR
 0008               347 RUSE1    EQU     00001000B        ; USER DEFINED READER (1)
 000C               348 RUSE2    EQU     00001100B        ; USER DEFINED READER (2)
                    349 ;-----
 0000               350 PTTY     EQU     00000000B        ; PUNCH = TTY
 0010               351 PPTP     EQU     00010000B        ; PUNCH = PTP
 0020               352 PUSE1    EQU     00100000B        ; USER DEFINED PUNCH (1)
 0030               353 PUSE2    EQU     00110000B        ; USER DEFINED PUNCH (2)
                    354 ;-----
 0000               355 LTTY     EQU     00000000B        ; LIST = TTY
 0040               356 LCRT     EQU     01000000B        ; LIST = CRT
 0080               357 LLPT     EQU     10000000B        ; LIST = LPT
 00C0               358 LUSE     EQU     11000000B        ; USER DEFINED LIST
                    359 ;
                    360 ; LOCAL I/O SUBSYSTEM INTERRUPT PORTS
                    361 ;
 00FB               362 IICP0    EQU     0FBH             ; INITIALIZATION COMMAND PORT 0
 00FA               363 IICP1    EQU     0FAH             ; INITIALIZATION COMMAND PORT 1
 00FB               364 IOCP0    EQU     0FBH             ; OPERATION COMMAND PORT 0
 00FA               365 IOCP1    EQU     0FAH             ; OPERATION COMMAND PORT 1
                    366 ;
                    367 ; LOCAL INTERRUPT STATUS AND CONTROL BITS
                    368 ;
 0001               369 ITTYO    EQU     00000001B        ; TTY OUTPUT INTERRUPT
 0002               370 ITTYI    EQU     00000010B        ; TTY INPUT INTERRUPT
 0004               371 IPTP     EQU     00000100B        ; PTP OUTPUT INTERRUPT
 0008               372 IPTR     EQU     00001000B        ; PTR INPUT INTERRUPT
 0010               373 ICRTO    EQU     00010000B        ; CRT OUTPUT INTERRUPT
 0020               374 ICRTI    EQU     00100000B        ; CRT INPUT INTERRUPT
 0040               375 ILPT     EQU     01000000B        ; LPT OUTPUT INTERRUPT
 0080               376 MENB     EQU     10000000B        ; ENABLE MONITOR INTERRUPTS EXCEPT LEVEL 7
                    377 ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
                    378 ;
                    379 ; BOOTSTRAP CONSTANTS
                    380 ;
 00E7               381 FSTOP    EQU     0E7H             ; FULL SYSTEM TOP OF MEMORY ADDRESS
 00F7               382 FSTP     EQU     0F7H             ; FULL SYSTEM TOP PAGE ADDRESS
 0004               383 FDOC     EQU     004H             ; FLOPPY DISK OPERATION COMPLETE
 007F               384 ACHRM    EQU     07FH             ; ASCII CHARACTER MASK
 LOC  OBJ         LINE        SOURCE STATEMENT

 ØØFF                 ITIMO   EQU     ØFFH            ; IOC TIMEOUT CONSTANT
 ØØFF                 LBMK    EQU     ØFFH            ; LOWER BYTE MASK
 ØØ41                 ICFG    EQU     Ø41H            ; CONSOLE CONFIGURATION STATUS
 ØØØ1                 ICNP    EQU     ØØ1H            ; INTEGRATED CONSOLE NOT PRESENT STATUS
 ØØ4Ø                 LSTE    EQU     Ø4ØH            ; LIST DEVICE VALUE FOR CONSOLE
 Ø4CD                 RTCC    EQU     1229            ; REAL TIME CLOCK 1MS CONSTANT
 ØØØ8                 DPRNT   EQU     Ø8H             ; DISK READY MASK
 ØDØØ                 TRKL    EQU     26*128          ; TRACK LENGTH
 ØØØ4                 PARML   EQU     4               ; PARAMETER LENGTH - 1
 F8Ø9                 COP     EQU     ØF8Ø9H          ; ENTRY POINT FOR CONSOLE OUT
 F821                 IOCDP1  EQU     ØF821H          ; ENTRY POINT FOR IOC DRIVER 1
 F844                 IOCDP2  EQU     ØF844H          ; ENTRY POINT FOR IOC DRIVER 2
                      ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
                      ;
                      ; PAGE Ø DEDICATED RAM LOCATIONS, INITIALIZED BY BOOTSTRAP PROM CODE.
                      ;
 ØØØØ                         ORG     Ø
                      RESET:
 ØØØØ                         DS      3               ; TRAP TO MONITOR RESTART
                      IOBYT:
 ØØØ3                         DS      1               ; I/O SYSTEM STATUS BYTE
                      MEMTOP:
 ØØØ4                         DS      2               ; TOP OF RAM, ONLY H SAVED
                      INITIO:
 ØØØ6                         DS      1               ; INITIAL I/O CONFIGURATION
                      ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
                      ;
                      ; BOOTSTRAP PROM CODE
                      ;
 E8ØØ                 BBASE   SET     ØE8ØØH
 E8ØØ                         ORG     BBASE
 E8ØØ C3Ø6E8                  JMP     BSØ             ; BRANCH AROUND DATE CODE BYTE
                      INIT:
 E8Ø3 ØØ                      DB      Ø               ; INITIALLY
                                                      ; CONSOLE = TTY,
                                                      ; READER = TTY,
                                                      ; PUNCH = TTY,
                                                      ; LIST = TTY
 E8Ø4 Ø3Ø1                    DW      DATE            ; DATE STAMP FOR BOOTSTRAP PROM
                      ;
                      ; FUNCTIONS:
                      ;
                      ;       A.      INITIALIZE INTERRUPT SYSTEM AND REAL TIME CLOCK
                      ;               Ø. INITIALIZE PORT ØFFH (CPUC)
                      ;               1. PROGRAM SYSTEM INTERRUPTS  (8259)
                      ;               2. MASK ALL SYSTEM INTERRUPTS BUT TRAP LOGIC
                      ;               3. PROGRAM I/O SUBSYSTEM INTERRUPTS  (8259)
                      ;               4. MASK ALL I/O SUBSYSTEM INTERRUPTS
                      ;               5. PROGRAM REAL TIME CLOCK
                      ;
                      BSØ:
 E8Ø6 F3                      DI                      ; DISABLE INTERRUPT SYSTEM
 E8Ø7 3EØ2                    MVI     A,MOVBOT        ; TURN ON RAM (ROM WILL NOW RESPOND ONLY TO ADDRESS E8ØØH)
 E8Ø9 D3FF                    OUT     CPUC
 E8ØB 3EØ1                    MVI     A,BOVROF        ; TURN OFF BUS OVERRIDE
 LOC  OBJ         LINE        SOURCE STATEMENT

 E8ØD D3FF         44Ø          OUT     CPUC
 E8ØF 3EØ5         441          MVI     A,ENABL          ; PSEUDO ENABLE OF INTERRUPTS
 E811 D3FF         442          OUT     CPUC
 E813 3EØ8         443          MVI     A,ENAXP          ; ENABLE AUXILIARY PROM
 E815 D3FF         444          OUT     CPUC
 E817 3E12         445          MVI     A,ICW1           ; OUTPUT INITIALIZATION COMMAND WORD 1
 E819 D3FD         446          OUT     SICPØ            ;       TO SYSTEM 8259
 E81B D3FB         447          OUT     IICPØ            ;       TO I/O 8259
 E81D 3EØØ         448          MVI     A,ICW2           ; OUTPUT INITIALIZATION COMMAND WORD 2
 E81F D3FC         449          OUT     SICP1            ;       TO SYSTEM 8259
 E821 D3FA         45Ø          OUT     IICP1            ;       TO I/O 8259
 E823 3EFE         451          MVI     A,NOT INTØ       ; INITIALIZE MASK REGISTER
 E825 D3FC         452          OUT     SOCP1            ;       FOR SYSTEM 8259
 E827 3EFF         453          MVI     A,NOT INTA       ; INITIALIZE MASK REGISTER
 E829 D3FA         454          OUT     IOCP1            ;       FOR I/O 8259
 E82B 3EB6         455          MVI     A,CTR2S OR MODE3 OR RLLM ; INITIALIZE 1MS REAL TIME CLOCK
 E82D D3F3         456          OUT     ITCP
 E82F 21CDØ4       457          LXI     H,RTCC
 E832 7D           458          MOV     A,L
 E833 D3F2         459          OUT     CTR2P
 E835 7C           46Ø          MOV     A,H
 E836 D3F2         461          OUT     CTR2P
                   462 ;
                   463 ;        B.      INITIALIZE RAM.
                   464 ;                1. COMPUTE SIZE OF RAM MEMORY.
                   465 ;                2. SET UP DEDICATED MEMORY LOCATIONS
                   466 ;                        USER I/O ENTRY POINTS (TOP OF MEMORY)
                   467 ;                        EXIT TEMPLATE
                   468 ;                        USER REGISTERS
                   469 ;                        USER INTERRUPT MASK
                   47Ø ;                        USER STACK
                   471 ;                        MONITOR STACK
                   472 ;                        RESTART ROUTINE JUMP ADDRESS
                   473 ;
 E838 21ØØØØ       474          LXI     H,Ø              ; INITIAL VALUE H:=Ø, L:=Ø
                   475 BS1:
 E83B 24           476          INR     H                ; INCREMENT BY 256 BYTE PAGES, I.E.1ØØH,2ØØH,...,F8ØØH
 E83C 7E           477          MOV     A,M              ; FETCH CONTENTS OF MEMORY
 E83D 2F           478          CMA                      ; INVERT IT
 E83E 77           479          MOV     M,A              ; ATTEMPT TO WRITE IT BACK INTO MEMORY
 E83F BE           48Ø          CMP     M                ; IS LOCATION READ/WRITE, I.E. EXISTING RAM
 E84Ø 2F           481          CMA                      ; INVERT AGAIN BACK TO ORIGINAL VALUE
 E841 77           482          MOV     M,A              ; WRITE ORIGINAL DATA VALUE BACK IN
 E842 CA3BE8       483          JZ      BS1              ; YES, CONTINUE (I.E. STILL CONTIGUOUS RAM)
 E845 2B           484          DCX     H                ; OTHERWISE, IT'S LAST ADDRESS IN RAM
                   485                                   ;   UP TO ØE7FFH
 E846 3EE7         486          MVI     A,FSTOP          ; LOAD FULL-SYSTEM-UP-TO-BOOT-ROM PAGE ADDRESS
 E848 BC           487          CMP     H                ; TEST FOR FULL-SYSTEM-UP-TO-BOOT ROM
 E849 C262E8       488          JNZ     BS2              ; JUMP IF LESS THAN ØE7FFH IN RAM
                   489                                   ; AT THIS POINT WE HAVE CONTIGUOUS RAM UP TO
                   49Ø                                   ;   ØE7FFH; SKIP OVER ØE8ØØ-EFFFH WHICH IS
                   491                                   ;   SHADOWED BY BOOT ROM AND THEREFORE
                   492                                   ;   INACCESSIBLE; CONTINUE TESTING RAM FROM
                   493                                   ;   ØFØØØH
 E84C 21ØØEF       494          LXI     H,ØEFØØH
```

```
LOC  OBJ        LINE        SOURCE STATEMENT

                BS1X:
E84F 24                 INR     H                  ; INCREMENT BY 256 BYTE PAGES
E85Ø 7E                 MOV     A,M                ; FETCH CONTENTS OF MEMORY
E851 2F                 CMA                        ; INVERT IT
E852 77                 MOV     M,A                ; ATTEMPT TO WRITE IT BACK INTO MEMORY
E853 BE                 CMP     M                  ; IS LOCATION READ/WRITE, I.E. EXISTING RAM
E854 2F                 CMA                        ; INVERT IT BACK AGAIN TO ORIGINAL VALUE
E855 77                 MOV     M,A                ; WRITE ORIGINAL DATA VALUE BACK IN
E856 CA4FE8             JZ      BS1X               ; YES, CONTINUE (I.E. STILL CONTIGUOUS RAM)
E859 2B                 DCX     H                  ; OTHERWISE HL POINT TO LAST CONTIGUOUS
                                                   ;   BYTE OF RAM
E85A 3EFØ               MVI     A,ØFØH
E85C BC                 CMP     H                  ; TEST IF H > ØFØH (I.E. THAT TOP OF MEMORY
                                                   ;   IS AT LEAST 512 BYTES ABOVE SHADOW BOOT
                                                   ;   ROM BECAUSE WE NEED SPACE FOR MONITOR
                                                   ;   WORK TEMPLATE)
E85D DA62E8             JC      BS2                ; IF H > ØFØH THEN CARRY=1 AND HL CONTAIN
                                                   ;    TRUE TOP OF MEMORY
E86Ø 26E7               MVI     H,FSTOP            ; OTHERWISE H <= ØFØH THEN CARRY=Ø, SO
                                                   ;    SET TOP OF MEMORY TO ØE7FFH, WHICH IS
                                                   ;    JUST BELOW THE START OF SHADOW BOOT ROM
                BS2:
E862 22Ø4ØØ             SHLD    MEMTOP             ; STORE TOP OF MEMORY
E865 Ø1C8EA             LXI     B,TOS              ; MOVE EXIT TEMPLATE TO RAM
E868 69                 MOV     L,C
E869 F9                 SPHL                       ; SET MONITOR'S STACK POINTER
                BS3:
E86A ØA                 LDAX    B
E86B 77                 MOV     M,A
E86C ØC                 INR     C                  ; MOVE BOTH POINTERS
E86D 2C                 INR     L
E86E C26AE8             JNZ     BS3                ; END ON PAGE BOUNDARY
E871 2ED1               MVI     L,SLOC AND HMSK    ; SET UP INITIAL VALUE FOR USER STACK
E873 74                 MOV     M,H                ; LOWER HALF OF STACK POINTER IS KNOWN
E874 35                 DCR     M                  ; MERELY SET UPPER HALF
                                                   ; TRAP TO MONITOR (AT LOCATIONS Ø,1,2)
E875 3EC3               MVI     A,(JMP RESTART)
E877 32ØØØØ             STA     RESET
E87A 21D4FE             LXI     H,RESTART          ; SET UP RESTART Ø FOR BREAKPOINT
E87D 22Ø1ØØ             SHLD    RESET+1            ;    LOGIC
                ;
                ;       C.      PROGRAM I/O DEVICES.
                ;               1. BAUD RATE GENERATOR FOR CRT
                ;               2. USART FOR CRT
                ;               3. BAUD RATE GENERATOR FOR TTY
                ;               4. USART FOR TTY
                ;
E88Ø 3E76               MVI     A,CTR1S OR MODE3 OR RLLM
E882 D3F3               OUT     ITCP
E884 212ØØØ             LXI     H,B24ØØ            ; CRT BAUD RATE
E887 7D                 MOV     A,L
E888 D3F1               OUT     CTR1P
E88A 7C                 MOV     A,H
E88B D3F1               OUT     CTR1P
E88D 3ECE               MVI     A,ST2 OR R16X OR CL8
```

```
  LOC  OBJ         LINE        SOURCE STATEMENT

  E88F D3F7         550         OUT     USCC
  E891 3E27         551         MVI     A,TXEN OR DTR OR RXEN OR RTS
  E893 D3F7         552         OUT     USCC
  E895 3E36         553         MVI     A,CTR0S OR MODE3 OR RLLM
  E897 D3F3         554         OUT     ITCP
  E899 21BA02       555         LXI     H,B0110         ; TTY BAUD RATE
  E89C 7D           556         MOV     A,L
  E89D D3F0         557         OUT     CTR0P
  E89F 7C           558         MOV     A,H
  E8A0 D3F0         559         OUT     CTR0P
  E8A2 3ECE         560         MVI     A,ST2 OR R16X OR CL8
  E8A4 D3F5         561         OUT     TTYC
  E8A6 3E25         562         MVI     A,TXEN OR RXEN OR RTS
  E8A8 D3F5         563         OUT     TTYC
                    564 ;
                    565 ;       D.      DETERMINE IF INTEGRATED CONSOLE PRESENT
                    566 ;
  E8AA 2EFF         567         MVI     L,ITIMO         ; LOAD TIMEOUT CONSTANT
                    568 BS4:
  E8AC DBC1         569         IN      IOCS            ; INPUT DBB STATUS
  E8AE E607         570         ANI     IBF OR OBF OR F0; MASK OFF STATUS FLAGS
                    571                                 ; AND TEST FOR SLAVE PRESENCE
  E8B0 CAC0E8       572         JZ      BS5             ; JUMP IF INTEGRATED CONSOLE PRESENT
  E8B3 CD23EA       573         CALL    BDLY            ; DELAY 1 MS FOR ANY RESETS TO COMPLETE
  E8B6 CD23EA       574         CALL    BDLY            ; DELAY 1 MS.
  E8B9 2D           575         DCR     L               ; DECREMENT TIMER
  E8BA CAE2E8       576         JZ      BS8             ; JUMP IF TIME EXPIRED
  E8BD C3ACE8       577         JMP     BS4             ; OTHERWISE TRY AGAIN
                    578 BS5:
  E8C0 3E11         579         MVI     A,CRTS          ; LOAD CRT DEVICE STATUS COMMAND
  E8C2 D3C1         580         OUT     IOCC            ; OUTPUT COMMAND TO IOC CONTROL PORT
  E8C4 2EFF         581         MVI     L,ITIMO         ; LOAD TIMEOUT CONSTANT
                    582 BS6:
  E8C6 DBC1         583         IN      IOCS            ; INPUT DBB STATUS
  E8C8 E607         584         ANI     IBF OR OBF OR F0; MASK OFF STATUS FLAGS
  E8CA FE01         585         CPI     OBF             ; TEST FOR SLAVE DONE; SOMETHING FOR THE MASTER
  E8CC CADCE8       586         JZ      BS7             ; JUMP IF DONE
  E8CF CD23EA       587         CALL    BDLY            ; DELAY 1 MS FOR ANY RESETS TO COMPLETE
  E8D2 CD23EA       588         CALL    BDLY            ; DELAY 1 MS.
  E8D5 2D           589         DCR     L               ; DECREMENT TIMER
  E8D6 CAE2E8       590         JZ      BS8             ; JUMP IF TIME EXPIRED
  E8D9 C3C6E8       591         JMP     BS6             ; OTHERWISE, TRY AGAIN
                    592 BS7:
  E8DC DBC0         593         IN      IOCI            ; INPUT CRT DEVICE STATUS FROM DBB
  E8DE 0F           594         RRC                     ; TEST FOR CRT READY
  E8DF DAEAE8       595         JC      BS9             ; JUMP IF READY (INTEGRATED CRT PRESENT)
                    596 BS8:                            ; INTEGRATED CRT NOT PRESENT/READY SO RECORD THIS FACT
  E8E2 2A0400       597         LHLD    MEMTOP          ; LOAD TOP OF MEMORY PAGE ADDRESS
  E8E5 2ECC         598         MVI     L,BLOC+1 AND LBMK ; LOAD CONFIGURATION ADDRESS
  E8E7 3E01         599         MVI     A,ICNP          ; LOAD INTEGRATED CONSOLE NOT PRESENT
  E8E9 77           600         MOV     M,A             ; STORE IN CONFIGURATION BYTE IN EXIT TEMPLATE
                    601 BS9:
                    602 ;
                    603 ;       E.      LOAD ISIS.T0 IF DISKETTE 0 IS READY
                    604 ;
```

```
 LOC  OBJ         LINE        SOURCE STATEMENT

 E8EA AF                      XRA     A
 E8EB 2F                      CMA                       ; A-REG = ØFFH
 E8EC F5                      PUSH    PSW               ; THREE-VALUED FLAG:
                                                        ;    ØFFH IF NEITHER FDCC NOR ISD SELECTED
                                                        ;    ØØH  IF FDCC SELECTED
                                                        ;    Ø1H  IF ISD SELECTED
 E8ED DB78                    IN      DSTS              ; SAMPLE FDCC STATUS
                                                        ; STATUS = ØØH IF NO CONTROLLER PRESENT
 E8EF E6Ø8                    ANI     ØØØØ1ØØØB         ; IS FDCC CONTROLLER PRESENT?
 E8F1 CA2ØE9                  JZ      BS11              ; JUMP TO ISD SECTION IF FDCC NOT PRESENT
 E8F4 DB78                    IN      DSTS              ; SAMPLE FDCC STATUS AGAIN
 E8F6 ØF                      RRC                       ; DRIVE Ø READY STATUS ROTATED INTO CARRY BIT
 E8F7 D28EE9                  JNC     BSX1              ; JUMP TO MONITOR IF FDCC CONTROLLER PRESENT
                                                        ;    AND DRIVE Ø NOT READY
                                                        ; THE FOLLOWING CODE IS USED TO WRITE THE DISK IOPB TO
                                                        ; PROCESSOR MEMORY SO THAT IF ICE IS BEING USED TO DEBUG
                                                        ; THE BOOT/MONITOR, THE DISK CONTROLLER CAN ACCESS THE IOPB
 E8FA 21ØØ1Ø                  LXI     H,1ØØØH           ; LOAD POINTER TO DESTINATION MEMORY
 E8FD 1134EA                  LXI     D,IOPB            ; LOAD POINTER TO SOURCE MEMORY FOR IOPB
 E9ØØ Ø6Ø7                    MVI     B,7               ; LOAD IOPB LENGTH COUNT
                          MLP:
 E9Ø2 1A                      LDAX    D                 ; LOAD BYTE OF IOPB
 E9Ø3 77                      MOV     M,A               ; MOVE TO MEMORY
 E9Ø4 23                      INX     H                 ; INCREMENT IOPB POINTER
 E9Ø5 13                      INX     D                 ; INCREMENT MEMORY POINTER
 E9Ø6 Ø5                      DCR     B                 ; DECREMENT IOPB LENGTH COUNT
 E9Ø7 C2Ø2E9                  JNZ     MLP               ; CONTINUE UNTIL ALL OF IOPB MOVED
 E9ØA 21ØØ1Ø                  LXI     H,1ØØØH           ; RELOAD POINTER TO IOPB
 E9ØD 7D                      MOV     A,L               ; A CONTAINS LSB OF IOPB ADDRESS
 E9ØE D379                    OUT     LOWW              ; LOW(IOPB)
 E91Ø 7C                      MOV     A,H               ; A CONTAINS MSB OF IOPB ADDRESS
 E911 D37A                    OUT     HI                ; HIGH(IOPB), START DISK I/O
                          BS1Ø:
 E913 DB78                    IN      DSTS              ; WAIT FOR FDCC TO COMPLETE
 E915 E6Ø4                    ANI     OPCPL             ; TEST FOR DISK COMPLETION
 E917 CA13E9                  JZ      BS1Ø
 E91A F1                      POP     PSW               ; GET READY TO SET FLAG TO NEW VALUE
 E91B AF                      XRA     A                 ; SET A TO ZERO TO INDICATE DRIVE OTHER THAN INTEGRATED
                                                        ; FLOPPY WAS ACCESSED CORRECTLY
 E91C F5                      PUSH    PSW               ; SAVE ON STACK
 E91D C38EE9                  JMP     BSX1              ; BYPASS INTEGRATED FLOPPY BOOT
                          ;
                          ; LOAD ISIS.TØ FROM INTEGRATED DISK IF AVAILABLE
                          ;
                          BS11:
 E92Ø 2AØ4ØØ                  LHLD    MEMTOP            ; LOAD TOP OF MEMORY PAGE ADDRESS
 E923 2ECC                    MVI     L,BLOC+1 AND LBMK ; LOAD CONFIGURATION ADDRESS
 E925 7E                      MOV     A,M
 E926 ØF                      RRC                       ; TEST FOR INTEGRATED CONSOLE PRESENT
 E927 DA8EE9                  JC      BSX1              ; JUMP IF IOC NOT PRESENT OR FUNCTIONAL
 E92A Ø61C                    MVI     B,RDSTS           ; LOAD FLOPPY DEVICE STATUS COMMAND
 E92C CD21F8                  CALL    IOCDP1            ; READ STATUS FROM I/O CONTROLLER
 E92F E6Ø8                    ANI     DPRNT             ; TEST FOR DRIVE PRESENT
 E931 CA8EE9                  JZ      BSX1              ; JUMP IF NOT PRESENT
 E934 Ø61C                    MVI     B,RDSTS           ; LOAD FLOPPY DEVICE STATUS COMMAND
 LOC  OBJ          LINE        SOURCE STATEMENT

 E936 CD21F8        660         CALL    IOCDP1          ; READ STATUS FROM I/O CONTROLLER
                    661                                 ; SECOND STATUS READ USED TO INSURE DRIVE READY
 E939 0F            662         RRC                     ; TEST FOR DRIVE READY
 E93A D28EE9        663         JNC     BSX1            ; JUMP IF DRIVE NOT READY
 E93D F1            664         POP     PSW             ; UNLOAD STACK
 E93E AF            665         XRA     A               ; SET A TO 1 TO INDICATE
 E93F 3C            666         INR     A               ; INTEGRATED FLOPPY WAS ACCESSED
 E940 F5            667         PUSH    PSW             ; SAVE ON STACK
 E941 2134EA        668         LXI     H,IOPB          ; LOAD POINTER TO IOPB
 E944 0615          669         MVI     B,WPBC          ; LOAD WRITE IOPB COMMAND
 E946 4E            670         MOV     C,M             ; LOAD FIRST BYTE OF IOPB
 E947 CD44F8        671         CALL    IOCDP2          ; SEND BYTE TO IOC
 E94A 1E04          672         MVI     E,PARML         ; LOAD IOPB LENGTH REMAINING
 E94C 0616          673         MVI     B,WPBCC         ; LOAD WRITE IOPB CONTINUE COMMAND
                    674 BS12:
 E94E 23            675         INX     H               ; MOVE POINTER TO NEXT BYTE OF IOPB
 E94F 4E            676         MOV     C,M             ; MOVE TO C
 E950 CD44F8        677         CALL    IOCDP2          ; SEND TO IOC
 E953 1D            678         DCR     E               ; DECREMENT IOPB LENGTH
 E954 C24EE9        679         JNZ     BS12            ; CONTINUE UNTIL ALL DATA TRANSMITTED
 E957 061C          680         MVI     B,RDSTS         ; LOAD DEVICE STATUS COMMAND
                    681 BS13:
 E959 CD21F8        682         CALL    IOCDP1          ; READ STATUS FROM IOC
 E95C E604          683         ANI     OPCPL           ; TEST FOR OPERATION COMPLETE
 E95E CA59E9        684         JZ      BS13            ; LOOP UNTIL DONE
 E961 061B          685         MVI     B,RRSTS         ; LOAD RESULT STATUS COMMAND
 E963 CD21F8        686         CALL    IOCDP1          ; READ RESULT STATUS FROM IOC
 E966 32FE2F        687         STA     TRK0-2          ; SAVE FOR TEST LATER
 E969 B7            688         ORA     A               ; SET CONDITION CODES
 E96A C28EE9        689         JNZ     BSX1            ; JUMP IF DISK OPERATION UNSUCCESSFUL
 E96D 210030        690         LXI     H,TRK0          ; LOAD POINTER TO DISK DESTINATION ADDRESS
 E970 11000D        691         LXI     D,TRKL          ; LOAD TRACK LENGTH
 E973 0619          692         MVI     B,RDBC          ; LOAD DISK READ DATA COMMAND
 E975 CD21F8        693         CALL    IOCDP1          ; LOAD DATA FROM IOC
 E978 77            694         MOV     M,A             ; MOVE TO MEMORY
 E979 1B            695         DCX     D               ; DECREMENT LENGTH
 E97A 23            696         INX     H               ; MOVE POINTER TO NEXT LOCATION
                    697 BS14:
 E97B DBC1          698         IN      IOCS            ; INPUT DBB STATUS
 E97D E607          699         ANI     IBF OR OBF OR F0; MASK OFF STATUS FLAGS
 E97F FE01          700         CPI     OBF             ; TEST FOR DATA IN BUFFER
 E981 C27BE9        701         JNZ     BS14            ; JUMP IF NO DATA
 E984 DBC0          702         IN      IOCI            ; INPUT DATA FROM DBB
 E986 77            703         MOV     M,A             ; MOVE TO MEMORY
 E987 23            704         INX     H               ; MOVE POINTER TO NEXT LOCATION
 E988 1B            705         DCX     D               ; DECREMENT LENGTH
 E989 7A            706         MOV     A,D             ; LOAD D FOLLOWED BY E
 E98A B3            707         ORA     E               ;
 E98B C27BE9        708         JNZ     BS14            ; CONTINUE UNTIL DONE
                    709 ;
                    710 ;       F.      DETERMINE COLD START LOCAL CONSOLE.
                    711 ;
                    712 ;-------------------------------
                    713 ; CONSOLE IS EITHER INTEGRATED CRT, SERIAL CRT, OR TTY
                    714 BSX1:
```

```
 LOC  OBJ         LINE        SOURCE STATEMENT

 E98E 2AØ4ØØ       715         LHLD    MEMTOP           ; LOAD TOP OF MEMORY PAGE ADDRESS
 E991 2ECC         716         MVI     L,BLOC+1 AND LBMK ; LOAD CONFIGURATION ADDRESS
 E993 7E           717         MOV     A,M              ; LOAD INTEGRATED CONSOLE FLAG
 E994 ØF           718         RRC                      ; TEST FOR INTEGRATED CONSOLE PRESENT
 E995 DAA4E9       719         JC      BSX2             ; JUMP IF INTEGRATED CONSOLE NOT PRESENT
 E998 Ø613         72Ø         MVI     B,KSTS           ; LOAD KEYBOARD STATUS COMMAND
 E99A CD21F8       721         CALL    IOCDP1           ; READ STATUS FROM IOC
 E99D ØF           722         RRC                      ; TEST FOR KEYBOARD PRESENT
 E99E ØF           723         RRC
 E99F 1641         724         MVI     D,ICFG           ; LOAD INITIAL CONFIGURATION
 E9A1 DACCE9       725         JC      BSX5             ; JUMP IF KEYBOARD PRESENT
                   726 ;-----------------------------------
                   727 ; CONSOLE IS EITHER SERIAL CRT OR TTY
                   728 BSX2:
 E9A4 AF           729         XRA     A                ; ZERO A
 E9A5 57           73Ø         MOV     D,A              ; D CONTAINS ØH, I.E.C=T,R=T,P=T,L=T
 E9A6 DBF5         731         IN      TTYS             ; GET TTY STATUS
 E9A8 E6Ø2         732         ANI     RRDY             ; IS IT READY?
 E9AA CAB2E9       733         JZ      BSX3             ; JUMP IF TTY NOT READY
 E9AD DBF4         734         IN      TTYI             ; OTHERWISE GET CHARACTER FROM TTY
 E9AF C3BDE9       735         JMP     BSX4
                   736 BSX3:
 E9B2 1641         737         MVI     D,ICFG           ; LOAD INITIAL CONFIGURATION STATUS
 E9B4 DBF7         738         IN      USCS             ; GET SERIAL CRT STATUS
 E9B6 E6Ø2         739         ANI     RRDY             ; IS IT READY/
 E9B8 CAA4E9       74Ø         JZ      BSX2             ; JUMP BACKWARDS IF CRT NOT READY
 E9BB DBF6         741         IN      USCI             ; OTHERWISE,  GET CHARACTER FROM CRT
                   742 BSX4:
 E9BD E67F         743         ANI     7FH              ; MASK OUT PARITY BIT
 E9BF FE2Ø         744         CPI     ' '              ; DID USER TYPE IN A SPACE CHARACTER?
 E9C1 C2A4E9       745         JNZ     BSX2             ; START ALL OVER IF NOT A SPACE CHARACTER
                   746                                  ; IN CASE OF INTEGRATED CONSOLE PRESENT BUT
                   747                                  ;    KEYBOARD DISCONNECTED, THE CONSOLE IS
                   748                                  ;    NOW A SERIAL CRT, SO UPDATE
                   749                                  ;    INTEGRATED CONSOLE FLAG
 E9C4 2AØ4ØØ       75Ø         LHLD    MEMTOP           ; LOAD TOP OF MEMORY PAGE ADDRESS
 E9C7 2ECC         751         MVI     L,BLOC+1 AND LBMK ; LOAD CONFIGURATION ADDRESS
 E9C9 3EØ1         752         MVI     A,ICNP           ; INTEGRATED CONSOLE NOT PRESENT
 E9CB 77           753         MOV     M,A              ; STORE IN CONFIGURATION BYTE IN EXIT TEMPLATE
                   754 ;-------------------------------
                   755 ; AT THIS POINT THE CONSOLE DEVICE HAS BEEN DETERMINED
                   756 BSX5:
 E9CC 21Ø3ØØ       757         LXI     H,IOBYT          ; HL POINTS TO I/O STATUS BYTE
 E9CF 72           758         MOV     M,D              ; REPLACE MODIFIED I/O STATUS BYTE
 E9DØ 2EØ6         759         MVI     L,INITIO         ; HL POINTS TO INITIAL I/O STATUS BYTE
 E9D2 72           76Ø         MOV     M,D              ; SET INITIAL I/O STATUS BYTE
                   761 ;
                   762 ;       G.      CALL THE DIAGNOSTIC PROGRAM
                   763 ;
 E9D3 CDØ3EB       764         CALL    DIAGBT
                   765 ;
                   766 ;       H.      IF DISK IS READY, TRANSFER TO ISIS.TØ
                   767 ;
 E9D6 F1           768         POP     PSW              ; UNLOAD FLAG
 E9D7 B7           769         ORA     A                ; SET CONDITION CODES
 LOC  OBJ          LINE        SOURCE STATEMENT

 E9D8 C2EAE9        77Ø         JNZ     BSX6            ; JUMP IF INTEGRATED CONSOLE ACCESSED
 E9DB DB7B          771         IN      RSTS            ; SAMPLE FDCC RESULT STATUS
 E9DD B7            772         ORA     A               ; SET CONDITION CODES
 E9DE C218EA        773         JNZ     BSX1Ø           ; JUMP IF ERROR CONDITION
 E9E1 DB78          774         IN      DSTS            ; SAMPLE FDCC STATUS
 E9E3 ØF            775         RRC                     ; IS IT READY?
 E9E4 D2ØDEA        776         JNC     BSX9            ; JUMP TO MONITOR IF DISK NOT READY
                    777                                 ; OTHERWISE, PRIOR TO TRANSFERRING CONTROL
                    778                                 ;    TO TØ.BOOT, WRITE AN INSTRUCTION TO
                    779                                 ;    TURN OFF BOOTSTRAP PROM
 E9E7 C3FEE9        78Ø         JMP     BSX8
                    781 BSX6:
 E9EA Ø7            782         RLC                     ; TEST FOR NON DISK ACCESS
 E9EB DAØDEA        783         JC      BSX9            ; JUMP IF NO ACCESS
 E9EE 3AFE2F        784         LDA     TRKØ-2          ; LOAD TEMPORARY STORAGE FOR RESULT BYTE
 E9F1 B7            785         ORA     A               ; SET CONDITION CODES
 E9F2 C218EA        786         JNZ     BSX1Ø           ; JUMP IF ERROR CONDITION
 E9F5 Ø61C          787         MVI     B,RDSTS         ; LOAD FLOPPY DEVICE STATUS COMMAND
 E9F7 CD21F8        788         CALL    IOCDP1          ; READ STATUS FROM I/O CONTROLLER
 E9FA ØF            789         RRC                     ; TEST FOR DRIVE READY
 E9FB D2ØDEA        79Ø         JNC     BSX9            ; JUMP IF NOT READY
                    791 BSX8:
 E9FE 3ED3          792         MVI     A,(OUT CPUC)    ; LOAD OUTPUT INSTRUCTION
 EAØØ 32FE2F        793         STA     TRKØ-2          ; STORE IN RAM BEFORE DISK BOOT
 EAØ3 3EFF          794         MVI     A,CPUC          ; LOAD PORT ADDRESS
 EAØ5 32FF2F        795         STA     TRKØ-1
 EAØ8 3EØ4          796         MVI     A,BTDGOF        ; TURN OFF BOOTSTRAP/DIAGNOSTIC ROM
 EAØA C3FE2F        797         JMP     TRKØ-2          ; EFFECT IS SAME AS: MVI   A,BTDGOF
                    798                                 ;                    OUT   CPUC
                    799                                 ;                    JMP   TRKØ
                    8ØØ ;
                    8Ø1 ;       I.      OTHERWISE, TYPE SIGN-ON FOR RAM MONITOR
                    8Ø2 ;
                    8Ø3 BSX9:
 EAØD 213BEA        8Ø4         LXI     H,VERS          ; HL POINTS TO ADDRESS OF SIGN-ON MESSAGE
 EA1Ø Ø61B          8Ø5         MVI     B,LVER          ; B CONTAINS LENGTH OF MESSAGE
 EA12 CD2AEA        8Ø6         CALL    PRTM            ; PRINT SIGN-ON MESSAGE
                    8Ø7 ;
                    8Ø8 ;       J.      BOOTSTRAP ALL DONE, SO BRANCH TO MONITOR.
                    8Ø9 ;
 EA15 C3ØØF8        81Ø         JMP     BEGIN           ; AT THIS POINT, INTERRUPTS ARE DISABLED
                    811 ;
                    812 ;       K.      PRINT DISK ERROR MESSAGE
                    813 ;
                    814 BSX1Ø:
 EA18 2156EA        815         LXI     H,ERMSG         ; HL POINTS TO ADDRESS OF DISK ERROR MESSAGE
 EA1B Ø6ØE          816         MVI     B,LERM          ; B CONTAINS LENGTH OF MESSAGE
 EA1D CD2AEA        817         CALL    PRTM            ; PRINT SIGN-ON MESSAGE
 EA2Ø C3ØDEA        818         JMP     BSX9            ; PRINT MESSAGE
                    819 ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
                    82Ø ;
                    821 ;       BDLY - BOOTSTRAP DELAY 1 MS SUBROUTINE
                    822 ;
                    823 BDLY:
 EA23 ØE7Ø          824         MVI     C,ONEMS         ; LOAD 1 MS. CONSTANT
 LOC  OBJ         LINE        SOURCE STATEMENT

                   825 BDLY1:
 EA25 0D           826         DCR     C                ; DECREMENT COUNTER
 EA26 C225EA       827         JNZ     BDLY1            ; JUMP IF NOT EXPIRED
 EA29 C9           828         RET
                   829 ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
                   830 ;
                   831 ;       PRTM - PRT SUBROUTINE FOR SIGN-ON MESSAGES
                   832 ;
                   833 PRTM:
 EA2A 4E           834         MOV     C,M              ; C CONTAINS A CHARACTER FROM THE MESSAGE
 EA2B CD09F8       835         CALL    COP              ; PRINT ON CONSOLE
 EA2E 23           836         INX     H
 EA2F 05           837         DCR     B
 EA30 C22AEA       838         JNZ     PRTM             ; KEEP LOOPING UNTIL ENTIRE MESSAGE IS OUTPUT
 EA33 C9           839         RET
                   840 ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
                   841 ;
                   842 ;       DISK I/O PARAMETER BLOCK
                   843 ;
                   844 IOPB:
 EA34 80           845         DB      80H              ; IOCW, NO UPDATE BIT SET
 EA35 04           846         DB      04H              ; I/O INSTRUCTION, READ DISK 0
 EA36 1A           847         DB      26               ; READ 26 SECTORS
 EA37 00           848         DB      0                ; TRACK 0
 EA38 01           849         DB      1                ; SECTOR 1
 EA39 0030         850         DW      TRK0             ; LOAD ADDRESS
                   851 ;
                   852 ;       MONITOR SIGN-ON MESSAGE
                   853 ;
 EA3B 0D           854 VERS:   DB      CR,LF,'SERIES II MONITOR, V'
 EA3C 0A
 EA3D 53455249
 EA41 45532049
 EA45 49204D4F
 EA49 4E49544F
 EA4D 522C2056
 EA51 31           855         DB      VER/10+'0','.',VER MOD 10+'0'
 EA52 2E
 EA53 33
 EA54 0D           856         DB      CR,LF
 EA55 0A
 001B              857 LVER    EQU     $-VERS           ; LENGTH OF SIGN-ON MESSAGE
                   858 ;
                   859 ;       MONITOR ERROR SIGN-ON MESSAGE
                   860 ;
 EA56 0D           861 ERMSG:  DB      CR,LF,'DISK ERROR',CR,LF
 EA57 0A
 EA58 4449534B
 EA5C 20455252
 EA60 4F52
 EA62 0D
 EA63 0A
 000E              862 LERM    EQU     $-ERMSG          ; LENGTH OF ERROR SIGN-ON MESSAGE
 EA64 56           863 BTCKSM: DB      056H             ; BOOT CHKSUMS TO 55H
                   864 ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-***-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
```

INTELLEC SERIES II MONITOR, VERSION 1.3, 1 MARCH 1979

```
LOC  OBJ          SOURCE STATEMENT

                  ;
                  ; EXIT CODE TEMPLATE, TO BE EXECUTED IN RAM
                  ; THIS CODE IS ORIGINATED SO AS TO BE ALIGNED
                  ; AGAINST THE TOP OF A PAGE (1 PAGE = 256 BYTES)
                  ;
EAC8                      ORG     BBASE + Ø2C8H
                  TOS:                              ; BASE OF MONITOR WORK STACK
EACØ              USER    EQU     TOS-8             ; BASE OF DEFAULT USER WORK STACK
EAC8 EE           ELOC:   DB      ØEEH              ; E REGISTER STORAGE
EAC9 DD           DLOC:   DB      ØDDH              ; D REGISTER
EACA CC           CLOC:   DB      ØCCH              ; C REGISTER
EACB BB           BLOC:   DB      ØBBH              ; B REGISTER
EACC ØØ                   DB      Ø                 ; CONFIGURATION BYTE
                                                    ; BIT Ø = Ø IF INTEGRATED CRT IS PRESENT
                                                    ;       = 1 IF INTEGRATED CRT NOT PRESENT
EACD FE           ILOC:   DB      NOT INTØ          ; INTERRUPT MASK
EACE FF           FLOC:   DB      ØFFH              ; CPU FLAGS
EACF AA           ALOC:   DB      ØAAH              ; A REGISTER
EADØ CØ                   DB      USER AND HMSK     ; LOW(SP)
EAD1 ØØ           SLOC:   DB      Ø                 ; HIGH(SP)
                  ;
                  EXIT:                             ; MONITOR STACK ORIGIN
EAD2 F3                   DI                        ; DISABLE INTERRUPTS TO PROTECT THIS SEQUENCE
EAD3 D1                   POP     D                 ; RESTORE D,E
EAD4 C1                   POP     B                 ; RESTORE B,C
EAD5 F1                   POP     PSW
EAD6 D3FC                 OUT     SOCP1
EAD8 F1                   POP     PSW               ; RESTORE A AND FLAGS
EAD9 E1                   POP     H                 ; RESTORE ORIGINAL STACK VALUE
EADA F9                   SPHL
EADB 213412               LXI     H,1234H           ; RESTORE H,L; 1234H IS FILLER WHICH WILL BE
                                                    ;    OVERWRITTEN BY RESTART ROUTINE
EADC              LLOC    EQU     $-2
EADD              HLOC    EQU     $-1
EADE FB                   EI                        ; ENABLE INTERRUPTS
EADF C38967               JMP     6789H             ; RETURN TO INTERRUPTED CODE; 6789H IS FILLER
                                                    ;    WHICH WILL BE OVERWRITTEN BY 'G' COMMAND
                                                    ;    AND RESTART ROUTINE
EAE1              PLOC    EQU     $-1
EAE2 ØØØØ         TLOC:   DW      Ø                 ; TRAP 1 ADDRESS
EAE4 ØØ                   DB      Ø                 ; TRAP 1 VALUE
EAE5 ØØØØ                 DW      Ø                 ; TRAP 2 ADDRESS
EAE7 ØØ                   DB      Ø                 ; TRAP 2 VALUE
                  XTBL:                             ; EXTENSIBLE I/O ENTRY POINTS
                                                    ;    FILLED IN WHEN USER GIVES ADDRESS OF OWN
                                                    ;    DRIVER ROUTINE VIA IODEF SYSTEM CALL IN MONITOR
                  CILOC:
EAE8 C3ØØØØ               JMP     Ø
                  COLOC:
EAEB C3ØØØØ               JMP     Ø
                  R1LOC:
EAEE C3ØØØØ               JMP     Ø
                  R2LOC:
EAF1 C3ØØØØ               JMP     Ø
                  P1LOC:
```

```
 LOC  OBJ          LINE        SOURCE STATEMENT

 EAF4 C30000        920         JMP     0
                    921 P2LOC:
 EAF7 C30000        922         JMP     0
                    923 L1LOC:
 EAFA C30000        924         JMP     0
                    925 CSLOC:
 EAFD C30000        926         JMP     0
                    927 ENDX:                             ; THIS LABEL SHOULD BE AT 0EA00H.
                    928 ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
                    929 ; SELECTION CODES FOR USER I/O ENTRY POINTS
                    930 ;
 0000               931 UCI     EQU     (CILOC-XTBL)/3
 0001               932 UCO     EQU     (COLOC-XTBL)/3
 0002               933 UR1     EQU     (R1LOC-XTBL)/3
 0003               934 UR2     EQU     (R2LOC-XTBL)/3
 0004               935 UP1     EQU     (P1LOC-XTBL)/3
 0005               936 UP2     EQU     (P2LOC-XTBL)/3
 0006               937 UL1     EQU     (L1LOC-XTBL)/3
 0007               938 UCS     EQU     (CSLOC-XTBL)/3
                    939 ; END OF BOOTSTRAP PROM CODE
                    940 ;********************************************************************************
 EB00               941 DIAGMN  EQU     0EB00H            ; STARTING ADDRESS OF DIAGNOSTIC PROGRAM
                    942                                   ;    WHEN ENTERED FROM CALL FROM MONITOR
 EB03               943 DIAGBT  EQU     0EB03H            ; STARTING ADDRESS OF DIAGNOSTIC PROGRAM
                    944                                   ;    WHEN ENTERED FROM CALL FROM BOOT
 EB00               945         ORG     0EB00H            ; WHEN BURNING THE PROM, THIS SECTION OF CODE
                    946                                   ;    WILL BE OVERLAYED BY THE REAL DIAGNOSTIC
                    947                                   ;    PROGRAM.
 EB00 C9            948         RET
 EB01 00            949         NOP
 EB02 00            950         NOP
 EB03 C9            951         RET                       ; 0EB03H
                    952                                   ;    BOOTSTRAP/DIAGNOSTIC PROM
                    953 ;********************************************************************************
                    954 ;********************************************************************************
                    955 ;********************************************************************************
                    956 ;***                                                                          ***
                    957 ;***             START OF MONITOR PROPER                                      ***
                    958 ;***                                                                          ***
                    959 ;********************************************************************************
                    960 ;********************************************************************************
                    961 ;********************************************************************************
 F800               962 BASE    SET     0F800H            ; BASE ADDRESS OF MONITOR
 F800               963         ORG     BASE              ; TOP 2K OF 64K ADDRESS SPACE
                    964 ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
                    965 ;
                    966 ; BRANCH TABLE FOR I/O SYSTEM (EXTERNAL I/O ENTRY POINTS)
                    967 ;
                    968 ; THE MONITOR IS ENTERED AT ENTRY POINT 'BEGIN' VIA A JUMP FROM THE BOOTSTRAP;
                    969 ; THIS IN TURN LEADS TO A JUMP TO ENTRY POINT 'START'.  THE OTHER ENTRIES
                    970 ; IN THIS "TABLE" ARE EXTERNAL I/O ENTRY POINTS KNOWN TO THE USER PLUS
                    971 ; THE DATE, VERSION, AND COPYRIGHT STAMPS.
                    972 BEGIN:
 F800 C351F8        973         JMP     START0            ; RESET ENTRY POINT
 F803 C3BEFB        974         JMP     CI                ; LOCAL CONSOLE INPUT
```

```
  LOC  OBJ         LINE        SOURCE STATEMENT

  F806 C30FFC       975         JMP     RI                ; READER INPUT
  F809 C39FFC       976         JMP     CO                ; LOCAL CONSOLE OUTPUT
  F80C C3E9FC       977         JMP     PO                ; PUNCH OUTPUT
  F80F C31EFD       978         JMP     LO                ; LIST OUTPUT
  F812 C344FD       979         JMP     CSTS              ; LOCAL CONSOLE INPUT STATUS
  F815 C383FD       980         JMP     IOCHK             ; I/O SYSTEM STATUS
  F818 C387FD       981         JMP     IOSET             ; SET I/O CONFIGURATION
  F81B C38CFD       982         JMP     MEMCHK            ; COMPUTE SIZE OF MEMORY
  F81E C394FD       983         JMP     IODEF             ; DEFINE USER I/O ENTRY POINTS
  F821 C37FFF       984         JMP     IOCDR1            ; IOC INPUT
  F824 0301         985         DW      DATE              ; DATE STAMP FOR MONITOR ROM
  F826 C3ADFD       986         JMP     UI                ; UPP INPUT
  F829 C3BEFD       987         JMP     UO                ; UPP OUTPUT
  F82C C3CEFD       988         JMP     UPPS              ; UPP STATUS
  F82F 13           989         DB      VERH              ; VERSION STAMP FOR MONITOR ROM
  F830 28432949     990         DB      '(C)INTEL CORP1979'     ; COPYRIGHT NOTICE IN ASCII REP
  F834 4E54454C
  F838 20434F52
  F83C 50313937
  F840 39
  F841 C3A6FF       991         JMP     IOCCOM            ; IOCCOM ENTRY POINT
  F844 C394FF       992         JMP     IOCDR2            ; IOC OUTPUT
                    993 ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
                    994 ;
                    995 ; 'ERROR' - ENTERED VIA JUMP FROM VARIOUS ROUTINES WHEN AN ERROR IS DETECTED
                    996 ; PROCESS: ABNORMAL EXIT FOR ALL MONITOR ERROR CONDITIONS. BECAUSE OF THE
                    997 ;          UNKNOWN STATE OF THE MONITOR AS A RESULT OF A COMMAND OR DATA ERROR,
                    998 ;          THE VALUE OF THE MONITOR STACK POINTER IS REINITIALIZED AND
                    999 ;          EXECUTION CONTINUES TO THE MAIN COMMAND LOOP.
                   1000 ; INPUT: MEMTOP,TOS
                   1001 ; OUTPUT: SP POINTS TO BASE OF MONITOR STACK IN TOP PAGE OF CONTIGUOUS RAM
                   1002 ; MODIFIED: H,L, SP
                   1003 ; STACK USAGE:
                   1004 ;
                   1005 ; REGISTER USAGE
                   1006 ; X = MODIFIED BY THIS ROUTINE, CONTENTS UNDEFINED.
                   1007 ; S = SET BY THIS ROUTINE, RETURNED AS A RESULT.
                   1008 ; U = USED AS INPUT.
                   1009 ;        A -
                   1010 ;        B -              C - S
                   1011 ;        D -              E -
                   1012 ;        H - X            L - X
                   1013 ;        CARRY - X        ZERO - X
                   1014 ;        SIGN - X         PARITY - X
                   1015 ;        SP - S           PC -
                   1016 ;        STACK USAGE: 2 BYTES
                   1017 ERROR:
  F847 2A0400      1018         LHLD    MEMTOP            ; H POINTS TO TOP PAGE OF MEMORY
  F84A 2EC8        1019         MVI     L,TOS AND 0FFH    ; L POINTS TO BASE OF STACK WITHIN THAT PAGE
  F84C F9          1020         SPHL                      ; SP NOW POINTS TO BASE OF MONITOR STACK
  F84D CDDEFC      1021         CALL    COMC              ; OUTPUT THE ERROR INDICATOR CHAR '#'
  F850 23          1022         DB      '#'
                   1023                                   ; FALL THROUGH TO MAIN COMMAND LOOP
                   1024 ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
                   1025 ;
```

```
LOC  OBJ       SOURCE STATEMENT

               ; MAIN COMMAND LOOP.
               ;
               ; THIS LOOP IS THE STARTING POINT OF ALL COMMAND SEQUENCES.
               ; IT IS ENTERED VIA A JUMP FROM THE BEGINNING OF THE MONITOR PROPER CODE,
               ; A FALL THROUGH FROM THE ERROR ROUTINE, OR A RETURN FROM A MONITOR COMMAND
               ; ROUTINE.
               ; IN THIS CODE INTERRUPTS ARE ENABLED AND A CARRIAGE RETURN
               ; AND LINE FEED ARE TYPED ALONG WITH THE PROMPT CHARACTER, '.'.
               ; WHEN A CHARACTER IS ENTERED FROM THE LOCAL CONSOLE KEYBOARD, IT
               ; IS CHECKED FOR VALIDITY, THEN A BRANCH TO THE PROPER
               START0:
F851 3E04              MVI     A,BTDGOF        ; DISABLE BOOT, I.E. SWITCH BOOT PROM
F853 D3FF              OUT     CPUC            ;    OUT OF ADDRESSABLE MEMORY SPACE
               START:
F855 FB                EI                      ; ENABLE INTERRUPTS
F856 CDFEFD            CALL    CRLF            ; TYPE <CR>,<LF>
F859 CDDEFC            CALL    COMC            ; OUTPUT A PERIOD
F85C 2E                DB      '.'
F85D CD61FF            CALL    TI              ; GET A CHARACTER, ECHO IT.
F860 FE0D              CPI     CR              ; IS IT A CARRIAGE RETURN?
F862 CA55F8            JZ      START           ; JUMP IF IT IS
F865 D641              SUI     'A'             ; OTHERWISE TEST FOR A-Z (VALID COMMAND RANGE)
F867 FA47F8            JM      ERROR           ; LESS THAN A, NOT A VALID COMMAND
F86A 0E02              MVI     C,2             ; ASSUME THE COMMAND NEEDS 2 PARAMETERS
F86C 1155F8            LXI     D,START         ; SET UP PSEUDO RETURN ADDRESS TO SIMULATE
F86F D5                PUSH    D               ;    EFFECT OF A CALL. COMMANDS WHICH PERFORM
                                               ;    A RETURN WILL CAUSE THE STACK TO BE
                                               ;    POPPED, THUS RETURNING TO ENTRY POINT
                                               ;    START.  THE 'G' COMMAND, HOWEVER, WIPES
                                               ;    OUT THIS ADDRESS WITH ANOTHER ADDRESS
                                               ;    OF ITS OWN CHOOSING (I.E. USER'S PC).
F870 2182F8            LXI     H,CTBL          ; LOAD POINTER TO PROCESSING ROUTINE PTRS
F873 FE1A              CPI     LCT             ; TEST FOR OVERRUN
F875 F247F8            JP      ERROR           ; IF SO, THEN ERROR
F878 5F                MOV     E,A             ; OTHERWISE, MOVE INDEX TO DE
F879 1600              MVI     D,0
F87B 19                DAD     D
F87C 19                DAD     D               ; HL := CTBLBASE + (2 * INDEX); HL NOW POINTS
                                               ;    TO PROPER COMMAND IN COMMAND BRANCH TABLE
F87D 7E                MOV     A,M             ; GET LSB OF BRANCH LOCATION
F87E 23                INX     H
F87F 66                MOV     H,M             ; GET MSB OF BRANCH LOCATION
F880 6F                MOV     L,A             ; HL POINTS TO ADDRESS OF COMMAND CODE
F881 E9                PCHL                    ; TAKE THE BRANCH
               ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
               ;
               ; COMMAND BRANCH TABLE.
               ;
               ; THIS TABLE CONTAINS THE ADDRESSES OF THE ENTRY POINTS OF
               ; ALL THE COMMAND PROCESSING ROUTINES.  IT IS ENTERED FROM THE MAIN
               ; COMMAND LOOP.  NOTE THAT AN ENTRY TO 'ERROR'
               ; IS AN ERROR CONDITION, I.E., NO COMMAND CORRESPONDING TO THAT
               ; CHARACTER EXISTS.
               CTBL:
F882 B6F8              DW      ASSIGN          ; A - ASSIGN I/O UNITS
```

```
LOC  OBJ          LINE        SOURCE STATEMENT

F884 47F8                 DW      ERROR           ; B -
F886 47F8                 DW      ERROR           ; C -
F888 33F9                 DW      DISP            ; D - DISPLAY RAM MEMORY
F88A 5FF9                 DW      EOF             ; E - ENDFILE A HEXADECIMAL FILE
F88C 7DF9                 DW      FILL            ; F - FILL MEMORY
F88E 8CF9                 DW      GOTO            ; G - GO TO MEMORY ADDRESS
F89Ø D5F9                 DW      HEXN            ; H - HEXADECIMAL SUM AND DIFFERENCE
F892 47F8                 DW      ERROR           ; I -
F894 47F8                 DW      ERROR           ; J -
F896 47F8                 DW      ERROR           ; K -
F898 47F8                 DW      ERROR           ; L -
F89A FØF9                 DW      MOVE            ; M - MOVE MEMORY
F89C Ø1FA                 DW      NULL            ; N - PUNCH NULLS FOR LEADER ON PAPER TAPE
F89E 47F8                 DW      ERROR           ; O -
F8AØ 47F8                 DW      ERROR           ; P -
F8A2 14FA                 DW      QUERY           ; Q - QUERY I/O SYSTEM STATUS
F8A4 52FA                 DW      READ            ; R - READ HEXADECIMAL PAPER TAPE FILE
F8A6 BFFA                 DW      SUBS            ; S - SUBSTITUTE MEMORY
F8A8 47F8                 DW      ERROR           ; T -
F8AA 47F8                 DW      ERROR           ; U -
F8AC 47F8                 DW      ERROR           ; V -
F8AE DDFA                 DW      WRITE           ; W - WRITE FILE TO PAPER TAPE IN HEX FORMAT
F8BØ 26FB                 DW      X               ; X - EXAMINE AND MODIFY REGISTERS
F8B2 47F8                 DW      ERROR           ; Y -
F8B4 A6FB                 DW      Z               ; Z - INVOKE THE DIAGNOSTIC PROGRAM
ØØ1A                LCT   EQU     ($-CTBL)/2      ; LCT = NUMBER OF 16-BIT ENTRIES IN TABLE
                    ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
                    ;
                    ; 'A' COMMAND - ASSIGN I/O DEVICE
                    ;
                    ; THIS ROUTINE MAPS SYMBOLIC DEVICE IDENTIFIERS TO BITS
                    ; IN THE I/O STATUS BYTE (IOBYT) TO ALLOW FOR LOCAL CONSOLE
                    ; MODIFICATION OF SYSTEM I/O CONFIGURATION.
                    ASSIGN:
F8B6 CD61FF               CALL    TI              ; GET LOGICAL DEVICE CHARACTER (C,R,P,L)
F8B9 21Ø3F9               LXI     H,LTBL          ; ADDRESS OF MASTER TABLE
F8BC ØEØ4                 MVI     C,4             ; MAXIMUM OF 4 ENTRIES
                    ;     ----------------------------------------------------------------
                    ASØ:                          ; HL POINTS TO IDENTIFYING CHARACTER IN LTBL
F8BE BE                   CMP     M               ; DOES A-REG CONTAIN C,R,P, OR L?
F8BF 23                   INX     H               ; HL POINTS TO CORRESPONDING DEVICE MASK
F8CØ CACDF8               JZ      AS1             ; YES IT DOES
F8C3 23                   INX     H
F8C4 23                   INX     H
F8C5 23                   INX     H               ; HL POINTS TO NEXT 4-BYTE ENTRY IN LTBL
F8C6 ØD                   DCR     C               ; DECREMENT LOOP COUNT
F8C7 C2BEF8               JNZ     ASØ             ; TRY NEXT ENTRY
F8CA C347F8               JMP     ERROR           ; NO MATCH, ERROR
                    ;     ----------------------------------------------------------------------
                    AS1:                          ; USER HAS SPECIFIED A VALID LOGICAL DEVICE
F8CD 46                   MOV     B,M             ; B := LOGICAL DEVICE MASK
F8CE 23                   INX     H               ; HL CONTAINS SUBORDINATE PHYS.DEV.TBL.ADDRESS
F8CF 5E                   MOV     E,M             ; E CONTAINS LSB OF PDT ADDRESS
F8DØ 23                   INX     H
F8D1 56                   MOV     D,M             ; D CONTAINS MSB OF PDT ADDRESS
```

```
LOC  OBJ          SOURCE STATEMENT

F8D2 EB                   XCHG                        ; HL POINTS TO I/O SYSTEM PHYSICAL DEVICE
                                                      ;    TABLE (I.E. ACT,ART,APT, OR ALT)
              ;           ------------------------------------------------------------------
              ALUP1:                                  ; SCAN INPUT UNTIL '='
F8D3 CD61FF               CALL    TI
F8D6 FE3D                 CPI     '='
F8D8 C2D3F8               JNZ     ALUP1
              ;           ------------------------------------------------------------------
              ALUP2:                                  ; SCAN INPUT WHILE ' ' (BLANK)
F8DB CD61FF               CALL    TI
F8DE FE20                 CPI     ' '
F8E0 CADBF8               JZ      ALUP2
              ;           ------------------------------------------------------------------
F8E3 0E04                 MVI     C,4                 ; SET TABLE LENGTH
              AS2:                                    ; INDEX THROUGH PHYSICAL UNIT TABLE
F8E5 BE                   CMP     M                   ; COMPARE DEVICE CHAR WITH LEGAL VALUES
F8E6 23                   INX     H                   ; HL CONTAINS DEVICE SELECT BIT PATTERN
F8E7 CAF2F8               JZ      AS3                 ; USER HAS SPECIFIED A VALID PHYS.DEVICE ASSIGNMNT
F8EA 23                   INX     H                   ; HL POINTS TO NEXT ENTRY WITHIN THE TABLE
F8EB 0D                   DCR     C
F8EC C2E5F8               JNZ     AS2                 ; CONTINUE LOOKUP
F8EF C347F8               JMP     ERROR               ; ERROR RETURN
              ;           ----------------------------------------------------------
              AS3:
              ALUP3:                                  ; SCAN INPUT UNTIL <CR>
F8F2 CD61FF               CALL    TI
F8F5 FE0D                 CPI     CR
F8F7 C2F2F8               JNZ     ALUP3
F8FA 3A0300               LDA     IOBYT               ; GET I/O STATUS
F8FD A0                   ANA     B                   ; B CONTAINS LOG DEV MASK.  CLEAR OUT THE
                                                      ;    APPROPRIATE FIELD IN IOBYT BECAUSE WE ARE
                                                      ;    GOING TO CHANGE IT.
F8FE B6                   ORA     M                   ; PUT IN THE NEW STATUS FIELD
F8FF 320300               STA     IOBYT               ; RETURN IT TO MEMORY
F902 C9                   RET                         ; RETURN CONTROL TO MAIN COMMAND LOOP
              ;
              ; MASTER I/O DEVICE TABLE
              ; 4 BYTES/ENTRY
              ;
              ;      BYTE 0 = IDENTIFYING CHARACTER
              ;      BYTE 1 = LOGICAL DEVICE MASK
              ;      BYTES 2,3 = ADDRESS OF SUBORDINATE PHYSICAL DEVICE TABLE
              ;
              LTBL:
F903 43                   DB      'C',CMSK
F904 FC
F905 13F9                 DW      ACT
F907 52                   DB      'R',RMSK
F908 F3
F909 1BF9                 DW      ART
F90B 50                   DB      'P',PMSK
F90C CF
F90D 23F9                 DW      APT
F90F 4C                   DB      'L',LMSK
F910 3F
```

```
 LOC  OBJ         LINE        SOURCE STATEMENT

 F911 2BF9        1187          DW      ALT
                  1188 ;
                  1189 ; I/O SYSTEM PHYSICAL DEVICE TABLES
                  1190 ; 2 BYTES/ENTRY
                  1191 ;
                  1192 ;   BYTE 0 = IDENTIFYING CHARACTER
                  1193 ;   BYTE 1 = DEVICE SELECT BIT PATTERN
                  1194 ;
                  1195 ACT:
 F913 54          1196          DB      'T',CTTY        ; LOCAL CONSOLE = TTY
 F914 00
 F915 43          1197          DB      'C',CCRT        ; LOCAL CONSOLE = CRT
 F916 01
 F917 42          1198          DB      'B',BATCH       ; BATCH MODE LOCAL CONSOLE = READ,LIST
 F918 02
 F919 31          1199          DB      '1',CUSE        ; USER DEFINED LOCAL CONSOLE DEVICE
 F91A 03
                  1200 ART:
 F91B 54          1201          DB      'T',RTTY        ; READER = TTY
 F91C 00
 F91D 50          1202          DB      'P',RPTR        ; READER = PTR
 F91E 04
 F91F 31          1203          DB      '1',RUSE1       ; USER DEFINED READER DEVICE 1
 F920 08
 F921 32          1204          DB      '2',RUSE2       ; USER DEFINED READER DEVICE 2
 F922 0C
                  1205 APT:
 F923 54          1206          DB      'T',PTTY        ; PUNCH = TTY
 F924 00
 F925 50          1207          DB      'P',PPTP        ; PUNCH = PTP
 F926 10
 F927 31          1208          DB      '1',PUSE1       ; USER DEFINED PUNCH DEVICE 1
 F928 20
 F929 32          1209          DB      '2',PUSE2       ; USER DEFINED PUNCH DEVICE 2
 F92A 30
                  1210 ALT:
 F92B 54          1211          DB      'T',LTTY        ; LIST = TTY
 F92C 00
 F92D 43          1212          DB      'C',LCRT        ; LIST = CRT
 F92E 40
 F92F 4C          1213          DB      'L',LLPT        ; LIST = LPT
 F930 80
 F931 31          1214          DB      '1',LUSE        ; USER DEFINED LIST DEVICE
 F932 C0
                  1215 ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
                  1216 ;
                  1217 ; 'D' COMMAND - DISPLAY CONTENTS OF MEMORY ON LIST DEVICE
                  1218 ;
                  1219 ; THIS ROUTINE EXPECTS TWO HEXADECIMAL PARAMETERS SPECIFYING
                  1220 ; THE BOUNDS OF A MEMORY AREA TO BE DISPLAYED ON THE
                  1221 ; LIST DEVICE.  THE MEMORY AREA IS DISPLAYED 16 BYTES
                  1222 ; PER LINE, WITH THE MEMORY ADDRESS OF THE FIRST BYTE
                  1223 ; PRINTED FOR REFERENCE.  ALL LINES ARE BLOCKED INTO INTEGRAL
                  1224 ; MULTIPLES OF 16 FOR CLARITY, SO THE FIRST AND LAST LINES MAY
                  1225 ; BE LESS THAN 16 BYTES IN ORDER TO SYNCHRONIZE THE DISPLAY.
 LOC  OBJ          SOURCE STATEMENT

                   DISP:
 F933 CD39FE               CALL    EXPR            ; GET TWO ADDRESSES
 F936 D1                   POP     D               ; GET HIGH ADDRESS
 F937 E1                   POP     H               ; GET LOW ADDRESS
                   DI0:
 F938 CD6AFE               CALL    LCRLF           ; PRINT CR,LF
 F93B CD07FE               CALL    DADR            ; PRINT MEMORY ADDRESS
                   DI1:
 F93E 0E20                 MVI     C,' '
 F940 CD14FD               CALL    LOM             ; PRINT SPACE
 F943 7E                   MOV     A,M
 F944 CD0CFE               CALL    DBYTE           ; PRINT DATA
 F947 CD4CFE               CALL    HILO            ; TEST FOR COMPLETION
 F94A DA56F9               JC      DI2             ; RETURN TO MAIN LOOP
 F94D 7D                   MOV     A,L
 F94E E60F                 ANI     0FH             ; PRINT CR,LF,ADDRESS ON MULTIPLE OF 16
 F950 C23EF9               JNZ     DI1
 F953 C338F9               JMP     DI0
                   DI2:
 F956 CD6AFE               CALL    LCRLF           ; WRITE CR,LF
 F959 0E00                 MVI     C,0
 F95B CD14FD               CALL    LOM             ; WRITE A NULL TO TRIGGER CLOSE
 F95E C9                   RET
                   ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
                   ;
                   ; 'E' COMMAND - PUNCH HEXADECIMAL END-OF-FILE
                   ;
                   ; THIS ROUTINE PRODUCES A TERMINATION RECORD WHICH PROPERLY
                   ; COMPLETES A HEXADECIMAL FILE CREATED BY 'W' COMMANDS.
                   ; IT EXPECTS ONE HEXADECIMAL PARAMETER, WHICH IT INTERPRETS AS THE
                   ; START ADDRESS TO BE LOADED INTO THE USER'S PROGRAM COUNTER (LOCATED
                   ; IN EXIT TEMPLATE) ON A SUBSEQUENT 'R' COMMAND; THIS START ADDRESS
                   ; WILL REPLACE THE STORED VALUE OF THE USER'S PROGRAM COUNTER ONLY
                   ; IF THE START ADDRESS IS NONZERO.
                   ;
                   EOF:
 F95F 0D                   DCR     C               ; C:=1; GET ONE PARAMETER
 F960 CD39FE               CALL    EXPR            ; PUT <START ADDRESS> ON TOP OF STACK
 F963 CDE5FC               CALL    POC             ; OUTPUT RECORD MARK (':')
 F966 3A                   DB      ':'
 F967 AF                   XRA     A               ; ZERO CHECKSUM
 F968 57                   MOV     D,A             ; D := 0; A := 0
 F969 CDAFFE               CALL    PBYTE           ; OUTPUT A RECORD LENGTH OF ZERO
 F96C E1                   POP     H               ; RETRIEVE START ADDRESS
 F96D CDAAFE               CALL    PADR            ; OUTPUT IT AS THE LOAD ADDRESS
 F970 3E01                 MVI     A,1             ; RECORD TYPE = 1
 F972 CDAFFE               CALL    PBYTE           ; OUTPUT RECORD TYPE
 F975 AF                   XRA     A               ; A := 0
 F976 92                   SUB     D               ; D CONTAINS RUNNING CHECKSUM
 F977 CDAFFE               CALL    PBYTE           ; OUTPUT CHECKSUM := -D
 F97A C309FA               JMP     NU0             ; PUNCH TRAILER AND RETURN
                   ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
                   ;
                   ; 'F' COMMAND - FILL RAM WITH 8-BIT CONSTANT
                   ;
 LOC  OBJ           SOURCE STATEMENT

                    ; THIS ROUTINE EXPECTS THREE HEXADECIMAL PARAMETERS, THE
                    ; FIRST AND SECOND (16 BITS) ARE INTERPRETED AS THE BOUNDS
                    ; OF A MEMORY AREA TO BE INITIALIZED TO A CONSTANT VALUE,
                    ; THE THIRD PARAMETER (8 BITS) IS THAT VALUE.
                    FILL:
 F97D ØC                    INR     C               ; C:=3; GET 3 PARAMETERS
 F97E CD39FE                CALL    EXPR
 F981 C1                    POP     B               ; C := 8-BIT CONSTANT
 F982 D1                    POP     D               ; DE := HIGH ADDRESS
 F983 E1                    POP     H               ; HL := LOW ADDRESS
                    FIØ:
 F984 71                    MOV     M,C             ; STORE CONSTANT IN MEMORY
 F985 CD4CFE                CALL    HILO            ; TEST FOR COMPLETION
 F988 D284F9                JNC     FIØ             ; CONTINUE LOOPING
 F98B C9                    RET                     ; GO BACK TO START
                    ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
                    ;
                    ; 'G' COMMAND - GO TO <ADDRESS>, OPTIONALLY SET BREAKPOINT(S)
                    ;
                    ; THE G COMMAND IS USED FOR TRANSFERRING CONTROL FROM THE
                    ; MONITOR TO A USER PROGRAM.  IT HAS SEVERAL MODES OF
                    ; OPERATION.
                    ; IF ONE HEXADECIMAL PARAMETER IS ENTERED, IT IS INTERPRETED
                    ; AS THE ENTRY POINT OF THE USER PROGRAM AND A TRANSFER TO
                    ; THAT LOCATION IS EXECUTED.
                    ; IF ADDITIONAL (UP TO 2) PARAMETERS ARE ENTERED, THESE ARE
                    ; CONSIDERED 'BREAKPOINTS', I.E., LOCATIONS WHERE
                    ; CONTROL IS TO BE RETURNED TO THE MONITOR WHEN THEY ARE
                    ; ENCOUNTERED IN COURSE OF EXECUTING THE USER PROGRAM.
                    ; IF THE FIRST PARAMETER IS NOT ENTERED, THE STORED VALUE
                    ; OF THE USER'S PROGRAM COUNTER (REGISTER P) IS USED AS
                    ; THE USER PROGRAM ENTRY POINT.
                    ;
                    ; THIS COMMAND WORKS IN THE FOLLOWING MANNER:
                    ;   1. IT FINDS THE EXIT CODE IN TOP OF RAM AND PLACES THIS ADDRESS IN THE
                    ;      MONITOR'S STACK, REPLACING THE RETURN ADDRESS TO ENTRY POINT START
                    ;      THAT WAS PLACED THERE BY THE MAIN COMMAND LOOP.
                    ;   2. IF THERE IS NO FURTHER INPUT (I.E. ONLY <CR>) THEN BY EXECUTING A
                    ;      RET, WE CAUSE EXECUTION OF THE EXIT CODE, WHICH CONTAINS A JUMP TO
                    ;      A) A DUMMY ADDRESS (IF IMPROPER USE OF COMMAND), B) THE PROGRAM
                    ;      COUNTER FROM WHEN THE USER PROGRAM WAS INTERRUPTED OR BREAKPOINT
                    ;      WAS ENCOUNTERED.
                    ;   3. IF THERE IS A START ADDRESS SPECIFIED, THIS VALUE IS STORED OVER
                    ;      THAT PART OF THE EXIT CODE WHICH CONTAINS THE JMP INSTRUCTION.
                    ;      IF THERE IS NO FURTHER INPUT, A RET IS EXECUTED AND THE EXIT
                    ;      CODE IS EXECUTED.
                    ;   4. IF TRAPS (BREAKPOINTS) ARE TO BE SET, THEN THEY ARE READ IN AND PLACED
                    ;      ON THE MONITOR STACK.  THEY ARE THEN STORED IN THE PROPER SECTION OF
                    ;      THE EXIT TEMPLATE. ALSO, IN THE USER'S PROGRAM THE INSTRUCTION SPECIFIED
                    ;      BY THE BREAKPOINT ADDRESS IS SAVED IN THE EXIT TEMPLATE AND REPLACED
                    ;      WITH A RST Ø INSTRUCTION.
                    ;   5. THE EXIT CODE IS EXECUTED AND CONTROL IS PASSED TO THE USER PROGRAM.
                    GOTO:
 F98C 2AØ4ØØ                LHLD    MEMTOP
 F98F 2ED2                  MVI     L,EXIT AND ØFFH ; HL NOW POINTS TO EXIT CODE IN TOP OF RAM
 LOC  OBJ         LINE        SOURCE STATEMENT

 F991 E3          1336         XHTL                          ; REPLACE THE START RETURN ADDRESS IN THE
                  1337                                       ;    STACK (PUSHED BY MAIN COMMAND LOOP) WITH
                  1338                                       ;    THIS EXIT CODE ADDRESS SO THAT WHEN THE
                  1339                                       ;    G COMMAND DOES A RETURN, THE EXIT CODE
                  134Ø                                       ;    WILL BE EXECUTED INSTEAD OF THE MAIN
                  1341                                       ;    COMMAND LOOP.
 F992 CDC5FE      1342         CALL    PCHK                  ; GET A CHARACTER, SET Z,C
 F995 CAA4F9      1343         JZ      GOØ                   ; IF ' ', ',', OR <CR>: JUMP, DON'T CHANGE PC
 F998 CD7AFE      1344         CALL    PAØ                   ; GET NEW PC VALUE
 F99B EB          1345         XCHG                          ; DE = NEW PC
 F99C 2AØ4ØØ      1346         LHLD    MEMTOP
 F99F 2EE1        1347         MVI     L,PLOC AND ØFFH ; HL NOW POINTS TO PLOC IN EXIT CODE IN TOP OF RAM
 F9A1 72          1348         MOV     M,D                   ; STORE MSB OF MODIFIED PC IN EXIT CODE IN RAM
 F9A2 2B          1349         DCX     H
 F9A3 73          135Ø         MOV     M,E                   ; STORE LSB OF MODIFIED PC IN EXIT CODE IN RAM
                  1351 GOØ:
 F9A4 DAD1F9      1352         JC      GO4                   ; JUMP IF <CR> (NO TRAPS TO BE SET)
 F9A7 11Ø2ØØ      1353         LXI     D,2                   ; SET COUNTER(S), D=Ø, E=2
                  1354 GO1:
 F9AA CDDEFC      1355         CALL    COMC                  ; ISSUE A PROMPT FOR A TRAP
 F9AD 2D          1356         DB      '-'
 F9AE CD74FE      1357         CALL    PARAM                 ; GET A TRAP
 F9B1 E5          1358         PUSH    H                     ; STACK IT
 F9B2 14          1359         INR     D                     ; UP 1 COUNTER
 F9B3 DABAF9      136Ø         JC      GO2                   ; TERMINATE IF CR ENTERED
 F9B6 1D          1361         DCR     E                     ; DOWN THE OTHER
 F9B7 C2AAF9      1362         JNZ     GO1                   ; GET ONE MORE TRAP
                  1363 GO2:                                  ; D CONTAINS HOW MANY TRAPS (1 OR 2)
 F9BA D247F8      1364         JNC     ERROR                 ; LAST TRAP NOT FOLLOWED BY CR
 F9BD 2AØ4ØØ      1365         LHLD    MEMTOP
 F9CØ 2EE2        1366         MVI     L,TLOC AND ØFFH ; HL NOW POINTS TO TLOC (BEGINNING OF TRAP
                  1367                                       ;    AREA) IN EXIT TEMPLATE IN TOP OF RAM
                  1368 GO3:                                  ; BC CONTAINS THE USER SPECIFIED TRAP ADDRESS
 F9C2 C1          1369         POP     B                     ; GET A TRAP (BREAKPOINT) ADDRESS
 F9C3 71          137Ø         MOV     M,C                   ; STORE LSB OF TRAP ADDRESS INTO TRAP AREA
 F9C4 23          1371         INX     H
 F9C5 7Ø          1372         MOV     M,B                   ; STORE MSB OF TRAP ADDRESS INTO TRAP AREA
 F9C6 23          1373         INX     H
 F9C7 ØA          1374         LDAX    B                     ; FETCH OPCODE BYTE
 F9C8 77          1375         MOV     M,A                   ; PUT IN TRAP AREA
 F9C9 23          1376         INX     H
 F9CA 3EC7        1377         MVI     A,(RST Ø)             ; REPLACE THE USER'S OPCODE IN USER PROGRAM
 F9CC Ø2          1378         STAX    B                     ;    WITH A RST Ø
 F9CD 15          1379         DCR     D
 F9CE C2C2F9      138Ø         JNZ     GO3                   ; DO SAME THING AGAIN FOR 2ND BREAKPOINT
                  1381 GO4:
 F9D1 CDFEFD      1382         CALL    CRLF
 F9D4 C9          1383         RET                           ; EXECUTE MONITOR EXIT CODE, RETURNING TO
                  1384                                       ;    USER CODE
                  1385 ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
                  1386 ; 'H' COMMAND - COMPUTE HEXADECIMAL SUM AND DIFFERENCE
                  1387 ;
                  1388 ; THIS ROUTINE EXPECTS TWO HEXADECIMAL PARAMETERS.
                  1389 ; IT COMPUTES THE SUM AND DIFFERENCE OF THE TWO VALUES
                  139Ø ; AND DISPLAYS THEM ON THE LOCAL CONSOLE DEVICE AS FOLLOWS:
```

```
LOC  OBJ          SOURCE STATEMENT

                  ; <P1+P2> <P1-P2>
                  HEXN:
F9D5 CD39FE               CALL    EXPR            ; GET TWO NUMBERS
F9D8 CDFEFD               CALL    CRLF
F9DB D1                   POP     D               ; DE CONTAINS P2
F9DC E1                   POP     H
F9DD E5                   PUSH    H               ; HL CONTAINS P1
F9DE 19                   DAD     D               ; HL := HL + DE := P1 + P2
F9DF CD56FE               CALL    LADR            ; DISPLAY SUM
F9E2 CD93FC               CALL    BLK             ; TYPE A SPACE
F9E5 E1                   POP     H               ; HL CONTAINS P1 AGAIN
F9E6 7D                   MOV     A,L             ; COMPUTE HL-DE
F9E7 93                   SUB     E               ; A := LSB OF P1 - LSB OF P2
F9E8 6F                   MOV     L,A             ; A := LSB OF (P1 - P2)
F9E9 7C                   MOV     A,H
F9EA 9A                   SBB     D               ; A := MSB OF P1 - MSB OF P2 WITH CARRY
F9EB 67                   MOV     H,A             ; H := MSB OF (P1 -P2)
F9EC CD56FE               CALL    LADR            ; DISPLAY DIFFERENCE
F9EF C9                   RET
                  ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
                  ;
                  ; 'M' COMMAND - MOVE A BLOCK OF MEMORY
                  ;
                  ; THIS ROUTINE EXPECTS THREE HEXADECIMAL PARAMETERS FROM THE
                  ; LOCAL CONSOLE. THE FIRST AND SECOND PARAMETERS ARE THE BOUNDS OF
                  ; THE MEMORY AREA TO BE MOVED, THE THIRD PARAMETER IS THE
                  ; STARTING ADDRESS OF THE DESTINATION AREA.
                  MOVE:
F9FØ ØC                   INR     C               ; GET THREE ADDRESSES
F9F1 CD39FE               CALL    EXPR
F9F4 C1                   POP     B               ; DESTINATION ADDRESS
F9F5 D1                   POP     D               ; SOURCE END ADDRESS
F9F6 E1                   POP     H               ; SOURCE START ADDRESS
                  MVØ:
F9F7 7E                   MOV     A,M             ; GET A DATA BYTE
F9F8 Ø2                   STAX    B               ; STORE AT DESTINATION
F9F9 Ø3                   INX     B               ; MOVE DESTINATION POINTER
F9FA CD4CFE               CALL    HILO            ; TEST FOR COMPLETION
F9FD D2F7F9               JNC     MVØ
FAØØ C9                   RET
                  ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
                  ;
                  ; 'N' COMMAND - PUNCH NULL CHARACTERS FOR TAPE LEADER/TRAILER
                  ;
                  ; THIS ROUTINE PUNCHES 6Ø NULL CHARACTERS ON THE DEVICE ASSIGNED
                  ; AS THE PUNCH.  IT IS ENTERED VIA A JUMP TO ENTRY POINT NUØ
                  ; FROM THE 'E' COMMAND AS WELL AS BEING INVOKED BY
                  ; THE 'N' COMMAND.
                  NULL:
FAØ1 CD61FF               CALL    TI              ; REQUIRE CR
FAØ4 FEØD                 CPI     CR
FAØ6 C247F8               JNZ     ERROR
                  NUØ:
FAØ9 Ø63C                 MVI     B,6Ø            ; SET TO PUNCH 6Ø NULLS
                  NLEADX:
```

```
 LOC  OBJ         LINE        SOURCE STATEMENT

 FAØB CDE5FC      1446         CALL    POC              ; PUNCH ONE ASCII NULL CHARACTER (=ØØH)
 FAØE ØØ          1447         DB      Ø
 FAØF Ø5          1448         DCR     B
 FA1Ø C2ØBFA      1449         JNZ     NLEADX           ; DO IT FOR 6Ø TIMES
 FA13 C9          145Ø         RET
                  1451 ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
                  1452 ;
                  1453 ; 'Q' COMMAND - I/O SYSTEM STATUS QUERY
                  1454 ;
                  1455 ; THIS COMMAND IS INVOKED BY TYPING THE LETTER Q.  THIS
                  1456 ; COMMAND PRODUCES A LISTING OF LOGICAL I/O DEVICES AND
                  1457 ; THEIR CORRESPONDING PHYSICAL DEVICE ASSIGNMENTS.  THE
                  1458 ; DATA DISPLAYED IS EQUIVALENT TO THE CURRENT VALUE OF IOBYT.
                  1459 QUERY:
 FA14 CD61FF      146Ø         CALL    TI               ; REQUIRE CR
 FA17 FEØD        1461         CPI     CR
 FA19 C247F8      1462         JNZ     ERROR
 FA1C Ø6Ø4        1463         MVI     B,4              ; SET UP OUTER LOOP COUNTER.
                  1464                                  ;    THERE ARE 4 LOGICAL DEVICES.
 FA1E 21Ø3F9      1465         LXI     H,LTBL           ; POINT HL AT LOGICAL DEVICE TABLE.
                  1466 QØ:                              ; OUTER LOOP
 FA21 CDFEFD      1467         CALL    CRLF             ; START A NEW LINE.
 FA24 4E          1468         MOV     C,M              ; DISPLAY LOGICAL DEVICE IDENTIFIER.
 FA25 CD95FC      1469         CALL    COM
 FA28 CDDEFC      147Ø         CALL    COMC             ; DISPLAY '='.
 FA2B 3D          1471         DB      '='
 FA2C 23          1472         INX     H                ; POINT AT MASK FOR LOGICAL DEVICE.
 FA2D 7E          1473         MOV     A,M              ; FETCH MASK.
 FA2E 2F          1474         CMA                      ; INVERT IT
 FA2F 4F          1475         MOV     C,A              ; PUT IN C
 FA3Ø 23          1476         INX     H                ; POINT AT PHYSICAL DEVICE TABLE
 FA31 5E          1477         MOV     E,M              ; ADDRESS OF SUBORDINATE
 FA32 23          1478         INX     H                ; TABLE
 FA33 56          1479         MOV     D,M
 FA34 23          148Ø         INX     H
 FA35 EB          1481         XCHG                     ; HL <- PHYSICAL DEVICE TABLE
 FA36 3AØ3ØØ      1482         LDA     IOBYT
 FA39 A1          1483         ANA     C                ; PHYSICAL SELECTION
 FA3A C5          1484         PUSH    B                ; SAVE OUTER LOOP COUNTER
 FA3B Ø6Ø4        1485         MVI     B,4              ; SET UP INNER LOOP COUNTER
                  1486 Q1:                              ; INNER LOOP
 FA3D 4E          1487         MOV     C,M              ; GET PHYSICAL DEVICE IDENTIFIER
 FA3E 23          1488         INX     H
 FA3F BE          1489         CMP     M                ; TEST FOR EQUALITY
 FA4Ø CA48FA      149Ø         JZ      Q2
 FA43 23          1491         INX     H                ; POINT AT NEXT ENTRY
 FA44 Ø5          1492         DCR     B                ; DECREMENT INNER LOOP
 FA45 C23DFA      1493         JNZ     Q1
                  1494 Q2:
 FA48 CD95FC      1495         CALL    COM              ; DISPLAY PHYSICAL DEVICE
 FA4B EB          1496         XCHG                     ; POINT AT MASTER TABLE
 FA4C C1          1497         POP     B
 FA4D Ø5          1498         DCR     B                ; DECREMENT OUTER LOOP
 FA4E C221FA      1499         JNZ     QØ
 FA51 C9          15ØØ         RET
 LOC  OBJ         LINE        SOURCE STATEMENT

                        ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
                        ;
                        ; 'R' COMMAND - READ HEXADECIMAL FILE
                        ;
                        ; THIS ROUTINE READS A HEXADECIMAL FILE FROM THE ASSIGNED
                        ; READER DEVICE AND LOADS IT INTO MEMORY.  ONE HEXADECIMAL
                        ; PARAMETER IS EXPECTED.  THIS PARAMETER IS A BASE ADDRESS
                        ; TO BE ADDED TO THE MEMORY ADDRESS OF EACH DATA BYTE ENCOUNTERED.
                        ; IN THIS WAY, HEXADECIMAL FILES MAY BE LOADED INTO MEMORY
                        ; IN AREAS OTHER THAN THAT FOR WHICH THEY WERE ASSEMBLED OR COMPILED.
                        ; ALL RECORDS READ ARE CHECKSUMMED AND COMPARED AGAINST THE
                        ; CHECKSUM IN THE RECORD. IF A CHECKSUM ERROR (OR TAPE READ ERROR)
                        ; OCCURS, THE ROUTINE TAKES AN ERROR EXIT.  NORMAL LOADING IS
                        ; TERMINATED WHEN AN EOF RECORD IS ENCOUNTERED. THE ADDRESS
                        ; GIVEN WHEN THE EOF RECORD WAS CREATED (VIA THE 'E' COMMAND) REPLACES
                        ; THE USER'S STORED PC VALUE ONLY IF THE ADDRESS WAS NONZERO.
                        ; A TRANSFER TO THE PROGRAM MAY THEN BE ACCOMPLISHED BY A 'G<CR>'.
                        READ:
 FA52 0D                        DCR     C                ; GET ONE ADDRESS; C := 1
 FA53 CD39FE                    CALL    EXPR             ; GET THE HEX BASE ADDRESS
 FA56 CDFEFD                    CALL    CRLF             ; OUTPUT A <CR>,<LF>
                        RED0:
 FA59 CD58FF                    CALL    RIX              ; GET AN ASCII CHARACTER FROM THE READER
 FA5C FE3A                      CPI     ':'              ; IS IT A START OF RECORD MARK (':')?
 FA5E C259FA                    JNZ     RED0             ; LOOP UNTIL WE FIND SUCH A RECORD MARK
 FA61 AF                        XRA     A
 FA62 57                        MOV     D,A              ; D WILL CONTAIN THE CHECKSUM; INITIALIZE TO 0
 FA63 CDDBFD                    CALL    BYTE             ; READ 2 ASCII CHAR REPRESENTING THE RECORD
                                                         ;    LENGTH AND DECODE THEM INTO 8 BITS BINARY
                                                         ;    STORING THE RESULT IN A-REG
 FA66 CA9EFA                    JZ      RED3             ; JUMP IF ZERO RECORD LENGTH BECAUSE THIS
                                                         ;    MEANS IT'S AN EOF RECORD SO WE'RE DONE
 FA69 5F                        MOV     E,A              ; E := RECORD LENGTH
 FA6A CDDBFD                    CALL    BYTE             ; GET MSB OF LOAD ADDRESS
 FA6D 67                        MOV     H,A              ; H := MSB OF LOAD ADDRESS
 FA6E CDDBFD                    CALL    BYTE             ; GET LSB OF LOAD ADDRESS
 FA71 6F                        MOV     L,A              ; L := LSB OF LOAD ADDRESS
 FA72 CDDBFD                    CALL    BYTE             ; GET RECORD TYPE AND IGNORE IT
 FA75 4B                        MOV     C,E              ; C := RECORD LENGTH
 FA76 E5                        PUSH    H                ; STORE LOAD ADDRESS ON THE STACK
 FA77 2100FF                    LXI     H,-256           ; COMPUTE BUFFER POINTER
 FA7A 39                        DAD     SP               ; HL NOW POINTS TO THAT PART OF THE MONITOR
                                                         ;    STACK ONE PAGE (256 BYTES) BELOW WHERE
                                                         ;    THE SP IS CURRENTLY POINTING
                                                         ; WE WILL NOW READ DATA FROM THE FILE RECORD
                                                         ; AND STORE THEM TEMPORARILY IN THE MONITOR'S
                                                         ; STACK STARTING FROM A LOW MEMORY ADDRESS AND
                                                         ; MOVING TOWARD A HIGHER MEMORY ADDRESS (REVERSE
                                                         ; OF USUAL PROCEDURE WHERE STACK GROWS DOWN)
                        RED1:
 FA7B CDDBFD                    CALL    BYTE             ; READ DATA; NOTE: 8 BITS OF MEMORY (DATA)
                                                         ;    IS REPRESENTED AS 2 HEX CHAR AND EACH HEX
                                                         ;    HEX CHAR IS REPRESENTED AS ONE 8 BIT ASCII CHAR
 FA7E 77                        MOV     M,A              ; PUT DATA IN MONITOR BUFFER
 FA7F 23                        INX     H                ; MOVE "UP" THE STACK
```

```
  LOC  OBJ         LINE        SOURCE STATEMENT

  FA80 1D                      DCR     E                ; DECREMENT RECORD LENGTH COUNT
  FA81 C27BFA                  JNZ     RED1             ; LOOP UNTIL RECORD LENGTH COUNTER IS 0
  FA84 CDDBFD                  CALL    BYTE             ; READ THE CHECKSUM RECORD FRAME --- PRIOR TO
                                                        ;    CALL TO BYTE, D-REG CONTAINED SUM OF DATA
                                                        ;    RECORDS. THE CHECKSUM FRAME SHOULD CONTAIN
                                                        ;    THE NEGATIVE OF THIS SUM. BYTE ADDS D AND A
                                                        ;    TOGETHER AND SETS THE ZERO BIT IF D = (-A)
  FA87 C247F8                  JNZ     ERROR            ; CHECKSUM ERROR
  FA8A D1                      POP     D                ; DE = LOAD ADDRESS; STACK ENTRY POINTED TO BY SP
                                                        ;    NOW CONTAINS BASE (BIAS) ADDRESS
  FA8B E3                      XTHL                     ; HL = BIAS ADDRESS; CONTENTS OF STACK ENTRY
                                                        ;    POINTED TO BY SP NOW IS ADDRESS ONE ABOVE
                                                        ;    WHERE LAST DATA IS STORED IN MONITOR STACK
  FA8C EB                      XCHG                     ; DE = BIAS ADDRESS, HL = LOAD ADDRESS
  FA8D 19                      DAD     D                ; HL = BIAS + LA
  FA8E 0600                    MVI     B,0              ; BC = RECORD LENGTH (RL)
  FA90 09                      DAD     B                ; HL = BIAS + LA + RL
  FA91 EB                      XCHG                     ; DE = BIAS + LA + RL, HL = BIAS
  FA92 E3                      XTHL                     ; HL POINTS TO ADDRESS 1 GREATER THAN WHERE LAST
                                                        ;    DATA IS STORED IN MONITOR STACK
                          ;------------------------------
                          RED2:                         ; LOAD INTO PROPER AREA IN RAM BUT IN
                                                        ;    REVERSE ORDER
  FA93 2B                      DCX     H                ; DECREMENT STACK BUFFER POINTER
  FA94 7E                      MOV     A,M              ; A := DATA
  FA95 1B                      DCX     D                ; DECREMENT MEMORY POINTER
  FA96 12                      STAX    D                ; PUT DATA IN DESIGNATED ADDRESS
  FA97 0D                      DCR     C                ; KEEP DOING THIS UNTIL RECORD LENGTH
  FA98 C293FA                  JNZ     RED2             ;    COUNT IS EXHAUSTED
  FA9B C359FA                  JMP     RED0             ; DONE WITH ONE RECORD, GO GET ANOTHER
                          ;-------------------------------
                          RED3:                         ; EOF RECORD - ENTIRE FILE HAS BEEN READ IN
  FA9E C5                      PUSH    B                ; SAVE B,C
  FA9F CDDBFD                  CALL    BYTE             ; GET MSB OF LOAD ADDRESS OF EOF RECORD ---
                                                        ;    THIS IS THE <START ADDRESS> SPECIFIED IN
                                                        ;    THE 'E' COMMAND. IF IT IS ZERO, DO NOT
                                                        ;    MODIFY THE USER'S STORED PC IN EXIT TEMPLATE
  FAA2 47                      MOV     B,A              ; B := MSB OF START ADDRESS
  FAA3 CDDBFD                  CALL    BYTE             ; GET LSB OF START ADDRESS
  FAA6 4F                      MOV     C,A              ; C := LSB OF START ADDRESS
  FAA7 B0                      ORA     B                ; SEE IF START ADDRESS IS 0000
  FAA8 CAB3FA                  JZ      RED4             ; JUMP IF IT IS (DON'T SET NEW PC)
  FAAB 2A0400                  LHLD    MEMTOP
  FAAE 2EE1                    MVI     L,PLOC AND 0FFH  ; HL POINTS TO PLOC IN EXIT CODE IN TOP OF RAM
  FAB0 70                      MOV     M,B              ; STORE MSB OF START ADDRESS
  FAB1 2B                      DCX     H                ; HL POINTS TO PLOC - 1 OF EXIT CODE
  FAB2 71                      MOV     M,C              ; STORE LSB OF START ADDRESS
                          RED4:                         ; FINISH PROCESSING EOF RECORD
  FAB3 C1                      POP     B                ; RESTORE B,C
  FAB4 CDDBFD                  CALL    BYTE             ; GET RECORD TYPE AND IGNORE IT
  FAB7 CDDBFD                  CALL    BYTE             ; GET CHECKSUM
  FABA C247F8                  JNZ     ERROR            ; JUMP IF CHECKSUM ERROR
  FABD E1                      POP     H                ; CUT BACK STACK POINTER
  FABE C9                      RET
                          ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
```

```
 LOC  OBJ          SOURCE STATEMENT

                   ;
                   ; 'S' COMMAND - SUBSTITUTE MEMORY
                   ;
                   ; THIS ROUTINE EXPECTS ONE PARAMETER FROM THE LOCAL CONSOLE, FOLLOWED
                   ; BY A SPACE. THE PARAMETER IS INTERPRETED AS A MEMORY LOCATION
                   ; AND THE ROUTINE WILL DISPLAY THE CONTENTS OF THAT LOCATION,
                   ; FOLLOWED BY A DASH (-).  TO MODIFY MEMORY, TYPE IN THE NEW DATA
                   ; FOLLOWED BY A SPACE OR A CARRIAGE RETURN. IF NO MODIFICATION
                   ; OF THE LOCATION IS REQUIRED, TYPE ONLY A SPACE OR CARRIAGE RETURN.
                   ; IF A SPACE WAS LAST TYPED, THE NEXT MEMORY LOCATION WILL BE DISPLAYED
                   ; AND MODIFICATION OF IT IS ALLOWED.  IF A CARRIAGE RETURN WAS ENTERED,
                   ; THE COMMAND IS TERMINATED.
                   ;
                   SUBS:
 FABF CD74FE               CALL    PARAM           ; GET MEMORY ADDRESS
 FAC2 D8                   RC                      ; ONLY CR ENTERED SO RETURN TO MAIN COMMAND LOOP
                   SUØ:
 FAC3 7E                   MOV     A,M             ; HL HAS REQUESTED MEMORY ADDRESS
 FAC4 CD5BFE               CALL    LBYTE           ; DISPLAY CONTENTS OF THAT ADDRESS
 FAC7 CDDEFC               CALL    COMC            ; OUTPUT PROMPT CHARACTER
 FACA 2D                   DB      '-'
 FACB CDC5FE               CALL    PCHK
 FACE D8                   RC                      ; CR ENTERED, RETURN TO COMMAND MODE
 FACF CAD9FA               JZ      SU1             ; SPACE ENTERED, SPACE BY
 FAD2 EB                   XCHG                    ; SAVE MEMORY ADDRESS
 FAD3 CD7AFE               CALL    PAØ             ; GET NEW VALUE
 FAD6 EB                   XCHG                    ; E = VALUE
 FAD7 73                   MOV     M,E             ; STORE NEW VALUE
 FAD8 D8                   RC                      ; CR ENTERED AFTER VALUE, RETURN
                   SU1:
 FAD9 23                   INX     H               ; HL POINTS TO NEXT MEMORY LOCATION
 FADA C3C3FA               JMP     SUØ
                   ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
                   ;
                   ; 'W' COMMAND - WRITE HEXADECIMAL FILE
                   ;
                   ; THIS ROUTINE EXPECTS TWO HEXADECIMAL PARAMETERS WHICH ARE
                   ; INTERPRETED AS THE BOUNDS OF A MEMORY AREA TO BE ENCODED
                   ; INTO HEXADECIMAL FORMAT AND PUNCHED ON THE ASSIGNED PUNCH
                   ; DEVICE.
                   WRITE:
 FADD CD39FE               CALL    EXPR            ; GET ADDRESS RANGE
 FAEØ CDFEFD               CALL    CRLF            ; NEW LINE
 FAE3 D1                   POP     D               ; DE := HIGH ADDRESS
 FAE4 E1                   POP     H               ; HL := LOW ADDRESS
                   WRØ:
 FAE5 CDE5FC               CALL    POC             ; EMIT RECORD MARK
 FAE8 3A                   DB      ':'
 FAE9 Ø11ØØØ               LXI     B,16            ; INITIALIZE B := Ø, C := AH (DECIMAL 16)
                   ;--------------------------------
 FAEC E5                   PUSH    H               ; SAVE HL
                   WR1:
 FAED Ø4                   INR     B               ; INCREMENT RECORD LENGTH
 FAEE ØD                   DCR     C
 FAEF CAF8FA               JZ      WR2             ; TERMINATE ON COUNT OF 16 BYTES
 LOC  OBJ              SOURCE STATEMENT

 FAF2 CD4CFE               CALL    HILO            ; OR END OF RANGE
 FAF5 D2EDFA               JNC     WR1             ; WHICHEVER OCCURS FIRST
                   ;------------------------------
                   WR2:                            ; OUTPUT A DATA RECORD
 FAF8 E1                   POP     H               ; RESTORE HL := LOW ADDRESS
 FAF9 D5                   PUSH    D               ; SAVE HIGH ADDRESS
 FAFA 1600                 MVI     D,0             ; INITIALIZE CHECKSUM D := 0
 FAFC 78                   MOV     A,B             ; A := RECORD LENGTH
 FAFD CDAFFE               CALL    PBYTE           ; EMIT RECORD LENGTH
 FB00 CDAAFE               CALL    PADR            ; EMIT HL := LOW ADDRESS
 FB03 AF                   XRA     A
 FB04 CDAFFE               CALL    PBYTE           ; EMIT RECORD TYPE = 1
                   ;------------------------------
                   WR3:
 FB07 7E                   MOV     A,M             ; FETCH DATA
 FB08 CDAFFE               CALL    PBYTE           ; EMIT IT
 FB0B 23                   INX     H               ; INCREMENT MEMORY ADDRESS
 FB0C 05                   DCR     B               ; DECREMENT COUNT
 FB0D C207FB               JNZ     WR3             ; LOOP UNTIL ENTIRE RECORD HAS BEEN OUTPUT
 FB10 AF                   XRA     A
 FB11 92                   SUB     D               ; D CONTAINS RUNNING CHECKSUM
 FB12 CDAFFE               CALL    PBYTE           ; EMIT CHECKSUM := -D
 FB15 D1                   POP     D               ; RESTORE DE := HIGH ADDRESS
 FB16 2B                   DCX     H               ; BACKUP MEMORY POINTER
                                                   ; NOW PUNCH CR,LF --- IGNORED BY THE 'R'
                                                   ;    COMMAND BUT HANDY IF LISTING PUNCHED
                                                   ;    TAPE ON THE TTY
 FB17 CDE5FC               CALL    POC             ; PUNCH CARRIAGE RETURN
 FB1A 0D                   DB      CR
 FB1B CDE5FC               CALL    POC             ; PUNCH LINE FEED CHARACTER
 FB1E 0A                   DB      LF
 FB1F CD4CFE               CALL    HILO            ; TEST FOR TERMINATION
 FB22 D2E5FA               JNC     WR0             ; IF NOT DONE, FORM NEXT RECORD AND OUTPUT IT
 FB25 C9                   RET
                   ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
                   ;
                   ; 'X' COMMAND - EXAMINE AND MODIFY CPU REGISTERS
                   ;
                   ; THIS ROUTINE ALLOWS THE OPERATOR TO EXAMINE AND/OR MODIFY
                   ; THE CONTENTS OF THE USER PROGRAM'S REGISTERS.  THE REGISTER
                   ; VALUES WERE STORED AS A RESULT OF A PREVIOUS BREAKPOINT AND
                   ; WILL BE RESTORED TO THE USER PROGRAM DURING A SUBSEQUENT 'G'
                   ; COMMAND.
                   X:
 FB26 2181FB               LXI     H,ACTBL         ; POINT TO ACCESS TABLE
 FB29 CDC5FE               CALL    PCHK            ; GET REGISTER IDENTIFIER
 FB2C DA6AFB               JC      X5              ; IF CARRY = 1, CR ENTERED
 FB2F 0E0C                 MVI     C,NREGS
                   X0:
 FB31 BE                   CMP     M
 FB32 CA3FFB               JZ      X1              ; MATCHED REGISTER IDENTIFIER
 FB35 23                   INX     H               ; POINT TO NEXT TABLE ENTRY
 FB36 23                   INX     H
 FB37 23                   INX     H
 FB38 0D                   DCR     C               ; DECREMENT REGISTER COUNTER
 LOC  OBJ         LINE        SOURCE STATEMENT

 FB39 C231FB      1721         JNZ     XØ               ; TRY AGAIN
 FB3C C347F8      1722         JMP     ERROR            ; NOT IN TABLE, ERROR
                  1723 X1:
 FB3F CD93FC      1724         CALL    BLK
                  1725 X2:
 FB42 CD25FE      1726         CALL    DREG             ; DISPLAY THE REGISTER
 FB45 CDDEFC      1727         CALL    COMC
 FB48 2D          1728         DB      '-'              ; TYPE PROMPT
 FB49 CDC5FE      1729         CALL    PCHK             ; SKIP IF NULL ENTRY
 FB4C D8          173Ø         RC                       ; CR ENTERED, RETURN TO COMMAND MODE
 FB4D CA6ØFB      1731         JZ      X4
 FB5Ø E5          1732         PUSH    H                ; SAVE POINTER TO ACTBL
 FB51 C5          1733         PUSH    B                ; SAVE PRECISION
 FB52 CD7AFE      1734         CALL    PAØ              ; GET NEW REG VALUE
 FB55 7D          1735         MOV     A,L
 FB56 12          1736         STAX    D                ; STORE LSB IN REGISTER AREA
 FB57 F1          1737         POP     PSW              ; RETRIEVE PRECISION (A)
 FB58 B7          1738         ORA     A                ; SET SIGN
 FB59 FA5FFB      1739         JM      X3               ; 8 BITS ONLY
 FB5C 13          174Ø         INX     D
 FB5D 7C          1741         MOV     A,H
 FB5E 12          1742         STAX    D                ; STORE MSB IN REGISTER AREA
                  1743 X3:
 FB5F E1          1744         POP     H                ; RETRIEVE ACTBL POINTER
                  1745 X4:
 FB6Ø AF          1746         XRA     A
 FB61 B6          1747         ORA     M
 FB62 F8          1748         RM                       ; END OF TABLE, RETURN TO COMMAND MODE
 FB63 78          1749         MOV     A,B              ; TEST DELIMITER
 FB64 FEØD        175Ø         CPI     CR
 FB66 C8          1751         RZ                       ; CR ENTERED, RETURN TO COMMAND MODE
 FB67 C342FB      1752         JMP     X2
                  1753 ;       ---------------------------------------------------------------
                  1754 X5:                              ; DISPLAY ALL THE REGISTER VALUES
 FB6A CDFEFD      1755         CALL    CRLF
                  1756 X6:
 FB6D CD93FC      1757         CALL    BLK              ; OUTPUT A SPACE
 FB7Ø AF          1758         XRA     A                ; CLEAR A
 FB71 B6          1759         ORA     M                ; SET CONDITION CODES
 FB72 F8          176Ø         RM                       ; ALL DONE, RETURN TO COMMAND MODE
 FB73 4E          1761         MOV     C,M              ; C CONTAINS A REGISTER IDENTIFIER (A,B,C,D...)
 FB74 CD95FC      1762         CALL    COM              ; PRINT CHARACTER
 FB77 CDDEFC      1763         CALL    COMC             ; PRINT EQUAL SIGN
 FB7A 3D          1764         DB      '='
 FB7B CD25FE      1765         CALL    DREG             ; DISPLAY REGISTER CONTENTS
 FB7E C36DFB      1766         JMP     X6               ; CONTINUE
                  1767 ;
                  1768 ; TABLE FOR ACCESSING REGISTERS
                  1769 ; TABLE CONTAINS:
                  177Ø ;       (1) REGISTER IDENTIFIER
                  1771 ;       (2) LOCATION ON STORAGE PAGE
                  1772 ;       (3) PRECISION
                  1773 ;
                  1774 ACTBL:
 FB81 41          1775         DB      'A',    ALOC AND HMSK,  Ø
```

```
LOC  OBJ           SOURCE STATEMENT

FB82 CF
FB83 ØØ
FB84 42                    DB      'B',    BLOC AND HMSK,  Ø
FB85 CB
FB86 ØØ
FB87 43                    DB      'C',    CLOC AND HMSK,  Ø
FB88 CA
FB89 ØØ
FB8A 44                    DB      'D',    DLOC AND HMSK,  Ø
FB8B C9
FB8C ØØ
FB8D 45                    DB      'E',    ELOC AND HMSK,  Ø
FB8E C8
FB8F ØØ
FB9Ø 46                    DB      'F',    FLOC AND HMSK,  Ø
FB91 CE
FB92 ØØ
FB93 48                    DB      'H',    HLOC AND HMSK,  Ø
FB94 DD
FB95 ØØ
FB96 49                    DB      'I',    ILOC AND HMSK,  Ø
FB97 CD
FB98 ØØ
FB99 4C                    DB      'L',    LLOC AND HMSK,  Ø
FB9A DC
FB9B ØØ
FB9C 4D                    DB      'M',    HLOC AND HMSK,  1
FB9D DD
FB9E Ø1
FB9F 5Ø                    DB      'P',    PLOC AND HMSK,  1
FBAØ E1
FBA1 Ø1
FBA2 53                    DB      'S',    SLOC AND HMSK,  1
FBA3 D1
FBA4 Ø1
FBA5 FF                    DB      -1
ØØØC               NREGS   EQU     ($-ACTBL)/3     ; LENGTH OF ACCESS TABLE
                   ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
                   ;
                   ; 'Z' COMMAND - TRANSFER CONTROL TO DIAGNOSTIC PROGRAM IN PROM
                   ; THIS ROUTINE EXPECTS A '$' AT WHICH POINT IT WILL CALL THE DIAGNOSTIC PROGRAM.
                   Z:
FBA6 CD61FF                CALL    TI              ; GET A CHARACTER FROM THE CONSOLE
FBA9 FE24                  CPI     '$'             ; IS IT A '$'?
FBAB C247F8                JNZ     ERROR           ; ERROR IF IT ISN'T
FBAE CD61FF                CALL    TI              ; GET A CHARACTER FROM THE CONSOLE
FBB1 FEØD                  CPI     CR              ; EXPECT A CARRIAGE RETURN
FBB3 C247F8                JNZ     ERROR           ; ERROR IF IT ISN'T
FBB6 3EØC                  MVI     A,BTDGON                ; TURN ON THE BOOT/DIAGNOSTIC PROM
FBB8 D3FF                  OUT     CPUC
FBBA CDØØEB                CALL    DIAGMN          ; CALL THE DIAGNOSTIC PROGRAM
FBBD C9                    RET                     ; RETURN TO MAIN COMMAND LOOP
                   ;****************************************************************************
                   ;*                                                                          *
                   ;*  END OF MONITOR COMMANDS, BEGINNING OF I/O ROUTINES                      *
```

```
  LOC  OBJ         LINE        SOURCE STATEMENT

                    ;*                                                                               *
                    ;*********************************************************************************
                    ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                    ; 'CI' - EXTERNALLY REFERENCED ROUTINE                                           ;
                    ;        ENTERED VIA CALL FROM 'TI' ROUTINE                                      ;
                    ; PROCESS: LOCAL CONSOLE INPUT CODE                                              ;
                    ; INPUT:                                                                         ;
                    ; OUTPUT: CHARACTER RETURNED IN A-REG                                            ;
                    ; MODIFIED: A, FLAGS                                                             ;
                    ; STACK USAGE: 2 BYTES                                                           ;
                    ; EXPLANATION: BASED ON I/O STATUS BYTE (IOBYT), DECIDE IF CONSOLE INPUT         ;
                    ;    DEVICE IS TTY, CRT, BATCH, OR USER-DEFINED DEVICE.  IF IT IS TTY OR CRT     ;
                    ;    LOOP UNTIL READ, INPUT THE CHARACTER, THEN RETURN.  IF IT IS BATCH,         ;
                    ;    JUMP TO 'RI' ROUTINE. IF IT IS USER-DEFINED DEVICE, JUMP TO @USER.          ;
                    ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                    CI:                             ; LOCAL CONSOLE INPUT
  FBBE 3AØ3ØØ               LDA     IOBYT           ; GET STATUS BYTE
  FBC1 E6Ø3                 ANI     NOT CMSK        ; LOOK AT ONLY CONSOLE FIELD
  FBC3 C2DØFB               JNZ     CIØ             ; JUMP IF CONSOLE IS NOT TTY
                    ;-------------------------------
                    ; CONSOLE = TTY
                    TTYIN:
  FBC6 DBF5                 IN      TTYS            ; TTY STATUS PORT
  FBC8 E6Ø2                 ANI     RRDY            ; CHECK FOR RECEIVE BUFFER READY
  FBCA CAC6FB               JZ      TTYIN           ; LOOP UNTIL IT IS READY
  FBCD DBF4                 IN      TTYI            ; INPUT CHARACTER FROM TTY
  FBCF C9                   RET                     ; RETURN; CHARACTER IN A-REG
                    ;-------------------------------
                    ; CONSOLE = CRT, BATCH, OR USER-DEFINED
                    CIØ:
  FBDØ FEØ1                 CPI     CCRT            ; LOCAL CONSOLE = CRT?
  FBD2 C2FDFB               JNZ     CI4             ; JUMP IF CONSOLE IS NOT CRT
  FBD5 E5                   PUSH    H               ; SAVE HL
  FBD6 2AØ4ØØ               LHLD    MEMTOP
  FBD9 2ECC                 MVI     L,ILOC-1 AND ØFFH; HL NOW POINTS TO CONFIGURATION BYTE STORED
                                                    ;     IN EXIT TEMPLATE IN TOP PAGE OF RAM
  FBDB 7E                   MOV     A,M             ; A := CONFIGURATION BYTE
  FBDC E1                   POP     H               ; RESTORE HL
  FBDD ØF                   RRC                     ; ROTATE BIT Ø INTO CARRY BIT, THUS CARRY = 1
                                                    ;    MEANS RUNNING ON SYSTEM WITHOUT INTEGRATED
                                                    ;    CRT
  FBDE D2EBFB               JNC     CI2             ; JUMP IF INTEGRATED CRT IS PRESENT
                    ;--------------------------------
                    ; CONSOLE = SERIAL CRT
                    CI1:
  FBE1 DBF7                 IN      USCS            ; INPUT CRT STATUS
  FBE3 E6Ø2                 ANI     RRDY            ; CHECK FOR RECEIVER BUFFER READY
  FBE5 CAE1FB               JZ      CI1             ; LOOP UNTIL IT IS READY
  FBE8 DBF6                 IN      USCI            ; GET CHARACTER FROM THE CRT
  FBEA C9                   RET                     ; RETURN; CHARACTER IS IN A-REG
                    ;------------------------------
                    ; CONSOLE = INTEGRATED CRT
                    CI2:
  FBEB C5                   PUSH    B               ; SAVE B,C
                    CI3:
```

```
 LOC  OBJ            SOURCE STATEMENT

 FBEC Ø613                  MVI     B,KSTS           ; LOAD KEYBOARD STATUS COMMAND
 FBEE CD7FFF                CALL    IOCDR1           ; INPUT KEYBOARD STATUS FROM IOC
 FBF1 E6Ø1                  ANI     KRDY             ; IS THE KEYBOARD READY?
 FBF3 CAECFB                JZ      CI3              ; LOOP UNTIL IT IS
 FBF6 Ø612                  MVI     B,KEYC           ; LOAD INPUT DATA COMMAND
 FBF8 CD7FFF                CALL    IOCDR1           ; INPUT DATA FROM THE KEYBOARD
 FBFB C1                    POP     B                ; RESTORE B,C
 FBFC C9                    RET                      ; RETURN; CHARACTER IS IN A-REG
                     ;------------------------------
                     ; CONSOLE IS BATCH OR USER-DEFINED DEVICE
                     CI4:
 FBFD FEØ2                  CPI     BATCH
 FBFF CAØFFC                JZ      RI               ; BATCH MODE, INPUT = READER
 FCØ2 3EE8                  MVI     A,CILOC AND HMSK; USER DEFINE LOCAL CONSOLE INPUT
 FCØ4 C38CFC                JMP     @USER
                     ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                     ; 'BREAK' - ENTERED VIA CALLS FROM 'BLK','COM','LOM' ROUTINES
                     ; PROCESS: TEST FOR OPERATOR INTERRUPTION OF COMMAND (I.E. DID OPERATOR
                     ;          DEPRESS THE "BREAK" KEY)
                     ; INPUT:
                     ; OUTPUT:
                     ; MODIFIED: A,FLAGS
                     ; STACK USAGE: 4 BYTES
                     BREAK:
 FCØ7 CD44FD                CALL    CSTS             ; SEE IF A KEY WAS DEPRESSED
 FCØA B7                    ORA     A
 FCØB C8                    RZ                       ; NO CHARACTER READY
 FCØC C361FF                JMP     TI               ; GET THE CHARACTER
                     ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                     ; 'RI' - EXTERNALLY REFERENCED ROUTINE                                         ;
                     ;        ENTERED VIA CALLS FROM 'CI','RIX' ROUTINES                            ;
                     ; PROCESS: READER INPUT CODE                                                   ;
                     ; INPUT:                                                                       ;
                     ; OUTPUT: CARRY = Ø AND VALID CHARACTER IN A-REG, OTHERWISE                    ;
                     ;         CARRY = 1 AND INVALID DATA (ZEROES) IN A-REG                         ;
                     ; MODIFIED: A, FLAGS                                                           ;
                     ; STACK USAGE: 8 BYTES                                                         ;
                     ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                     RI:                                     ; READER INPUT
 FCØF E5                    PUSH    H                ; SAVE HL
 FC1Ø 3AØ3ØØ                LDA     IOBYT            ; GET STATUS BYTE
 FC13 E6ØC                  ANI     NOT RMSK         ; GET READER BITS
 FC15 C258FC                JNZ     RI5              ; JUMP IF READER IS NOT THE TTY
                     ;------------------------------
                     ; READER = TTY
 FC18 C5                    PUSH    B                ; SAVE BC
 FC19 3EØD                  MVI     A,DISABL         ; HOLD UP INTERRUPTS WHILE TAPE IS ADVANCING
 FC1B D3FF                  OUT     CPUC
 FC1D DBF4                  IN      TTYI             ; CLEAR RECEIVE BUFFER BY READING IN ANY
                                                     ;    DATA THAT MAY BE THERE
                     RIØ:
 FC1F DBF5                  IN      TTYS             ; READ IN USART STATUS
 FC21 E6Ø4                  ANI     TXBE             ; CHECK FOR TRANSMITTER BUFFER EMPTY
 FC23 CA1FFC                JZ      RIØ              ; TRY AGAIN IF NOT EMPTY
 FC26 3E27                  MVI     A,TADV           ; ADVANCE THE TAPE
```

```
  LOC  OBJ          SOURCE STATEMENT

  FC28 D3F5                OUT     TTYC            ; OUTPUT THE ADVANCE COMMAND
  FC2A 0628                MVI     B,RADCT         ; INITIALIZE TIMER FOR 45 MS.
                   RI1:
  FC2C CD1EFE              CALL    DELAY           ; DELAY FOR 1 MILLISECONDS
  FC2F 05                  DCR     B               ; DECREMENT TIMER
  FC30 C22CFC              JNZ     RI1             ; JUMP IF TIMER NOT EXPIRED
  FC33 3E25                MVI     A,COMD          ; STOP THE READER ADVANCE
  FC35 D3F5                OUT     TTYC            ; OUTPUT STOP COMMAND
  FC37 06FA                MVI     B,RTOCT         ; INITIALIZE TIMER FOR 250 MS.
                   RI2:
  FC39 DBF5                IN      TTYS            ; INPUT READER STATUS
  FC3B E602                ANI     RRDY            ; CHECK FOR RECEIVER BUFFER READY
  FC3D C24CFC              JNZ     RI4             ; YES - DATA IS READY
  FC40 CD1EFE              CALL    DELAY           ; DELAY 1 MS
  FC43 05                  DCR     B               ; DECREMENT TIMER
  FC44 C239FC              JNZ     RI2             ; JUMP IF TIMER NOT EXPIRED
                   RI3:
  FC47 AF                  XRA     A               ; ZERO A, RESET CARRY
  FC48 37                  STC                     ; SET CARRY INDICATING EOF
  FC49 C34FFC              JMP     RI4B
                   RI4:
  FC4C DBF4                IN      TTYI
  FC4E B7                  ORA     A               ; CLEAR CARRY
                   RI4B:
  FC4F F5                  PUSH    PSW             ; SAVE DATA
  FC50 3E05                MVI     A,ENABL         ; PERMIT INTERRUPTS TO GO THROUGH
  FC52 D3FF                OUT     CPUC
  FC54 F1                  POP     PSW
  FC55 C1                  POP     B               ; RESTORE BC
  FC56 E1                  POP     H
  FC57 C9                  RET                     ; RETURN
                   ;------------------------------
                   ; READER IS PTR, USER-DEV-1, OR USER-DEV-2
                   RI5:
  FC58 FE04                CPI     RPTR            ; IS READER THE PAPER TAPE READER?
  FC5A C282FC              JNZ     RI8             ; JUMP IF IT ISN'T
                   ;------------------------------
                   ; READER = PAPER TAPE READER
  FC5D C5                  PUSH    B               ; SAVE BC
  FC5E 0650                MVI     B,RDRC OR PTRADV; LOAD READER ADVANCE 1 FRAME COMMAND
  FC60 CDE4FF              CALL    PIOCOM          ; OUTPUT THE COMMAND
  FC63 26FA                MVI     H,TOUT          ; 250 MS. TIMEOUT COUNTER
                   RI6:
  FC65 0611                MVI     B,RSTC          ; LOAD READER STATUS COMMAND
  FC67 CDB5FF              CALL    PIODR1          ; READ STATUS
  FC6A E601                ANI     PTRDY           ; IS THE READER READY?
  FC6C C279FC              JNZ     RI7             ; JUMP IF IT IS
  FC6F CD1EFE              CALL    DELAY           ; STALL FOR 1 MS.
  FC72 25                  DCR     H               ; 250 MS. TIMEOUT LOOP
  FC73 C265FC              JNZ     RI6
  FC76 C347FC              JMP     RI3             ; 250 MS. ARE UP; RETURN WITH CARRY = 1 (EOF COND)
                   RI7:                            ; THE PAPER TAPE READER IS READY
  FC79 0610                MVI     B,RDRC          ; LOAD READER COMMAND
  FC7B CDB5FF              CALL    PIODR1          ; READ A CHARACTER FROM THE PAPER TAPE READER
  FC7E B7                  ORA     A               ; RESET CARRY BIT
  LOC  OBJ         LINE        SOURCE STATEMENT

  FC7F C1          1972         POP     B                ; RESTORE BC
  FC8Ø E1          1973         POP     H
  FC81 C9          1974         RET                      ; RETURN SUCCESSFULLY WITH CARRY = Ø
                   1975 ;-------------------------------
                   1976 ; READER IS USER-DEFINED DEVICE 1 OR DEVICE 2
                   1977 RI8:
  FC82 E1          1978         POP     H
  FC83 FEØ8        1979         CPI     RUSE1
  FC85 3EEE        198Ø         MVI     A,R1LOC AND HMSK
  FC87 CA8CFC      1981         JZ      @USER            ; READER = USER-DEFINED DEVICE 1
  FC8A 3EF1        1982         MVI     A,R2LOC AND HMSK
                   1983 ;******JMP      @USER            ; READER = USER-DEFINED DEVICE 2
                   1984 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                   1985 ; '@USER' - ENTERED VIA JUMPS FROM 'LO','LOM','RI','CI','BLK','COM',
                   1986 ;           'CO','POC','PO','CSTS' ROUTINES
                   1987 ;          ENTERED VIA FALL-THRU FROM 'RI' ROUTINE
                   1988 ; PROCESS: USER-DEFINED I/O ENTRY POINT TRANSFER LOGIC
                   1989 ; INPUT: A-REG CONTAINS LSB ADDRESS PTR INTO USER-DEFINED ENTRY POINT TABLE (XTBL)
                   199Ø ; OUTPUT:
                   1991 ; MODIFIED:
                   1992 ; STACK USAGE:
                   1993 @USER:
  FC8C E5          1994         PUSH    H                ; SAVE HL, CREATE A STACK ENTRY
  FC8D 2AØ4ØØ      1995         LHLD    MEMTOP
  FC9Ø 6F          1996         MOV     L,A              ; HL NOW POINTS TO PROPER USER ENTRY POINT IN
                   1997                                  ;    XTBL IN EXIT TEMPLATE IN TOP PAGE OF RAM
  FC91 E3          1998         XTHL                     ; RESTORE HL; SP NOW POINTS TO USER ENTRY POINT
  FC92 C9          1999         RET                      ; BEGIN EXECUTING AT THIS ENTRY POINT
                   2ØØØ ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                   2ØØ1 ; 'CO' - EXTERNALLY REFERENCED ROUTINE                                         ;
                   2ØØ2 ;         ENTERED VIA CALL FROM 'TI' ROUTINE                                   ;
                   2ØØ3 ; 'BLK' - ENTERED VIA CALLS FROM 'H', 'X' COMMANDS                             ;
                   2ØØ4 ; 'COM' - ENTERED VIA CALLS FROM 'Q', 'X' COMMANDS                             ;
                   2ØØ5 ;          ENTERED VIA JUMPS FROM 'COMC', 'HXD' ROUTINES                       ;
                   2ØØ6 ; 'TTYOUT' - ENTERED VIA JUMPS FROM 'LOM','LO','POC','PO' ROUTINES             ;
                   2ØØ7 ; 'CRTOUT' - ENTERED VIA JUMPS FROM 'LOM','LO' ROUTINES                        ;
                   2ØØ8 ;             ENTERED VIA CALL FROM BOOTSTRAP PROGRAM                          ;
                   2ØØ9 ; PROCESS: LOCAL CONSOLE OUTPUT CODE                                           ;
                   2Ø1Ø ; INPUT: VALUE IN C-REG                                                        ;
                   2Ø11 ; OUTPUT: DATA OUTPUT TO APPROPRIATE DEVICE                                    ;
                   2Ø12 ; MODIFIED: A, FLAGS, C                                                        ;
                   2Ø13 ; STACK USAGE: 2 BYTES                                                         ;
                   2Ø14 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                   2Ø15 BLK:                             ; PRINT A BLANK
  FC93 ØE2Ø        2Ø16         MVI     C,' '
                   2Ø17 COM:                             ; LOCAL CONSOLE OUTPUT
  FC95 3AØ3ØØ      2Ø18         LDA     IOBYT            ; GET STATUS BYTE
  FC98 E6Ø3        2Ø19         ANI     NOT CMSK         ; LOOK ONLY AT CONSOLE FIELD
  FC9A FEØ2        2Ø2Ø         CPI     BATCH            ; IS CONSOLE = BATCH?
  FC9C C4Ø7FC      2Ø21         CNZ     BREAK            ; IF SO, DO NOT HONOR BREAK KEY IN BATCH MODE
                   2Ø22                                  ; IF IT ISN'T, THEN TEST FOR BREAK KEY
                   2Ø23 CO:                              ; EXTERNAL ENTRY POINT
  FC9F 3AØ3ØØ      2Ø24         LDA     IOBYT            ; GET STATUS BYTE
  FCA2 E6Ø3        2Ø25         ANI     NOT CMSK         ; LOOK ONLY AT CONSOLE FIELD
  FCA4 C2B2FC      2Ø26         JNZ     COØ              ; JUMP IF CONSOLE IS NOT TTY
```

```
LOC  OBJ           SOURCE STATEMENT

                   ;--------------------------------
                   ; CONSOLE = TTY
                   TTYOUT:
FCA7 DBF5                  IN      TTYS               ; LOCAL CONSOLE = TTY; GET TTY STATUS
FCA9 E601                  ANI     TRDY               ; IS IT READY?
FCAB CAA7FC                JZ      TTYOUT             ; LOOP UNTIL IT IS
FCAE 79                    MOV     A,C                ; LOAD CHARACTER TO BE OUTPUT
FCAF D3F4                  OUT     TTYO               ; OUTPUT CHARACTER
FCB1 C9                    RET                        ; RETURN
                   ;--------------------------------
                   ; CONSOLE IS CRT, BATCH, OR USER-DEFINED
                   CO0:
FCB2 FE02                  CPI     BATCH              ; CONSOLE = BATCH?
FCB4 CA1EFD                JZ      LO                 ; JUMP TO LIST OUTPUT IF IT IS
FCB7 FE01                  CPI     CCRT               ; LOCAL CONSOLE = CRT?
FCB9 3EEB                  MVI     A,COLOC AND 0FFH
FCBB C28CFC                JNZ     @USER              ; JUMP IF IT ISN'T, I.E. CONSOLE IS
                                                      ;    USER DEFINED LOCAL CONSOLE OUTPUT
                   ;--------------------------------
                   ; CONSOLE = CRT
                   CRTOUT:
FCBE E5                    PUSH    H                  ; SAVE H,L
FCBF 2A0400                LHLD    MEMTOP
FCC2 2ECC                  MVI     L,ILOC-1 AND 0FFH; HL NOW POINTS TO CONFIGURATION BYTE IN EXIT TEMPLATE
FCC4 7E                    MOV     A,M                ; A NOW CONTAINS THIS CONFIGURATION BYTE
FCC5 E1                    POP     H                  ; RESTORE H,L
FCC6 0F                    RRC                        ; ROTATE BIT 0 INTO CARRY BIT; THUS CARRY
                                                      ;    = 1 IF INTEGRATED CRT NOT PRESENT
FCC7 D2D5FC                JNC     CRTOT2             ; JUMP IF INTEGRATED CRT
                   ;--------------------------------
                   ; CONSOLE = SERIAL CRT
                   CRTOT1:                            ; INTELLEC WITH SERIALLY CONNECTED CRT
FCCA DBF7                  IN      USCS               ; INPUT CRT STATUS
FCCC E601                  ANI     TRDY               ; IS IT READY?
FCCE CACAFC                JZ      CRTOT1             ; LOOP UNTIL IT IS
FCD1 79                    MOV     A,C                ; MOVE CHARACTER TO BE OUTPUT TO C-REG
FCD2 D3F6                  OUT     USCO               ; OUTPUT IT TO THE CRT
FCD4 C9                    RET
                   ;--------------------------------
                   ; CONSOLE = INTEGRATED CRT
                   CRTOT2:                            ; INTELLEC WITH INTEGRATED CRT
FCD5 79                    MOV     A,C                ; MOVE CHARACTER TO BE OUTPUT TO A-REG
FCD6 C5                    PUSH    B                  ; SAVE B,C
                                                      ; CRT IS ALWAYS READY AND PRESENT - NO NEED
                                                      ; TO CHECK ITS STATUS
FCD7 0610                  MVI     B,CRTC             ; LOAD OUTPUT TO CRT COMMAND
FCD9 CD94FF                CALL    IOCDR2             ; OUTPUT DATA TO CRT
FCDC C1                    POP     B                  ; RESTORE B,C
FCDD C9                    RET
                   ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                   ; 'COMC' - ENTERED VIA CALLS FROM 'G','Q','S','X' COMMANDS AND 'ERROR',
                   ;         'START','CRLF','RESTART' ROUTINES
                   ; PROCESS: LOCAL CONSOLE OUTPUT OF CONSTANT DATA
                   ; INPUT: SP
                   ; OUTPUT: CONTENTS OF ADDRESS POINTED TO BY SP IS A RETURN ADDRESS TWO GREATER
```

```
LOC  OBJ        SOURCE STATEMENT

                ;         THAN THAT OF THE CALL COMC INSTRUCTION
                ; MODIFIED: C,H,L
                ; STACK USAGE: 2 BYTES
                COMC:
FCDE E3                 XTHL                    ; SINCE COMC WAS CALLED, SP NOW POINTS TO A STACK
                                                ;    ENTRY CONTAINING THE ADDRESS OF THE NEXT
                                                ;    INSTRUCTION, WHICH IN THIS CASE IS A DB.
                                                ;    HL NOW POINTS TO THIS DB.
FCDF 4E                 MOV     C,M             ; C NOW CONTAINS THE CHARACTER TO BE OUTPUT
FCEØ 23                 INX     H               ; BUMP RETURN ADDRESS,I.E. POINT IT BEYOND THE DB.
FCE1 E3                 XTHL                    ; SP MODIFIED, HL IS AS IT WAS ORIGINALLY
FCE2 C395FC             JMP     COM             ; OUTPUT IT
                ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                ; 'PO' - EXTERNALLY REFERENCED ROUTINE                                         ;
                ;        ENTERED VIA CALL FROM 'PBYTE' ROUTINE                                 ;
                ; 'POC' - ENTERED VIA CALLS FROM 'E','N','W' COMMANDS AND 'LEAD','PEOL'        ;
                ;        ROUTINES                                                              ;
                ; PROCESS: PUNCH OUTPUT CODE                                                   ;
                ; INPUT: VALUE IN C-REG                                                        ;
                ; OUTPUT:                                                                      ;
                ; MODIFIED: A, FLAGS, C                                                        ;
                ; STACK USAGE: 2 BYTES                                                         ;
                ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                POC:                                    ; PUNCH A CONSTANT
FCE5 E3                 XTHL                    ; SINCE POC ENTERED VIA CALL, SP POINTS TO STACK
                                                ;    ENTRY CONTAINING ADDRESS OF NEXT INSTRUCTION
                                                ;    WHICH IS A DB. HL NOW POINTS TO THIS DB.
FCE6 4E                 MOV     C,M             ; C NOW CONTAINS CHARACTER TO BE PUNCHED
FCE7 23                 INX     H               ; BUMP RETURN ADDRESS,I.E. POINT IT BEYOND DB
FCE8 E3                 XTHL                    ; SP MODIFIED, HL IS AS IT WAS ORIGINALLY
                PO:                             ; PUNCH OUTPUT
FCE9 3AØ3ØØ             LDA     IOBYT           ; GET STATUS BYTE
FCEC E63Ø               ANI     NOT PMSK        ; GET PUNCH BITS
FCEE CAA7FC             JZ      TTYOUT          ; JUMP IF PUNCH ISN'T TTY
FCF1 FE1Ø               CPI     PPTP            ; IS PUNCH = PAPER TAPE PUNCH?
FCF3 C2Ø8FD             JNZ     PO1             ; JUMP IF IT ISN'T
                ;-------------------------------
                ; PUNCH = PAPER TAPE PUNCH
FCF6 C5                 PUSH    B               ; SAVE BC
                POØ:                            ; PUNCH = PTP
FCF7 Ø613               MVI     B,PSTC          ; LOAD PUNCH STATUS COMMAND
FCF9 CDB5FF             CALL    PIODR1          ; READ STATUS
FCFC E6Ø1               ANI     PTPRY           ; IS THE PUNCH READY?
FCFE CAF7FC             JZ      POØ             ; LOOP UNTIL READY
FDØ1 Ø612               MVI     B,PUNC          ; LOAD PUNCH OUTPUT COMMAND
FDØ3 CDCEFF             CALL    PIODR3          ; OUTPUT CHARACTER THAT WAS IN C-REG
FDØ6 C1                 POP     B               ; RESTORE BC
FDØ7 C9                 RET
                ;-------------------------------
                ; PUNCH IS USER-DEFINED DEVICE 1 OR DEVICE 2
                PO1:
FDØ8 FE2Ø               CPI     PUSE1
FDØA 3EF4               MVI     A,P1LOC AND ØFFH
FDØC CA8CFC             JZ      @USER           ; PUNCH = USER DEFINED PUNCH 1
FDØF 3EF7               MVI     A,P2LOC AND ØFFH
```

```
LOC  OBJ          LINE        SOURCE STATEMENT

FD11 C38CFC               JMP     @USER           ; PUNCH = USER DEFINED PUNCH 2
                  ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                  ; 'LO' - EXTERNALLY REFERENCED ROUTINE                                         ;
                  ;        ENTERED VIA JUMPS FROM 'COM','CO','BLK' ROUTINES                      ;
                  ; 'LOM' - ENTERED VIA CALLS FROM 'D' COMMAND AND 'DBYTE','LCRLF' ROUTINES      ;
                  ;         ENTERED VIA JUMPS FROM 'DBYTE','LCRLF' ROUTINES                      ;
                  ; PROCESS: LIST OUTPUT                                                         ;
                  ; INPUT: VALUE IN C-REG                                                        ;
                  ; OUTPUT:                                                                      ;
                  ; MODIFIED: A, FLAGS, C                                                        ;
                  ; STACK USAGE: 2 BYTES                                                         ;
                  ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                  LOM:                            ; LIST OUTPUT ON CONSOLE
FD14 3AØ3ØØ               LDA     IOBYT
FD17 E6Ø3                 ANI     NOT CMSK        ; LOOK ONLY AT CONSOLE FIELD OF IOBYT
FD19 FEØ2                 CPI     BATCH           ; IS CONSOLE ASSIGNED TO BATCH MODE?
FD1B C4Ø7FC               CNZ     BREAK           ; IF IT ISN'T, WE SHOULD TEST FOR BREAK KEY
                                                  ;    I.E. IN BATCH MODE THE BREAK KEY IS NOT
                                                  ;    HONORED
                  LO:                             ; LIST OUTPUT
FD1E 3AØ3ØØ               LDA     IOBYT           ; GET STATUS BYTE
FD21 E6CØ                 ANI     NOT LMSK        ; LOOK AT LIST FIELD
FD23 CAA7FC               JZ      TTYOUT          ; JUMP IF LIST = TTY
FD26 FE4Ø                 CPI     LCRT
FD28 CABEFC               JZ      CRTOUT          ; JUMP IF LIST = CRT
FD2B FECØ                 CPI     LUSE            ; TEST FOR USER DEFINED LIST DEVICE
FD2D 3EFA                 MVI     A,L1LOC AND ØFFH; A := LSB OF L1LOC ADDRESS
FD2F CA8CFC               JZ      @USER           ; JUMP IF LIST = USER-DEFINED DEVICE
                  ;-------------------------------
                  ; LIST = LPT
FD32 C5                   PUSH    B               ; SAVE BC
                  LPØ:
FD33 Ø615                 MVI     B,LSTC          ; LOAD LINE PRINTER STATUS COMMAND
FD35 CDB5FF               CALL    PIODR1          ; READ STATUS
FD38 E6Ø1                 ANI     LPTRY           ; IS IT READY?
FD3A CA33FD               JZ      LPØ             ; LOOP UNTIL IT IS
FD3D Ø614                 MVI     B,LPTC          ; LOAD LINE PRINTER PRINT COMMAND
FD3F CDCEFF               CALL    PIODR3          ; OUTPUT CHARACTER CONTAINED IN C-REG
FD42 C1                   POP     B               ; RESTORE BC
FD43 C9                   RET
                  ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                  ; 'CSTS' - EXTERNALLY REFERENCED ROUTINE                                       ;
                  ;          ENTERED VIA CALL FROM 'BREAK' ROUTINE                               ;
                  ; PROCESS: LOCAL CONSOLE INPUT STATUS                                          ;
                  ; INPUT:                                                                       ;
                  ; OUTPUT: A-REG CONTAINS ØØ IF NO KEY HAS BEEN DEPRESSED,                      ;
                  ;         A-REG CONTAINS FFH IF A KEY HAS BEEN DEPRESSED                       ;
                  ; MODIFIED: A, FLAGS                                                           ;
                  ; STACK USAGE: 2 BYTES                                                         ;
                  ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                  CSTS:                           ; LOCAL CONSOLE INPUT STATUS
FD44 3AØ3ØØ               LDA     IOBYT           ; GET STATUS BYTE
FD47 E6Ø3                 ANI     NOT CMSK        ; LOOK ONLY AT CONSOLE FIELD OF IOBYT
FD49 C253FD               JNZ     CSØ             ; JUMP IF CONSOLE IS NOT TTY
                  ;-------------------------------
```

```
LOC  OBJ          LINE        SOURCE STATEMENT

                  ; CONSOLE = TTY
FD4C DBF5                 IN      TTYS            ; GET TTY STATUS
FD4E E6Ø2                 ANI     RRDY            ; IS RECEIVE BUFFER READY? (IF TTY KEY WAS
                                                  ;    DEPRESSED, ZERO BIT WILL BE RESET)
FD5Ø C374FD               JMP     CS2
                  ;-------------------------------
                  ; CONSOLE = CRT, BATCH, OR USER-DEFINED
                  CSØ:
FD53 FEØ1                 CPI     CCRT            ; CONSOLE = CRT?
FD55 C279FD               JNZ     CS3             ; JUMP IF CONSOLE IS NOT CRT
FD58 E5                   PUSH    H               ; SAVE H,L
FD59 2AØ4ØØ               LHLD    MEMTOP
FD5C 2ECC                 MVI     L,ILOC-1 AND ØFFH; HL POINTS TO CONFIGURATION BYTE IN EXIT TEMPLATE
FD5E 7E                   MOV     A,M             ; A CONTAINS THIS CONFIGURATION BYTE
FD5F E1                   POP     H               ; RESTORE H,L
FD6Ø ØF                   RRC                     ; ROTATE BIT Ø INTO CARRY; THUS CARRY = 1
                                                  ;    MEANS INTEGRATED CRT NOT PRESENT
FD61 D26BFD               JNC     CS1             ; JUMP IF INTEGRATED CRT PRESENT
                  ;--------------------------------
                  ; CONSOLE = SERIAL CRT
FD64 DBF7                 IN      USCS            ; GET CRT STATUS
FD66 E6Ø2                 ANI     RRDY            ; IS RECEIVE BUFFER READY? (IF KEY HAS BEEN
                                                  ;    DEPRESSED, ZERO BIT WILL BE RESET)
FD68 C374FD               JMP     CS2
                  ;-------------------------------
                  ; CONSOLE = INTEGRATED CRT
                  CS1:                            ; INTELLEC WITH INTEGRATED CRT
FD6B C5                   PUSH    B               ; SAVE B,C
FD6C Ø613                 MVI     B,KSTS          ; LOAD CRT STATUS COMMAND
FD6E CD7FFF               CALL    IOCDR1          ; GET CRT STATUS
FD71 E6Ø1                 ANI     KRDY            ; IS RECEIVE BUFFER READY? (IF KEY HAS BEEN
                                                  ;    DEPRESSED, ZERO BIT WILL BE RESET)
FD73 C1                   POP     B               ; RESTORE B,C
                  CS2:                            ; COMMON RETURN POINT FOR CRT,TTY
FD74 3EØØ                 MVI     A,FALSE         ; INITIALIZE A-REG TO ØØ
FD76 C8                   RZ                      ; RETURN WITH A := ØØ IF NO DATA AVAILABLE
FD77 2F                   CMA
FD78 C9                   RET                     ; RETURN WITH A := FF IF DATA AVAILABLE
                  ;-------------------------------
                  ; CONSOLE = BATCH OR USER-DEFINED DEVICE
                  CS3:
FD79 FEØ2                 CPI     BATCH           ; IS IT BATCH?
FD7B 3EFF                 MVI     A,TRUE
FD7D C8                   RZ                      ; RETURN IF CONSOLE IS BATCH; A := FF
FD7E 3EFD                 MVI     A,CSLOC AND ØFFH; CONSOLE = USER DEFINED LOCAL CONSOLE, BRANCH
                                                  ;    TO USER'S OWN STATUS ROUTINE
FD8Ø C38CFC               JMP     @USER
                  ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                  ; 'IOCHK' - EXTERNALLY REFERENCED ROUTINE                                       ;
                  ; PROCESS: GET I/O SYSTEM STATUS                                                ;
                  ; INPUT:                                                                        ;
                  ; OUTPUT: STATUS BYTE RETURNED IN A-REG                                         ;
                  ; MODIFIED: A                                                                   ;
                  ; STACK USAGE: 2 BYTES                                                          ;
                  ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

```
 LOC  OBJ          LINE        SOURCE STATEMENT

                   2247 IOCHK:
 FD83 3AØ3ØØ       2248         LDA     IOBYT           ; GET STATUS BYTE
 FD86 C9           2249         RET                     ; RETURN
                   225Ø ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                   2251 ; 'IOSET' - EXTERNALLY REFERENCED ROUTINE                                    ;
                   2252 ; PROCESS: SET I/O CONFIGURATION                                             ;
                   2253 ; INPUT: NEW I/O STATUS BYTE IN C-REG                                        ;
                   2254 ; OUTPUT: IOBYT CONTAINS NEW I/O CONFIGURATION                               ;
                   2255 ; MODIFIED: A, C                                                             ;
                   2256 ; STACK USAGE: 2 BYTES                                                       ;
                   2257 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                   2258 IOSET:
 FD87 79           2259         MOV     A,C
 FD88 32Ø3ØØ       226Ø         STA     IOBYT           ; PUT NEW IOBYT IN MEMORY
 FD8B C9           2261         RET                     ; RETURN
                   2262 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                   2263 ; 'MEMCHK' - EXTERNALLY REFERENCED ROUTINE                                   ;
                   2264 ; PROCESS: RETURN ADDRESS OF CONTIGUOUS END OF USER MEMORY                   ;
                   2265 ; INPUT: MEMTOP,USER                                                         ;
                   2266 ; OUTPUT: ADDRESS IS RETURNED IN B-REG (MSB) AND A-REG (LSB)                 ;
                   2267 ; MODIFIED: A,B,FLAGS                                                        ;
                   2268 ; STACK USAGE: 2 BYTES                                                       ;
                   2269 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                   227Ø MEMCHK:
 FD8C 3AØ5ØØ       2271         LDA     MEMTOP+1        ; MSB OF ADDRESS OF TOP PAGE OF MEMORY
 FD8F 3D           2272         DCR     A               ; CHANGE IT TO THE PAGE BELOW THE TOP PAGE
                   2273                                 ;    RECALL TOP PAGE IS USED BY MONITOR SO
                   2274                                 ;    USER SHOULD NOT ACCESS IT
 FD9Ø 47           2275         MOV     B,A             ; SO MSB GOES IN B-REG
 FD91 3ECØ         2276         MVI     A,USER AND ØFFH ; LSB IN A-REG
 FD93 C9           2277         RET                     ; AB POINTS TO BASE OF USER STACK IN SECOND
                   2278                                 ;    FROM TOP PAGE OF RAM
                   2279 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                   228Ø ; 'IODEF' - EXTERNALLY REFERENCED ROUTINE                                    ;
                   2281 ; PROCESS: DEFINE USER I/O ENTRY POINTS                                      ;
                   2282 ; INPUT: SELECTION CODE IN C-REG, USER ENTRY POINT ADDRESS IN D,E            ;
                   2283 ; OUTPUT:                                                                    ;
                   2284 ; MODIFIED: A, FLAGS                                                         ;
                   2285 ; STACK USAGE: 8 BYTES                                                       ;
                   2286 ; EXPLANATION: POINT HL TO TABLE OF USER ENTRY POINTS IN TOP OF RAM;         ;
                   2287 ;    SUBSTITUTE IN THERE THE ADDRESS GIVEN BY THE USER IN DE REGISTERS.      ;
                   2288 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                   2289 IODEF:
 FD94 E5           229Ø         PUSH    H               ; SAVE H & L
 FD95 C5           2291         PUSH    B               ; SAVE B & C
 FD96 2AØ4ØØ       2292         LHLD    MEMTOP          ; GET XTBL+1
 FD99 2EE9         2293         MVI     L,XTBL+1 AND ØFFH; HL NOW POINTS TO XTBL+1 IN TOP PAGE OF RAM
 FD9B 79           2294         MOV     A,C             ; A := LOGICAL DEVICE CATEGORY
 FD9C FEØ8         2295         CPI     UCS+1
 FD9E D247F8       2296         JNC     ERROR           ; INVALID SELECTION CODE
 FDA1 81           2297         ADD     C               ; DOUBLE INDEX
 FDA2 81           2298         ADD     C               ; TRIPLE INDEX
 FDA3 4F           2299         MOV     C,A
 FDA4 Ø6ØØ         23ØØ         MVI     B,Ø
 FDA6 Ø9           23Ø1         DAD     B               ; COMPUTE PROPER INDEX INTO XTBL
 LOC  OBJ          LINE        SOURCE STATEMENT

 FDA7 73                   MOV     M,E             ; STORE BRANCH OPERAND IN INSTRUCTION
 FDA8 23                   INX     H
 FDA9 72                   MOV     M,D             ; STORE THE USER-DEFINED I/O ENTRY ROUTINE
                                                   ;    ADDRESS IN THE PROPER PLACE IN XTBL,
                                                   ;    SO IT LOOKS LIKE:
                                                   ;    JMP <USER-DEFINED ADDRESS>
 FDAA C1                   POP     B               ; RESTORE B & C
 FDAB E1                   POP     H               ; RESTORE H & L
 FDAC C9                   RET
                  ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                  ; 'UI' - EXTERNALLY REFERENCED ROUTINE                                     ;
                  ; PROCESS: INPUT A CHARACTER FROM THE UPP                                  ;
                  ; INPUT: B CONTAINS MSB OF PROM ADDRESS                                    ;
                  ;        C CONTAINS LSB OF PROM ADDRESS                                    ;
                  ; OUTPUT: DATA IN A-REG
                  ; MODIFIED:A,FLAGS                                                         ;
                  ; STACK USAGE: 6 BYTES                                                     ;
                  ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                  UI:
                                                   ; IT IS ASSUMED THE 'UPPS' ROUTINE HAS BEEN
                                                   ; CALLED AND THAT THE UPP UNIT IS READY
 FDAD C5                   PUSH    B               ; SAVE B,C
 FDAE 0617                 MVI     B,RPPC          ; LOAD THE READ PROM COMMAND
                                                   ; C CONTAINS PROM LOW ADDRESS
 FDB0 CDCEFF               CALL    PIODR3          ; OUTPUT READ PROM COMMAND
                                                   ; OUTPUT PROM LOW ADDRESS
 FDB3 C1                   POP     B               ; RESTORE B,C; B CONTAINS PROM HIGH ADDRESS
 FDB4 C5                   PUSH    B               ; SAVE B,C
 FDB5 48                   MOV     C,B             ; C CONTAINS PROM HIGH ADDRESS
 FDB6 CDD1FF               CALL    PIODR4          ; OUTPUT PROM HIGH ADDRESS
 FDB9 C1                   POP     B               ; RESTORE B,C
 FDBA CDB8FF               CALL    PIODR2          ; INPUT PROM DATA
 FDBD C9                   RET
                  ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                  ; 'UO' - EXTERNALLY REFERENCED ROUTINE                                     ;
                  ; PROCESS: OUTPUT A CHARACTER TO THE UPP                                   ;
                  ; INPUT: C CONTAINS THE CHARACTER TO BE WRITTEN INTO THE PROM              ;
                  ;        D CONTAINS THE MSB OF THE PROM ADDRESS                            ;
                  ;        E CONTAINS THE LSB OF THE PROM ADDRESS                            ;
                  ; OUTPUT:                                                                  ;
                  ; MODIFIED:A,FLAGS                                                         ;
                  ; STACK USAGE: 8 BYTES                                                     ;
                  ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                  UO:
                                                   ; IT IS ASSUMED THE 'UPPS' ROUTINE HAS BEEN
                                                   ; CALLED AND THAT THE UPP UNIT IS READY
 FDBE C5                   PUSH    B               ; SAVE B,C
 FDBF 0616                 MVI     B,WPPC          ; LOAD WRITE PROM COMMAND
 FDC1 4B                   MOV     C,E             ; LOAD PROM LOW ADDRESS
 FDC2 CDCEFF               CALL    PIODR3          ; OUTPUT WRITE PROM COMMAND
                                                   ; OUTPUT PROM LOW ADDRESS
 FDC5 4A                   MOV     C,D             ; LOAD PROM HIGH ADDRESS
 FDC6 CDD1FF               CALL    PIODR4          ; OUTPUT PROM HIGH ADDRESS
 FDC9 C1                   POP     B               ; RESTORE B,C; C CONTAINS THE DATA TO BE
                                                   ;    WRITTEN TO THE PROM
```

```
LOC  OBJ          SOURCE STATEMENT

FDCA CDD1FF              CALL    PIODR4          ; OUTPUT DATA TO PROM
FDCD C9                  RET
                 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                 ; 'UPPS' - EXTERNALLY REFERENCED ROUTINE                                       ;
                 ; PROCESS: INPUT THE UPP STATUS BYTE                                           ;
                 ; INPUT:                                                                       ;
                 ; OUTPUT: A-REG CONTAINS THE UPP STATUS BYTE                                   ;
                 ; MODIFIED:                                                                    ;
                 ; STACK USAGE: 8 BYTES                                                         ;
                 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                 UPPS:
FDCE C5                  PUSH    B               ; SAVE BC
FDCF Ø618                MVI     B,RPSTC         ; B CONTAINS STATUS COMMAND
FDD1 CDB5FF              CALL    PIODR1          ; GET UPP STATUS BYTE
FDD4 F5                  PUSH    PSW             ; SAVE IT ON THE STACK
FDD5 CDB8FF              CALL    PIODR2          ; GET PIO DEVICE STATUS BYTE AND IGNORE IT
FDD8 F1                  POP     PSW             ; A NOW CONTAINS UPP STATUS BYTE
FDD9 C1                  POP     B               ; RESTORE BC
FDDA C9                  RET
                 ;******************************************************************************
                 ;*                                                                            *
                 ;*  END OF I/O SUBROUTINES, BEGINNING OF MONITOR SUBROUTINES                  *
                 ;*                                                                            *
                 ;******************************************************************************
                 ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
                 ; 'BYTE' - ENTERED VIA CALL FROM 'R' COMMAND
                 ; PROCESS: READ TWO 8-BIT ASCII CHARACTERS, DECODE INTO ONE 8-BIT BINARY WORD
                 ; INPUT: D CONTAINS RUNNING CHECKSUM
                 ; OUTPUT: DECODED BYTE IN A-REG, RUNNING CHECKSUM IN D-REG, ZERO BIT SET OR RESET
                 ; MODIFIED: A,F,C,D
                 ; STACK USAGE:
                 BYTE:
FDDB C5                  PUSH    B               ; SAVE B,C
FDDC CD58FF              CALL    RIX             ; READ ONE ASCII CHAR FROM TAPE, PUT IN A-REG
FDDF CD98FE              CALL    NIBBLE          ; CONVERT 8-BIT ASCII TO 4-BIT HEXADECIMAL VALUE
FDE2 Ø7                  RLC                     ; SHIFT FOUR PLACES TO THE LEFT
FDE3 Ø7                  RLC
FDE4 Ø7                  RLC
FDE5 Ø7                  RLC                     ; MOVE HEX CHAR TO 4 MSB OF A-REG
FDE6 4F                  MOV     C,A             ; STORE TEMPORARILY IN C
FDE7 CD58FF              CALL    RIX             ; GET ANOTHER ASCII CHAR FROM READER
FDEA CD98FE              CALL    NIBBLE          ; CONVERT TO 4 BIT HEX; NOW LSB OF A-REG
FDED B1                  ORA     C               ; ASSEMBLE IT ALL TOGETHER
FDEE 4F                  MOV     C,A             ; STORE IT TEMPORARILY IN C
FDEF 82                  ADD     D               ; UPDATE CHECKSUM (ZERO BIT IS SET/RESET)
FDFØ 57                  MOV     D,A             ; D CONTAINS UPDATED CHECKSUM
FDF1 79                  MOV     A,C             ; LOAD THE CONVERTED WORD
FDF2 C1                  POP     B
FDF3 C9                  RET                     ; RETURN
                 ;/////////////////////////////////////////////////////////////////////////////
                 ; 'CONV' - ENTERED VIA CALLS FROM 'DBYTE','HXD','PBYTE' ROUTINES
                 ; PROCESS: CONVERT 4 BIT HEX VALUE TO ASCII CHARACTER
                 ; INPUT : Ø,1,2,3,4,5,6,7,8,9,A,B,C,D,E, OR F IN HEX IN A-REG
                 ; OUTPUT: 3ØH,...,39H,41H,...,46H IN C-REG
                 ; MODIFIED: A, FLAGS, C
```

```
LOC  OBJ         LINE        SOURCE STATEMENT

                 ; STACK USAGE:
                 ;
                 CONV:
FDF4 E6ØF                ANI     ØFH                 ; ONLY 4 LSB ARE SIGNIFICANT, SO MASK 4 MSB
FDF6 C69Ø                ADI     9ØH                 ; SET UP A-REG SO THAT A-F CAUSE CARRY
FDF8 27                  DAA
FDF9 CE4Ø                ACI     4ØH                 ; ADD IN CARRY AND ADJUST UPPER NIBBLE
FDFB 27                  DAA
FDFC 4F                  MOV     C,A                 ; STORE CONVERTED RESULT IN C-REG
FDFD C9                  RET                         ; RETURN
                 ;///////////////////////////////////////////////////////////////////////////////
                 ; 'CRLF' - ENTERED VIA CALLS FROM 'G','H','Q','R','W','X' COMMANDS AND
                 ;          'START' ROUTINE
                 ; PROCESS: TYPE CARRIAGE RETURN AND LINE FEED ON LOCAL CONSOLE
                 ; INPUT:
                 ; OUTPUT:
                 ; MODIFIED:
                 ; STACK USAGE:
                 CRLF:
FDFE CDDEFC              CALL    COMC                ; OUTPUT <CR> ON CONSOLE
FEØ1 ØD                  DB      CR
FEØ2 CDDEFC              CALL    COMC                ; OUTPUT <LF> ON CONSOLE
FEØ5 ØA                  DB      LF
FEØ6 C9                  RET
                 ;///////////////////////////////////////////////////////////////////////////////
                 ; 'DADR' - ENTERED VIA CALL FROM 'D' COMMAND
                 ; PROCESS: PRINT CONTENTS OF HL IN HEX FORMAT ON LIST DEVICE
                 ; INPUT: HL CONTAINS <LOW ADDRESS> OF 'D' COMMAND
                 ; OUTPUT:
                 ; MODIFIED: A
                 ; STACK USAGE:
                 DADR:
FEØ7 7C                  MOV     A,H                 ; PRINT MSB OF LOW ADDRESS
FEØ8 CDØCFE              CALL    DBYTE
FEØB 7D                  MOV     A,L                 ; PRINT LSB OF LOW ADDRESS
                 ;*******JMP     DBYTE
                 ;///////////////////////////////////////////////////////////////////////////////
                 ; 'DBYTE' - ENTERED VIA CALLS FROM 'D' COMMAND AND 'DADR' ROUTINE
                 ;           ENTERED VIA FALL-THRU FROM 'DADR' ROUTINE
                 ; PROCESS: LIST A BYTE ON THE LIST DEVICE AS TWO ASCII CHARACTERS
                 ; INPUT: A CONTAINS THE BYTE TO BE LISTED
                 ; OUTPUT:
                 ; MODIFIED:
                 ; STACK USAGE:
                 DBYTE:
FEØC F5                  PUSH    PSW                 ; SAVE A COPY OF A-REG
FEØD ØF                  RRC
FEØE ØF                  RRC
FEØF ØF                  RRC
FE1Ø ØF                  RRC                         ; WANT TO LOOK ONLY AT BITS 4-7 OF A-REG
FE11 CDF4FD              CALL    CONV                ; CONVERT 4 MSB OF ORIGINAL A-REG TO 1 ASCII CHAR
FE14 CD14FD              CALL    LOM                 ; OUTPUT ON LIST DEVICE
FE17 F1                  POP     PSW                 ; RETRIEVE ORIGINAL VALUE
FE18 CDF4FD              CALL    CONV                ; CONVERT 4 LSB OF ORIGINAL A-REG TO 1 ASCII CHAR
FE1B C314FD              JMP     LOM                 ; OUTPUT ON LIST DEVICE
```

```
LOC  OBJ         SOURCE STATEMENT

                 ;//////////////////////////////////////////////////////////////////////////////
                 ; 'DELAY' - ENTERED VIA CALL FROM 'RI' ROUTINE
                 ; PROCESS: 1.Ø MS. DELAY
                 ; INPUT: ONEMS
                 ; OUTPUT: ROUTINE IDLES FOR 1.Ø MS.
                 ; MODIFIED: C, FLAGS
                 ; STACK USAGE: 2 BYTES
                 DELAY:
FE1E ØE7Ø                MVI     C,ONEMS          ; LOAD 1 MS.CONSTANT (USE 3BH IN ICE ENVIRONMENT)
                 DLY1:
FE2Ø ØD                  DCR     C                ; DECREMENT COUNTER
FE21 C22ØFE              JNZ     DLY1             ; JUMP IF NOT EXPIRED
FE24 C9                  RET                      ; RETURN
                 ;//////////////////////////////////////////////////////////////////////////////
                 ; 'DREG' - ENTERED VIA CALL FROM 'X' COMMAND
                 ; PROCESS: DISPLAY THE CONTENTS OF A USER REGISTER
                 ; INPUT: HL POINTS TO CHARACTER IN ACTBL OF 'X' COMMAND
                 ; OUTPUT: HL POINTS TO NEXT CHARACTER IN ACTBL,
                 ;         DE CONTAINS ADDRESS OF REGISTER LOCATION
                 ;         B CONTAINS REGISTER PRECISION
                 ; MODIFIED:
                 ; STACK USAGE:
                 DREG:
FE25 23                  INX     H                ; HL POINTS TO LOCATION ENTRY IN ACTBL OF 'X' COMMAND
FE26 5E                  MOV     E,M              ; INCREMENT HL TO POINT AT DISPLACEMENT
FE27 3AØ5ØØ              LDA     MEMTOP+1
FE2A 57                  MOV     D,A              ; D := MSB OF ADDRESS OF TOP PAGE OF MEMORY
                                                  ; DE POINTS TO THAT PART OF THE EXIT TEMPLATE
                                                  ;    CONTAINING SAVED REGISTER VALUES
FE2B 23                  INX     H                ; HL POINTS TO PRECISION IN ACTBL
FE2C 46                  MOV     B,M              ; PRECISION, Ø=8 BITS, 1=16 BITS
FE2D 23                  INX     H                ; POINT AT NEXT REGISTER IDENTIFIER
FE2E 1A                  LDAX    D                ; 8/16 BIT DISPLAY AND MODIFICATION
FE2F CD5BFE              CALL    LBYTE            ; MSB OF 16 BIT REG, ALL OF 8 BIT REG
FE32 Ø5                  DCR     B                ; TEST PRECISION
FE33 F8                  RM                       ; 8 BIT DISPLAY, RETURN
FE34 1B                  DCX     D
FE35 1A                  LDAX    D
FE36 C35BFE              JMP     LBYTE            ; LSB OF 16 BIT REG
                 ;//////////////////////////////////////////////////////////////////////////////
                 ; 'EXPR' - ENTERED VIA CALLS FROM 'D','E','F','H','M','R','W' COMMANDS
                 ; PROCESS: EVALUATE EXPRESSION "<EXPR>,<EXPR>,<EXPR>"
                 ; INPUT: C-REG CONTAINS THE NUMBER OF PARAMETERS REQUIRED (1,2, OR 3)
                 ; OUTPUT: STACK CONTAINS THE PARAMETERS IN REVERSE ORDER
                 ; MODIFIED: F,C,H,L,SP
                 ; STACK USAGE:
                 EXPR:
FE39 CD74FE              CALL    PARAM            ; GET A HEXADECIMAL PARAMETER, RETURNED IN HL
FE3C E3                  XTHL                     ; PUT THE PARAMETER IN THE STACK; HL NOW
                                                  ;    CONTAINS RETURN ADDRESS OF CALL TO 'EXPR'
FE3D E5                  PUSH    H                ; PUT RETURN ADDRESS ON TOP OF STACK
FE3E ØD                  DCR     C                ; DECREMENT PARAMETER COUNT; CARRY BIT UNAFFECTED
FE3F D246FE              JNC     EXØ              ; JUMP IF COMMA ENTERED (PARAM CALLS PCHK)
FE42 C247F8              JNZ     ERROR            ; INCORRECT PARAM COUNT
FE45 C9                  RET
```

```
LOC  OBJ          LINE        SOURCE STATEMENT

                       EXØ:
FE46 C239FE                    JNZ     EXPR            ; GET ANOTHER PARAMETER
FE49 C347F8                    JMP     ERROR           ; NOT TERMINATED WITH CR
                       ;//////////////////////////////////////////////////////////////////////////////
                       ; 'HILO' - ENTERED VIA CALLS FROM 'D','F','M','W' COMMANDS
                       ; PROCESS: COMPARE HL WITH DE
                       ; INPUT: ADDRESS VALUES IN HL AND DE
                       ; OUTPUT: IF HL <= DE THEN CARRY = Ø;
                       ;         IF HL  > DE THEN CARRY = 1
                       ; MODIFIED: HL,A,F
                       ; STACK USAGE:
                       HILO:
FE4C 23                        INX     H               ; INCREMENT HL ADDRESS
FE4D 7C                        MOV     A,H             ; TEST FOR HL = Ø
FE4E B5                        ORA     L               ; ZERO BIT SET IF H=L=ØØ, I.E. HL MUST
                                                       ;    HAVE BEEN FFFFH
FE4F 37                        STC                     ; CARRY := 1
FE5Ø C8                        RZ
FE51 7B                        MOV     A,E             ; DE - HL, SET/RESET CARRY
FE52 95                        SUB     L               ; (LSB OF HIGH ADDR) - (MSB OF LOW ADDR)
FE53 7A                        MOV     A,D
FE54 9C                        SBB     H               ; (MSB OF HIGH ADDR) - (MSB OF LOW ADDR)
FE55 C9                        RET                     ; RETURN
                       ;/////////////////////////////////////////////////////////////////////////////
                       ; 'LADR' - ENTERED VIA CALLS FROM 'H' COMMAND AND 'RESTART' ROUTINE
                       ; PROCESS: PRINT CONTENTS OF HL IN HEX ON LOCAL CONSOLE DEVICE
                       ; INPUT: HL CONTAINS THE HEX VALUE TO BE OUTPUT(16 BITS)
                       ; OUTPUT:
                       ; MODIFIED: H,L,A
                       ; STACK USAGE:
                       LADR:
FE56 7C                        MOV     A,H
FE57 CD5BFE                    CALL    LBYTE           ; PRINT 8 MSB OF HEX VALUE ON CONSOLE
FE5A 7D                        MOV     A,L
                       ;*******JMP     LBYTE           ; PRINT 8 LSB OF HEX VALUE ON CONSOLE
                       ;/////////////////////////////////////////////////////////////////////////////
                       ; 'LBYTE' - ENTERED VIA CALLS FROM 'S' COMMAND AND 'DREG','LADR' ROUTINES
                       ;           ENTERED VIA JUMP FROM 'DREG' ROUTINE
                       ;           ENTERED VIA FALL-THRU FROM 'LADR' ROUTINE
                       ; PROCESS: LIST A BYTE AS TWO ASCII CHARACTERS
                       ; INPUT: A-REG CONTAINS THE 8 BITS TO BE CONVERTED TO ASCII
                       ; OUTPUT:
                       ; MODIFIED: A,F
                       ; STACK USAGE: 6 BYTES
                       LBYTE:
FE5B F5                        PUSH    PSW             ; SAVE A-REG
FE5C ØF                        RRC
FE5D ØF                        RRC
FE5E ØF                        RRC
FE5F ØF                        RRC                     ; LOOK ONLY AT 4 MSB OF THE BYTE VALUE
FE6Ø CD64FE                    CALL    HXD             ; CONVERT IT TO ONE ASCII CHAR AND OUTPUT IT
FE63 F1                        POP     PSW             ; RETRIEVE ORIGINAL VALUE
                       ;*******JMP     HXD             ; CONVERT 4 LSB OF BYTE TO ASCII AND OUTPUT IT
                       ;//////////////////////////////////////////////////////////////////////////////
                       ; 'HXD' - ENTERED VIA CALL FROM 'LBYTE' ROUTINE
```

```
 LOC  OBJ         LINE        SOURCE STATEMENT

                  ;         ENTERED VIA FALL-THRU FROM 'LBYTE' ROUTINE
                  ; PROCESS: CONVERT 4 LSB IN A-REG INTO ONE ASCII CHAR IN A-REG, PRINT IT
                  ;          ON LOCAL CONSOLE DEVICE
                  ; INPUT: NIBBLE TO BE CONVERTED IS IN BITS Ø-3 OF A-REG
                  ; OUTPUT:
                  ; MODIFIED: A-REG
                  ; STACK USAGE:
                  HXD:
 FE64 CDF4FD              CALL    CONV            ; CONVERT 4 BITS TO ONE 8-BIT ASCII CHAR
 FE67 C395FC              JMP     COM             ; OUTPUT ON LOCAL CONSOLE
                  ;////////////////////////////////////////////////////////////////////////////
                  ; 'LCRLF' - ENTERED VIA CALL FROM 'D' COMMAND
                  ; PROCESS: PRINT <CR>,<LF> ON LIST DEVICE
                  ; INPUT:
                  ; OUTPUT:
                  ; MODIFIED: C
                  ; STACK USAGE: 4 BYTES
                  LCRLF:
 FE6A ØEØD                MVI     C,CR
 FE6C CD14FD              CALL    LOM             ; OUTPUT <CR> TO LIST DEVICE
 FE6F ØEØA                MVI     C,LF
 FE71 C314FD              JMP     LOM             ; OUTPUT <LF> TO LIST DEVICE
                  ;//////////////////////////////////////////////////////////////////////////////
                  ; 'PARAM' - ENTERED VIA CALLS FROM 'G','S' COMMANDS AND 'EXPR' ROUTINE
                  ; 'PAØ' - ENTERED VIA CALLS FROM 'G','S','X' COMMANDS
                  ; PROCESS: COLLECT A HEXADECIMAL PARAMETER
                  ; INPUT:
                  ; OUTPUT: HEXADECIMAL PARAMETER IN HL
                  ; MODIFIED: A,F,B,H,L
                  ; STACK USAGE:
                  PARAM:
 FE74 CDC5FE              CALL    PCHK            ; GET FIRST CHARACTER
 FE77 CA47F8              JZ      ERROR           ; DISALLOW NULL PARAMETERS
                  PAØ:
 FE7A 21ØØØØ              LXI     H,Ø             ; INTIALIZE HL := ØØØØ
                  PA1:
 FE7D 47                  MOV     B,A             ; SAVE CHAR IN CASE IT'S A DELIMITER
 FE7E CD98FE              CALL    NIBBLE          ; CONVERT THE ASCII CHARACTER TO HEX; MUST BE
                                                  ;    Ø-9,A-F; IF NOT THE CARRY BIT IS SET
 FE81 DA9ØFE              JC      PA2             ; NOT LEGAL CHAR, TREAT AS DELIMITER
 FE84 29                  DAD     H               ; *2
 FE85 29                  DAD     H               ; *4
 FE86 29                  DAD     H               ; *8
 FE87 29                  DAD     H               ; *16 --- SHIFT THE OLD HEX VALUES 4 PLACES TO LEFT
 FE88 B5                  ORA     L               ; PUT NEW HEX VALUE IN 4 LSB OF L-REG
 FE89 6F                  MOV     L,A
 FE8A CD61FF              CALL    TI              ; GET SUBSEQUENT CHARACTERS
 FE8D C37DFE              JMP     PA1             ; DECODE NEXT CHARACTER
                  PA2:
 FE9Ø 78                  MOV     A,B             ; A := B := DELIMITER CHARACTER
 FE91 CDC8FE              CALL    P2C             ; IS IT A VALID DELIMITER?
 FE94 C247F8              JNZ     ERROR           ; JUMP TO ERROR IF IT ISN'T
 FE97 C9                  RET
                  ;////////////////////////////////////////////////////////////////////
                  ; 'NIBBLE' - ENTERED VIA CALLS FROM 'BYTE','PARAM','PAØ' ROUTINES
```

```
LOC  OBJ         LINE        SOURCE STATEMENT

                 ; PROCESS: DECODE 8-BIT ASCII CHAR IN A-REG INTO 4-BIT HEX DIGIT IN A-REG,
                 ;          FILTER OUT ALL CHARACTERS NOT IN THE ASCII CODING SEQUENCE
                 ;          Ø,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F.
                 ; INPUT: 8-BIT ASCII CHAR IN A-REG
                 ; OUTPUT: VALID HEX EQUIVALENT IN A-REG AND CARRY = Ø, OTHERWISE
                 ;         GARBAGE IN A-REG AND CARRY = 1 (INDICATING ILLEGAL CHARACTER)
                 ; MODIFIED: A, FLAGS
                 ; STACK USAGE: 2 BYTES
                 NIBBLE:
FE98 D63Ø                SUI     'Ø'               ; IF THE ASCII CHAR IS BETWEEN ØØ AND 2FH,
FE9A D8                  RC                        ;    THEN RETURN WITH CARRY = 1
FE9B C6E9                ADI     'Ø' - 'G'         ; IF THE ASCII CHAR IS GREATER THAN 46H,
FE9D D8                  RC                        ;    THEN RETURN WITH CARRY = 1
FE9E C6Ø6                ADI     6                 ; ORIGINAL ASCII CHAR WAS BETWEEN 3ØH AND 46H INCL.
FEAØ F2A6FE              JP      NIØ               ; JUMP IF IT WAS 41H THRU 46H (I.E. A-F)
FEA3 C6Ø7                ADI     7                 ; ORIGINAL ASCII CHAR WAS BETWEEN 3ØH AND 4ØH INCL.
FEA5 D8                  RC                        ; RETURN WITH CARRY = 1 IF ASCII CHAR WAS
                                                   ;    BETWEEN 3AH AND 4ØH INCLUSIVE
                 NIØ:                              ; VALID VALUE: 3ØH-39H,41H-46H
FEA6 C6ØA                ADI     1Ø                ; A-REG NOW CONTAINS HEX EQUIV.(Ø-9,A-F)
FEA8 B7                  ORA     A                 ; CLEAR ERROR FLAG (I.E. RESET CARRY BIT)
FEA9 C9                  RET                       ; RETURN WITH VALID VALUE
                 ;//////////////////////////////////////////////////////////////////////
                 ; 'PADR' - ENTERED VIA CALLS FROM 'E','W' COMMANDS
                 ; PROCESS: PUNCH CONTENTS OF HL IN HEX ON PUNCH DEVICE
                 ; INPUT: HL CONTAINS 8-BIT LOAD ADDRESS
                 ; OUTPUT:
                 ; MODIFIED: A
                 ; STACK USAGE: 4 BYTES
                 PADR:
FEAA 7C                  MOV     A,H               ; A := MSB OF LOAD ADDRESS
FEAB CDAFFE              CALL    PBYTE             ; EMIT FRAMES 3 & 4
FEAE 7D                  MOV     A,L               ; A := LSB OF LOAD ADDRESS
                 ;*******JMP     PBYTE             ; EMIT FRAMES 5 & 6
                 ;///////////////////////////////////////////////////////////////////////
                 ; 'PBYTE' - ENTERED VIA CALLS FROM 'E','W' COMMANDS AND 'PADR' ROUTINE
                 ;           ENTERED VIA FALL-THRU FROM 'PADR' ROUTINE
                 ; PROCESS: PUNCH A BYTE AS 2 ASCII CHARACTERS
                 ; INPUT: A-REG CONTAINS BYTE TO BE CONVERTED, D CONTAINS RUNNING CHECKSUM
                 ; OUTPUT: D CONTAINS UPDATED CHECKSUM
                 ; MODIFIED: A,F,D,E
                 ; STACK USAGE:
                 PBYTE:
FEAF 5F                  MOV     E,A               ; SAVE BYTE TO BE CONVERTED IN E-REG
FEBØ ØF                  RRC
FEB1 ØF                  RRC
FEB2 ØF                  RRC
FEB3 ØF                  RRC                       ; LOOK ONLY AT 4 MSB OF THE BYTE
FEB4 CDF4FD              CALL    CONV              ; CONVERT IT TO 1 ASCII CHARACTER
FEB7 CDE9FC              CALL    PO                ; PUNCH IT
FEBA 7B                  MOV     A,E               ; NOW LOOK ONLY AT 4 LSB OF BYTE
FEBB CDF4FD              CALL    CONV              ; CONVERT IT TO ONE ASCII CHAR
FEBE CDE9FC              CALL    PO                ; PUNCH IT
FEC1 7B                  MOV     A,E
FEC2 82                  ADD     D                 ; UPDATE THE RUNNING CHECKSUM
```

```
 LOC  OBJ          LINE        SOURCE STATEMENT

 FEC3 57           2687         MOV     D,A              ; STORE IT BACK IN THE D-REG
 FEC4 C9           2688         RET                      ; RETURN
                   2689 ;//////////////////////////////////////////////////////////////////////////
                   269Ø ; 'PCHK' - ENTERED VIA CALLS FROM 'G','S','X' COMMANDS AND 'PARAM' ROUTINE
                   2691 ; 'P2C' - ENTERED VIA CALLS FROM 'PARAM','PAØ' ROUTINES
                   2692 ; PROCESS: TEST FOR NULL INPUT PARAMETER (LOOK FOR SPACE,COMMA,OR <CR>)
                   2693 ; INPUT:
                   2694 ; OUTPUT: CHARACTER IN A-REG
                   2695 ;         IF SPACE OR COMMA, THEN ZERO = 1 AND CARRY = Ø
                   2696 ;         IF <CR>,           THEN ZERO = 1 AND CARRY = 1
                   2697 ;         IF NONE OF ABOVE,  THEN ZERO = Ø AND CARRY = Ø
                   2698 ; MODIFIED: A, FLAGS
                   2699 ; STACK USAGE: 4 BYTES
                   27ØØ PCHK:
 FEC5 CD61FF       27Ø1         CALL    TI               ; GET A CHARACTER
                   27Ø2 P2C:
 FEC8 FE2Ø         27Ø3         CPI     ' '
 FECA C8           27Ø4         RZ                       ; IF SPACE, THEN  ZERO = 1 & CARRY = Ø
 FECB FE2C         27Ø5         CPI     ','
 FECD C8           27Ø6         RZ                       ; IF COMMA, THEN ZERO = 1 & CARRY = Ø
 FECE FEØD         27Ø7         CPI     CR
 FEDØ 37           27Ø8         STC
 FED1 C8           27Ø9         RZ                       ; IF <CR>, THEN ZERO = 1 & CARRY = 1
 FED2 3F           271Ø         CMC
 FED3 C9           2711         RET                      ; IF NONE OF THE THREE, THEN ZERO=CARRY=Ø
                   2712 ;//////////////////////////////////////////////////////////////////////////
                   2713 ;/ 'RESTART' - ENTERED VIA JUMP FROM LOCATION Ø                             /
                   2714 ;/ PROCESS: BREAKPOINT/INTERRUPT/RESTART PROCESSING                        /
                   2715 ;/ INPUT:                                                                  /
                   2716 ;/ OUTPUT:                                                                 /
                   2717 ;/ MODIFIED:                                                               /
                   2718 ;/ EXPLANATION:                                                            /
                   2719 ;/ THIS ROUTINE IS ENTERED VIA A RESTART Ø (RST Ø) INSTRUCTION.  THE       /
                   272Ø ;/ INSTRUCTION IS ENCOUNTERED EITHER IN THE USER PROGRAM (AS A BREAKPOINT)  /
                   2721 ;/ OR IS INPUT VIA A LOCAL CONSOLE INTERRUPT (I.E. USER HAS ACTIVATED THE   /
                   2722 ;/ INTERRUPT Ø SWITCH).  THIS ROUTINE SAVES THE STATE OF THE CALLING        /
                   2723 ;/ PROCESS AND TURNS CONTROL OVER TO THE MONITOR.  THIS IS DONE IN THE     /
                   2724 ;/ FOLLOWING MANNER:                                                       /
                   2725 ;/    1. THE USER ENVIRONMENT IS SAVED BY PUSHING THE REGISTERS ON TOP      /
                   2726 ;/       OF THE USER'S OWN WORK STACK.                                      /
                   2727 ;/    2. PROGRAM THE 8259 WITH THE MONITOR'S OWN INTERRUPT MASK REGISTER.   /
                   2728 ;/    3. THE MONITOR'S EXIT TEMPLATE IS FOUND AND THE REGISTER VALUES       /
                   2729 ;/       REPRESENTING THE USER'S STATE ARE POPPED OFF THE USER WORK STACK   /
                   273Ø ;/       AND STORED IN THE APPROPRIATE PLACES IN THE EXIT TEMPLATE.         /
                   2731 ;/    4. TEST TO SEE IF THE POINT AT WHICH USER PROGRAM INTERRUPTION        /
                   2732 ;/       OCCURRED (VALUE OF PROGRAM COUNTER) COINCIDES WITH A BREAKPOINT    /
                   2733 ;/       ADDRESS.                                                           /
                   2734 ;/       A. IF IT DOESN'T, THEN RESTART CODE WAS ENTERED VIA A CONSOLE      /
                   2735 ;/          INTERRUPT SO SEND EOI TO THE 8259.                              /
                   2736 ;/       B. IF IT DOES, THEN PROGRAM THE EXIT CODE TO 1) LOAD THE CORRECT   /
                   2737 ;/          H AND L VALUES AND TO 2) JUMP TO THE ADDRESS INDICATED BY THE PC/
                   2738 ;/          (PUSHED ON STACK AT TIME OF RST Ø INSTRUCTION OR WHEN CONSOLE   /
                   2739 ;/          INTERRUPT). ALSO, RESTORE THE TRAP VALUES AT THE PROPER         /
                   274Ø ;/          TRAP ADDRESSES.                                                 /
                   2741 ;/    5. RETURN CONTROL TO THE MONITOR (BY JUMPING TO START).               /
```

```
LOC  OBJ         LINE        SOURCE STATEMENT

                 ;/                                                                         /
                 ;///////////////////////////////////////////////////////////////////////////
                 RESTART:
FED4 F3                  DI                          ; DISABLE IF SOFTWARE TRAP
                                                     ; SAVE USER'S ENVIRONMENT
FED5 E5                  PUSH    H                   ; SAVE H,L
FED6 D5                  PUSH    D                   ; SAVE D,E
FED7 C5                  PUSH    B                   ; SAVE B,C
FED8 F5                  PUSH    PSW                 ; SAVE A,FLAGS
FED9 D1                  POP     D                   ; TEMPORARILY SAVE PSW IN D & E
FEDA E5                  PUSH    H                   ; DUMMY PUSH TO RESERVE SPACE IN STACK FOR
                                                     ;    CURRENT INTERRUPT MASK AND CONFIGURATION
                                                     ;    BYTE
FEDB 2AØ4ØØ              LHLD    MEMTOP
FEDE 2ECC                MVI     L,ILOC-1 AND ØFFH; HL NOW POINTS TO CONFIGURATION BYTE IN
                                                     ;    EXIT CODE IN TOP PAGE OF RAM
FEEØ 6E                  MOV     L,M                 ; L NOW CONTAINS THIS CONFIGURATION BYTE
FEE1 DBFC                IN      SOCP1               ; INPUT CURRENT INTERRUPT MASK REGISTER ---
                                                     ;    THIS MASK IS THE USER'S, SO SAVE IT
FEE3 67                  MOV     H,A                 ; H NOW CONTAINS THIS INTERRUPT MASK
FEE4 E3                  XTHL                        ; THE INTERRUPT MASK AND CONFIGURATION BYTE
                                                     ;    ARE NOW ON TOP OF THE USER STACK
FEE5 D5                  PUSH    D                   ; NOW PUT THE ORIGINAL PSW ON TOP OF THE STACK
FEE6 3EFE                MVI     A,NOT INTØ          ; SET MONITOR'S DEFAULT INTERRUPT MASK
FEE8 D3FC                OUT     SOCP1               ; OUTPUT NEW MASK
FEEA 2AØ4ØØ              LHLD    MEMTOP
FEED 2ED2                MVI     L,EXIT AND ØFFH     ; HL NOW POINTS TO EXIT CODE AT TOP OF RAM
FEEF EB                  XCHG                        ; SO NOW DE POINTS TO EXIT CODE AT TOP OF RAM
FEFØ 21ØCØØ              LXI     H,12                ; H := ØØ, L := ØC  (DECIMAL VALUE 12)
FEF3 39                  DAD     SP                  ; EFFECT OF THIS IS TO CUT BACK THE USER'S
                                                     ;    STACK TO WHAT IT WAS BEFORE ENTERING
                                                     ;    THIS RESTART ROUTINE AND BEFORE THE PC
                                                     ;    WAS PUSHED ON BY RST Ø OR INTERRUPT.
                                                     ;    HL CONTAINS THIS 'OLD' STACK ADDRESS.
FEF4 Ø6Ø5                MVI     B,5                 ; COUNT FOR TRANSFER OF MACHINE STATE
                                                     ;    TO EXIT TEMPLATE STORAGE (MOVE THE STACK)
FEF6 EB                  XCHG                        ; HL NOW POINTS TO EXIT CODE AT TOP OF RAM
                                                     ; DE NOW POINTS TO USER STACK AS IT WAS
                                                     ;    PRIOR TO RST Ø OR CONSOLE INTERRUPT.
                 ;       -------------------------------------------------------------------------
                 RSTØ:                               ; MOVE THE MACHINE STATE FROM THE USER'S STACK
                                                     ; TO THE RESERVED AREA IN THE EXIT TEMPLATE
                                                     ; IN TOP PAGE OF RAM.
                                                     ;    B=5      !  B=4  !   B=3  !  B=2 ! B=1
                                                     ;------------!--------!--------!------!------
FEF7 2B                  DCX     H                   ;            !        !        !      !
FEF8 72                  MOV     M,D                 ;SLOC=MSB(SP)!ALOC=A  !ILOC=INT!BLOC=B!DLOC=D
FEF9 2B                  DCX     H                   ;            !        !        !      !
FEFA 73                  MOV     M,E                 ;    =LSB(SP)!FLOC=FLG!    =FLG!CLOC=C!ELOC=E
FEFB D1                  POP     D                   ;DE=AF       !DE=INT,F!DE=BC   !DE=DE !DE=HL
FEFC Ø5                  DCR     B                   ;B=4         !B=3     !B=2     !B=1   !B=Ø
FEFD C2F7FE              JNZ     RSTØ
                 ;       -------------------------------------------------------------------------
                                                     ; AT THIS POINT, HL POINTS TO THE BASE OF
                                                     ;    THE MONITOR STACK (TOS) IN TOP PAGE OF
```

```
 LOC  OBJ         LINE        SOURCE STATEMENT

                                                       ;    RAM. DE CONTAINS THE H & L VALUES THE
                                                       ;    USER HAD PRIOR TO ENTERING THE RESTART
                                                       ;    ROUTINE.
 FF00 C1                        POP     B              ; BC = OLD PC (PUSHED ON USER STACK BY
                                                       ;    RST 0 OR INTERRUPT)
 FF01 0B                        DCX     B              ; DECREMENT TO POINT AT TRAPPED CODE
 FF02 F9                        SPHL                   ; SP NOW POINTS TO TOS (BASE OF MONITOR STACK)
 FF03 2A0400                    LHLD    MEMTOP
 FF06 2EE2                      MVI     L,TLOC AND 0FFH ; HL NOW POINTS TO TLOC IN TOP PAGE OF RAM
                                                       ;    I.E. LSB OF TRAP 1 ADDRESS
 FF08 7E                        MOV     A,M            ; TEST IF THIS IS A PROGRAMMED RESTART OR A
 FF09 91                        SUB     C              ;    LOCAL CONSOLE INTERRUPT BY COMPARING THE
                                                       ;    PC VALUE WITH TRAP 1 ADDRESS
                                                       ;    A := LSB OF TRAP 1 ADDRESS
 FF0A 23                        INX     H              ; HL POINTS TO MSB OF TRAP 1 ADDRESS
 FF0B C213FF                    JNZ     RSTA           ; PC DID NOT MATCH TRAP 1 ADDRESS
 FF0E 7E                        MOV     A,M            ; A := MSB OF TRAP 1 ADDRESS
 FF0F 98                        SBB     B
 FF10 CA25FF                    JZ      RST1           ; PC MATCHES TRAP 1 --- A PROGRAMMED RESTART
                        RSTA:                          ; REPEAT SAME STEPS AS ABOVE BUT SEE IF PC
                                                       ;    MATCHES 2ND BREAKPOINT (TRAP 2 ADDRESS)
 FF13 23                        INX     H              ; HL POINTS TO TRAP 1 OPCODE VALUE
 FF14 23                        INX     H              ; HL POINTS TO LSB OF TRAP 2 ADDRESS
 FF15 7E                        MOV     A,M            ; A := LSB OF TRAP 2 ADDRESS
 FF16 91                        SUB     C
 FF17 23                        INX     H              ; HL POINTS TO MSB OF TRAP 2 ADDRESS
 FF18 C220FF                    JNZ     RSTB           ; PC DID NOT MATCH TRAP 2 ADDRESS
 FF1B 7E                        MOV     A,M            ; A := MSB OF TRAP 2 ADDRESS
 FF1C 98                        SBB     B
 FF1D CA25FF                    JZ      RST1           ; PC MATCHES TRAP 2 --- A PROGRAMMED RESTART
                        RSTB:                          ; NOT A PROGRAMMED RESTART, BUT A
 FF20 3E20                      MVI     A,EOI          ;    CONSOLE INTERRUPT SO SEND EOI TO 8259
 FF22 D3FD                      OUT     SOCP0
 FF24 03                        INX     B              ; ADJUST PC FOR LOCAL CONSOLE RESTART
                                                       ;    I.E. GET READY TO POINT PC TO
                                                       ;    RESUMPTION POINT IN CODE IT WAS
                                                       ;    EXECUTING WHEN INTERRUPTED
                                                       ;    BC POINTS TO NEXT INSTR TO BE EXECUTED
                                                       ;    WHEN CONTROL IS RETURNED TO USER PROGRAM
                        RST1:                          ; PROGRAMMED RESTART AT A BREAKPOINT (TRAP)
                                                       ;    ALSO FALLTHROUGH FROM CONSOLE INTERRUPT
 FF25 2A0400                    LHLD    MEMTOP
 FF28 2EDC                      MVI     L,LLOC AND 0FFH ; HL NOW POINTS TO LLOC IN EXIT CODE IN TOP OF RAM
 FF2A 73                        MOV     M,E            ; USER'S L VALUE PRIOR TO RESTART IS STORED IN LLOC
 FF2B 23                        INX     H
 FF2C 72                        MOV     M,D            ; USER'S H VALUE PRIOR TO RESTART IS STORED IN HLOC
                        ;       ----------------------------------------------------------------
 FF2D 2EE0                      MVI     L,PLOC-1 AND 0FFH; HL POINTS TO LSB OF JMP INSTR IN EXIT CODE
 FF2F 71                        MOV     M,C            ; SAVE LSB OF USER'S PC
 FF30 23                        INX     H
 FF31 70                        MOV     M,B            ; SAVE MSB OF USER'S PC.  EFFECT IS TO LOAD THE
                                                       ;    PROPER ADDRESS INTO THE EXIT TEMPLATE FOR THE
                                                       ;    JUMP BACK TO THE USER'S PROGRAM.
                        ;       ----------------------------------------------------------------
 FF32 C5                        PUSH    B
```

```
LOC  OBJ          LINE        SOURCE STATEMENT

FF33 CDDEFC              CALL    COMC
FF36 23                  DB      '#'
FF37 E1                  POP     H                 ; RETRIEVE OLD PC FOR DISPLAY
FF38 CD56FE              CALL    LADR              ; DISPLAY PC
                  ;      ------------------------------------------------------------
                                                   ; CLEAR TRAPS
FF3B 2AØ4ØØ              LHLD    MEMTOP
FF3E 2EE2                MVI     L,TLOC AND ØFFH ; HL NOW POINTS TO TLOC IN TOP PAGE OF RAM
FF4Ø 16Ø2                MVI     D,2               ; SET COUNT FOR TWO TRAPS
                  RST2:
FF42 4E                  MOV     C,M               ; C := LSB OF TRAP ADDRESS
FF43 AF                  XRA     A
FF44 77                  MOV     M,A               ; ZERO OUT LSB OF TRAP ADDRESS
FF45 23                  INX     H
FF46 46                  MOV     B,M               ; B := MSB OF TRAP ADDRESS
FF47 77                  MOV     M,A               ; ZERO OUT MSB OF TRAP ADDRESS
FF48 23                  INX     H                 ; HL NOW POINTS TO TRAP VALUE
FF49 79                  MOV     A,C               ; BC CONTAINS THE TRAP ADDRESS
FF4A BØ                  ORA     B                 ; TEST FOR VALID TRAP
FF4B CA5ØFF              JZ      RST3              ; TRAP ADDRESS IS Ø, SO NO TRAP TO RESTORE
FF4E 7E                  MOV     A,M               ; GET OPCODE BYTE, I.E. TRAP VALUE
FF4F Ø2                  STAX    B                 ; PUT IT BACK IN CORRECT PLACE IN USER PROGRAM,
                                                   ;    I.E. REPLACE THE RST Ø INSTR WITH ORIGINAL
                                                   ;    OPCODE.
                  RST3:
FF5Ø 23                  INX     H                 ; POINT TO TRAP 2 ADDRESS IF D=2
FF51 15                  DCR     D
FF52 C242FF              JNZ     RST2              ; REPEAT FOR TRAP 2
FF55 C355F8              JMP     START             ; ENTER MONITOR (INTERRUPTS STILL DISABLED)
                  ;/////////////////////////////////////////////////////////////////////////
                  ; 'RIX' - ENTERED VIA CALLS FROM 'R' COMMAND AND 'BYTE' ROUTINE
                  ; PROCESS: GET A CHARACTER FROM READER, MASK OFF PARITY BIT
                  ; INPUT:
                  ; OUTPUT: CHARACTER IN A-REG, BIT 7 IS Ø
                  ; MODIFIED: A,F
                  ; STACK USAGE:
                  RIX:
FF58 CDØFFC              CALL    RI                ; GET CHARACTER FROM READER DEVICE
FF5B DA47F8              JC      ERROR             ; READER TIMEOUT ERROR
FF5E E67F                ANI     7FH               ; MASK OUT THE PARITY BIT
FF6Ø C9                  RET                       ; RETURN
                  ;/////////////////////////////////////////////////////////////////////////
                  ; 'TI' - ENTERED VIA CALLS FROM 'A','N','Q' COMMANDS AND 'START','PARAM'
                  ;         'PAØ','PCHK' ROUTINES
                  ;         ENTERED VIA JUMP FROM 'BREAK'
                  ; PROCESS: INPUT FROM LOCAL CONSOLE, ECHO, RETURN IN A-REG
                  ; INPUT:
                  ; OUTPUT: CHARACTER IN A-REG
                  ; MODIFIED: A,F
                  ; STACK USAGE:
                  TI:
FF61 C5                  PUSH    B                 ; SAVE STATE OF B- & C-REGS
FF62 CDBEFB              CALL    CI                ; GET A CHARACTER FROM THE CONSOLE
FF65 E67F                ANI     7FH               ; MASK OFF PARITY BIT
FF67 CD76FF              CALL    UC                ; CONVERT TO UPPER CASE
```

```
LOC  OBJ          LINE        SOURCE STATEMENT

FF6A FE03                  CPI      ETX            ; TEST FOR BREAK
FF6C CA47F8                JZ       ERROR          ; ABORT COMMAND
FF6F 4F                    MOV      C,A            ; MOVE INPUT CHARACTER TO C-REG
FF70 CD9FFC                CALL     CO             ; ECHO IT
FF73 79                    MOV      A,C
FF74 C1                    POP      B              ; RESTORE STATE OF B & C
FF75 C9                    RET                     ; RETURN
                   ;///////////////////////////////////////////////////////////////////////////
                   ; 'UC' - ENTERED VIA CALL FROM 'TI' ROUTINE
                   ; PROCESS: CONVERT CHARACTER IN A-REG FROM LOWER CASE TO UPPER CASE
                   ; INPUT: LOWER OR UPPER CASE CHAR IN A-REG
                   ; OUTPUT: UPPER CASE CHARACTER IN A-REG
                   ; MODIFIED: A,F
                   ; STACK USAGE:
                   UC:
FF76 FE61                  CPI      'A'+20H
FF78 F8                    RM                      ; CHAR < LC(A) , I.E. IF THE CHAR IN A-REG
                                                   ;    IS NOT LOWER CASE, THEN IT HAS VALUE
                                                   ;    < 61H, SO A - 61H WILL BE MINUS. IF
                                                   ;    IT IS IN LOWER CASE, THE RESULT WILL
                                                   ;    BE POSITIVE.
FF79 FE7B                  CPI      'Z'+20H+1
FF7B F0                    RP                      ; CHAR > LC(Z) , I.E. WE KNOW THE A-REG IS
                                                   ;    UPPER CASE OR SPECIAL CHAR. IF IT IS A
                                                   ;    SPECIAL CHAR, A - 78H WILL BE 0 OR
                                                   ;    GREATER SO RETURN.
FF7C E6DF                  ANI      NOT 20H        ; FORCE UPPER CASE
FF7E C9                    RET
                   ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
                   ;*                                                                  *
                   ;*      I/O CONTROLLER INTERFACE DRIVERS                            *
                   ;*                                                                  *
                   ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
                   ; 'IOCDR1' - ENTERED VIA CALLS FROM 'CI','CSTS' ROUTINES
                   ; PROCESS: GET DEVICE STATUS OR GET DATA FROM PERIPHERAL
                   ; INPUT: B CONTAINS THE COMMAND (STATUS REQUEST OR INPUT DATA REQUEST)
                   ; OUTPUT: A CONTAINS THE REQUESTED INFORMATION
                   ; MODIFIED: A,FLAGS,B
                   ; STACK USAGE:
                   IOCDR1:
FF7F CDA6FF                CALL     IOCCOM         ; OUTPUT 'GET DEVICE STATUS COMMAND' OR
                                                   ;    'INPUT DATA COMMAND' TO IOC CONTROL
                                                   ;    PORT
                   IOCXXX:
FF82 DBC1                  IN       IOCS           ; INPUT DBB STATUS
FF84 E607                  ANI      IBF OR OBF OR F0; MASK OFF STATUS FLAGS
FF86 FE01                  CPI      OBF            ; TEST FOR SLAVE DONE; SOMETHING FOR THE MASTER
FF88 C282FF                JNZ      IOCXXX         ; IF NOT, CONTINUE TO LOOP
FF8B DBC0                  IN       IOCI           ; OTHERWISE, INPUT THE DATA FROM THE DBB
FF8D F5                    PUSH     PSW            ; SAVE A-REG
FF8E 3E05                  MVI      A,ENABL        ; ENABLE INTERRUPTS
FF90 D3FF                  OUT      CPUC
FF92 F1                    POP      PSW            ; RESTORE A-REG
FF93 C9                    RET
                   ;---------------------------------------------------------------------------
```

```
 LOC  OBJ         LINE        SOURCE STATEMENT

                  ; 'IOCDR2' - ENTERED VIA CALLS FROM 'BLK','COM','CO','CRTOUT' ROUTINES
                  ; PROCESS: OUTPUT DATA TO THE PERIPHERAL DEVICE
                  ; INPUT: B CONTAINS THE COMMAND TO OUTPUT THE DATA
                  ;        C CONTAINS THE DATA TO BE OUTPUT
                  ; OUTPUT:
                  ; MODIFIED: A,FLAGS,B,C
                  ; STACK USAGE:
                  IOCDR2:
FF94 CDA6FF               CALL    IOCCOM          ; OUTPUT 'OUTPUT DATA COMMAND' TO IOC
                                                  ;    CONTROL PORT
                  IOCYYY:
FF97 DBC1                 IN      IOCS            ; INPUT DBB STATUS
FF99 E6Ø7                 ANI     IBF OR FØ OR OBF; TEST FOR SLAVE PROCESSOR READY
FF9B C297FF               JNZ     IOCYYY          ; CONTINUE TO LOOP UNTIL IT IS READY
FF9E 79                   MOV     A,C             ; LOAD DATA TO BE WRITTEN
FF9F D3CØ                 OUT     IOCO            ; OUTPUT DATA TO THE DBB
FFA1 3EØ5                 MVI     A,ENABL         ; ENABLE INTERRUPTS
FFA3 D3FF                 OUT     CPUC
FFA5 C9                   RET
                  ;-----------------------------------------------------------------------
                  ; 'IOCCOM' - COMMON ROUTINE TO IOC DRIVERS
                  ;            ENTERED VIA CALLS FROM 'IOCDR1' AND 'IOCDR2'
                  ; PROCESS: OUTPUT COMMAND TO THE IOC
                  ; INPUT: B CONTAINS THE COMMAND
                  ; OUTPUT:
                  ; MODIFIED: A,FLAGS
                  ; STACK USAGE:
                  IOCCOM:
FFA6 3EØD                 MVI     A,DISABL        ; BLOCK ALL INTERRUPTS
FFA8 D3FF                 OUT     CPUC
                  IOCZZZ:
FFAA DBC1                 IN      IOCS            ; INPUT DBB STATUS
FFAC E6Ø7                 ANI     FØ OR IBF OR OBF; TEST FOR SLAVE PROCESSOR IDLE
FFAE C2AAFF               JNZ     IOCZZZ          ; LOOP UNTIL IT IS IDLE
FFB1 78                   MOV     A,B             ; LOAD COMMAND
FFB2 D3C1                 OUT     IOCC            ; OUTPUT COMMAND TO IOC CONTROL PORT
FFB4 C9                   RET
                  ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
                  ;*                                                                   *
                  ;*                PARALLEL I/O INTERFACE DRIVERS                     *
                  ;*                                                                   *
                  ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
                  ; 'PIODR1' - ENTERED VIA CALLS FROM 'RI','PO','POC','LO','UPPS'
                  ; 'PIODR2' - ENTERED VIA CALLS FROM 'UI','UPPS' ROUTINES
                  ; PROCESS: GET DEVICE STATUS OR GET DATA FROM A PERIPHERAL
                  ; INPUT: B CONTAINS THE COMMAND (STATUS REQUEST OR INPUT DATA REQUEST)
                  ; OUTPUT: A CONTAINS THE REQUESTED INFORMATION
                  ; MODIFIED: A, FLAGS, B
                  ; STACK USAGE:
                  PIODR1:
FFB5 CDE4FF               CALL    PIOCOM          ; OUTPUT 'GET DEVICE STATUS COMMAND' OR
                                                  ;    'INPUT DATA COMMAND' OR OTHER SUCH
                                                  ;    COMMAND TO THE PIO CONTROL PORT
                  PIODR2:
FFB8 3EØD                 MVI     A,DISABL        ; BLOCK ALL INTERRUPTS
```

```
 LOC  OBJ          LINE        SOURCE STATEMENT

 FFBA D3FF         3Ø17          OUT     CPUC
 FFBC DBF9         3Ø18          IN      PIOS             ; INPUT DBB STATUS
 FFBE E6Ø7         3Ø19          ANI     FØ OR IBF OR OBF; MASK OFF STATUS FLAGS
 FFCØ FEØ1         3Ø2Ø          CPI     OBF              ; TEST FOR SLAVE DONE; SOMETHING FOR THE MASTER
 FFC2 C2B8FF       3Ø21          JNZ     PIODR2           ; LOOP UNTIL SLAVE IS READY
 FFC5 DBF8         3Ø22          IN      PIOI             ; OTHERWISE INPUT THE DATA FROM THE DBB
 FFC7 F5           3Ø23          PUSH    PSW              ; SAVE A-REG
 FFC8 3EØ5         3Ø24          MVI     A,ENABL          ; ENABLE INTERRUPTS
 FFCA D3FF         3Ø25          OUT     CPUC
 FFCC F1           3Ø26          POP     PSW              ; RESTORE A-REG
 FFCD C9           3Ø27          RET
                   3Ø28 ;---------------------------------------------------------------------
                   3Ø29 ; 'PIODR3' - ENTERED VIA CALLS FROM 'POC','PO','LO','UI','UO' ROUTINES
                   3Ø3Ø ; 'PIODR4' - ENTERED VIA CALLS FROM 'UI','UO'
                   3Ø31 ; PROCESS: OUTPUT DATA TO A PERIPHERAL DEVICE
                   3Ø32 ; INPUT: B CONTAINS THE COMMAND TO OUTPUT THE DATA
                   3Ø33 ;        C CONTAINS THE DATA TO BE OUTPUT
                   3Ø34 ; OUTPUT:
                   3Ø35 ; MODIFIED: A,FLAGS,B, C
                   3Ø36 ; STACK USAGE:
                   3Ø37 PIODR3:
 FFCE CDE4FF       3Ø38          CALL    PIOCOM           ; OUTPUT 'OUTPUT DATA COMMAND' TO PIO
                   3Ø39 PIODR4:
 FFD1 3EØD         3Ø4Ø          MVI     A,DISABL         ; BLOCK ALL INTERRUPTS
 FFD3 D3FF         3Ø41          OUT     CPUC
 FFD5 DBF9         3Ø42          IN      PIOS             ; INPUT DBB STATUS
 FFD7 E6Ø7         3Ø43          ANI     FØ OR IBF OR OBF; TEST FOR SLAVE PROCESSOR READY
 FFD9 C2D1FF       3Ø44          JNZ     PIODR4           ; LOOP UNTIL IT IS READY
 FFDC 79           3Ø45          MOV     A,C              ; LOAD DATA TO BE WRITTEN
 FFDD D3F8         3Ø46          OUT     PIOO             ; OUTPUT DATA TO THE DBB
 FFDF 3EØ5         3Ø47          MVI     A,ENABL          ; ENABLE INTERRUPTS
 FFE1 D3FF         3Ø48          OUT     CPUC
 FFE3 C9           3Ø49          RET
                   3Ø5Ø ;-----------------------------------------------------------------
                   3Ø51 ; 'PIOCOM' - COMMON ROUTINE OF PIO DRIVERS
                   3Ø52 ;             ENTERED VIA CALLS FROM 'PIODR1', 'PIODR3', 'RI' ROUTINES
                   3Ø53 ; INPUT: B CONTAINS THE COMMAND
                   3Ø54 ; OUTPUT:
                   3Ø55 ; MODIFIED: A,FLAGS
                   3Ø56 ; STACK USAGE:
                   3Ø57 PIOCOM:
 FFE4 3EØD         3Ø58          MVI     A,DISABL         ; BLOCK ALL INTERRUPTS
 FFE6 D3FF         3Ø59          OUT     CPUC
                   3Ø6Ø PIOZZZ:
 FFE8 DBF9         3Ø61          IN      PIOS             ; INPUT DBB STATUS
 FFEA E6Ø7         3Ø62          ANI     FØ OR IBF OR OBF; TEST FOR SLAVE PROCESSOR IDLE
 FFEC C2E8FF       3Ø63          JNZ     PIOZZZ           ; LOOP UNTIL IT IS IDLE
 FFEF 78           3Ø64          MOV     A,B              ; LOAD THE COMMAND
 FFFØ D3F9         3Ø65          OUT     PIOC             ; OUTPUT THE COMMAND TO THE PIO CONTROL PORT
 FFF2 3EØ5         3Ø66          MVI     A,ENABL          ; ENABLE INTERRUPTS
 FFF4 D3FF         3Ø67          OUT     CPUC
 FFF6 C9           3Ø68          RET
                   3Ø69 ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-
 FFFD              3Ø7Ø          ORG     ØFFFDH
 FFFD 6C           3Ø71 MNCKSM: DB       Ø6CH             ; CHKSUM MONITOR TO Ø1EH
```

```
  LOC  OBJ         LINE        SOURCE STATEMENT

  FFFE ØØ          3Ø72          DB      ØØ              ; UNUSED BYTE
  FFFF Ø1          3Ø73          DB      Ø1              ; Ø, IF SERIES I MONITOR
                   3Ø74                                  ; 1, IF SERIES II MONITOR
                   3Ø75 ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
                   3Ø76 ;
                   3Ø77 ; END OF PROGRAM
                   3Ø78 ;
                   3Ø79 ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
                   3Ø8Ø          END

PUBLIC SYMBOLS


EXTERNAL SYMBOLS


USER SYMBOLS
@USER  A FC8C   ACHRM  A ØØ7F   ACT    A F913   ACTBL  A FB81   ALOC   A EACF   ALT    A F92B   ALUP1  A F8D3
ALUP2  A F8DB   ALUP3  A F8F2   APT    A F923   ART    A F91B   ASØ    A F8BE   AS1    A F8CD   AS2    A F8E5
AS3    A F8F2   ASSIGN A F8B6   BØ11Ø  A Ø2BA   B24ØØ  A ØØ2Ø   B96ØØ  A ØØØ7   BASE   A F8ØØ   BATCH  A ØØØ2
BBASE  A E8ØØ   BCDC   A ØØØ1   BDLY   A EA23   BDLY1  A EA25   BEGIN  A F8ØØ   BLK    A FC93   BLOC   A EACB
BOVROF A ØØØ1   BOVRON A ØØØ9   BREAK  A FCØ7   BSØ    A E8Ø6   BS1    A E83B   BS1Ø   A E913   BS11   A E92Ø
BS12   A E94E   BS13   A E959   BS14   A E97B   BS1X   A E84F   BS2    A E862   BS3    A E86A   BS4    A E8AC
BS5    A E8CØ   BS6    A E8C6   BS7    A E8DC   BS8    A E8E2   BS9    A E8EA   BSX1   A E98E   BSX1Ø  A EA18
BSX2   A E9A4   BSX3   A E9B2   BSX4   A E9BD   BSX5   A E9CC   BSX6   A E9EA   BSX8   A E9FE   BSX9   A EAØD
BTCKSM A EA64   BTDGOF A ØØØ4   BTDGON A ØØØC   BYTE   A FDDB   CCRT   A ØØØ1   CI     A FBBE   CIØ    A FBDØ
CI1    A FBE1   CI2    A FBEB   CI3    A FBEC   CI4    A FBFD   CILOC  A EAE8   CL5    A ØØØØ   CL6    A ØØØ4
CL7    A ØØØ8   CL8    A ØØØC   CLERR  A ØØ1Ø   CLOC   A EACA   CMSK   A ØØFC   CNOTD  A ØØØ8   CO     A FC9F
COØ    A FCB2   COLOC  A EAEB   COM    A FC95   COMC   A FCDE   COMD   A ØØ25   CONC   A ØØC1   CONI   A ØØCØ
CONO   A ØØCØ   CONS   A ØØC1   CONV   A FDF4   COP    A F8Ø9   CPUC   A ØØFF   CPUS   A ØØFE   CR     A ØØØD
CRLF   A FDFE   CRTC   A ØØ1Ø   CRTOT1 A FCCA   CRTOT2 A FCD5   CRTOUT A FCBE   CRTS   A ØØ11   CSØ    A FD53
CS1    A FD6B   CS2    A FD74   CS3    A FD79   CSLOC  A EAFD   CSMEM  A ØØØ8   CSTS   A FD44   CTBL   A F882
CTRØP  A ØØFØ   CTRØS  A ØØØØ   CTR1P  A ØØF1   CTR1S  A ØØ4Ø   CTR2P  A ØØF2   CTR2S  A ØØ8Ø   CTTY   A ØØØØ
CUSE   A ØØØ3   DADR   A FEØ7   DATE   A Ø1Ø3   DBYTE  A FEØC   DECHO  A ØØØ7   DELAY  A FE1E   DIØ    A F938
DI1    A F93E   DI2    A F956   DIAGBT A EBØ3   DIAGMN A EBØØ   DISABL A ØØØD   DISAXP A ØØØØ   DISP   A F933
DLOC   A EAC9   DLY1   A FE2Ø   DPRNT  A ØØØ8   DREG   A FE25   DSR    A ØØ8Ø   DSTAT  A ØØØ3   DSTS   A ØØ78
DTR    A ØØØ2   ELOC   A EAC8   ENABL  A ØØØ5   ENAXP  A ØØØ8   ENDX   A EBØØ   ENHM   A ØØ8Ø   EOF    A F95F
EOI    A ØØ2Ø   ERESET A ØØØ1   ERMSG  A EA56   ERROR  A F847   ETX    A ØØØ3   EXØ    A FE46   EXIT   A EAD2
EXPR   A FE39   FØ     A ØØØ4   FALSE  A ØØØØ   FDOC   A ØØØ4   FIØ    A F984   FILL   A F97D   FLOC   A EACE
FRDY   A ØØØ1   FSTOP  A ØØE7   FSTP   A ØØF7   GOØ    A F9A4   GO1    A F9AA   GO2    A F9BA   GO3    A F9C2
GO4    A F9D1   GOTO   A F98C   HEXN   A F9D5   HI     A ØØ7A   HILO   A FE4C   HLOC   A EADD   HMSK   A ØØFF
HXD    A FE64   IBF    A ØØØ2   ICFG   A ØØ41   ICNP   A ØØØ1   ICRTI  A ØØ2Ø   ICRTO  A ØØ1Ø   ICW1   A ØØ12
ICW2   A ØØØØ   IICPØ  A ØØFB   IICP1  A ØØFA   ILOC   A EACD   ILPT   A ØØ4Ø   INIT   A E8Ø3   INITIO A ØØØ6
INTØ   A ØØØ1   INT1   A ØØØ2   INT2   A ØØØ4   INT3   A ØØØ8   INT4   A ØØ1Ø   INT5   A ØØ2Ø   INT6   A ØØ4Ø
INT7   A ØØ8Ø   INTA   A ØØØØ   IOBYT  A ØØØ3   IOCC   A ØØC1   IOCCOM A FFA6   IOCDP1 A F821   IOCDP2 A F844
IOCDR1 A FF7F   IOCDR2 A FF94   IOCHK  A FD83   IOCI   A ØØCØ   IOCO   A ØØCØ   IOCPØ  A ØØFB   IOCP1  A ØØFA
IOCS   A ØØC1   IOCXXX A FF82   IOCYYY A FF97   IOCZZZ A FFAA   IODEF  A FD94   IOPB   A EA34   IOSET  A FD87
IPTP   A ØØØ4   IPTR   A ØØØ8   ITCP   A ØØF3   ITIMO  A ØØFF   ITTYI  A ØØØ2   ITTYO  A ØØØ1   KEYC   A ØØ12
KINT   A ØØ14   KRDY   A ØØØ1   KSTS   A ØØ13   L1LOC  A EAFA   LADR   A FE56   LBMK   A ØØFF   LBYTE  A FE5B
LCRLF  A FE6A   LCRT   A ØØ4Ø   LCT    A ØØ1A   LCTR   A ØØØØ   LERM   A ØØØE   LF     A ØØØA   LLOC   A EADC
LLPT   A ØØ8Ø   LMSK   A ØØ3F   LO     A FD1E   LOM    A FD14   LOWW   A ØØ79   LPØ    A FD33   LPTC   A ØØ14
LPTRY  A ØØØ1   LSTC   A ØØ15   LSTE   A ØØ4Ø   LTBL   A F9Ø3   LTTY   A ØØØØ   LUSE   A ØØCØ   LVER   A ØØ1B
MEMCHK A FD8C   MEMTOP A ØØØ4   MENB   A ØØ8Ø   MLP    A E9Ø2   MNCKSM A FFFD   MODEØ  A ØØØØ   MODE1  A ØØØ2
MODE2  A ØØØ4   MODE3  A ØØØ6   MODE4  A ØØØ8   MODE5  A ØØØA   MOVBOT A ØØØ2   MOVE   A F9FØ   MVØ    A F9F7
```

```
ISIS-II 8Ø8Ø/8Ø85 MACRO ASSEMBLER, V3.Ø        MODULE   PAGE   59
 INTELLEC SERIES II MONITOR, VERSION 1.3, 1 MARCH 1979

NIØ    A FEA6   NIBBLE A FE98   NLEADX A FAØB   NREGS  A ØØØC   NUØ    A FAØ9   NULL   A FAØ1   OBF    A ØØØ1
OCW3   A ØØØB   ONEMS  A ØØ7Ø   OPCPL  A ØØØ4   P1LOC  A EAF4   P2C    A FEC8   P2LOC  A EAF7   PAØ    A FE7A
PA1    A FE7D   PA2    A FE9Ø   PACIFY A ØØØØ   PADR   A FEAA   PARAM  A FE74   PARML  A ØØØ4   PBYTE  A FEAF
PCHK   A FEC5   PCOMP  A ØØØ2   PENB   A ØØ1Ø   PEVEN  A ØØ2Ø   PGRDY  A ØØØ1   PIOC   A ØØF9   PIOCOM A FFE4
PIODR1 A FFB5   PIODR2 A FFB8   PIODR3 A FFCE   PIODR4 A FFD1   PIOI   A ØØF8   PIOO   A ØØF8   PIOS   A ØØF9
PIOZZZ A FFE8   PLOC   A EAE1   PMSK   A ØØCF   PNIB   A ØØ1Ø   PO     A FCE9   POØ    A FCF7   PO1    A FDØ8
POC    A FCE5   PPTP   A ØØ1Ø   PRTM   A EA2A   PSOCK  A ØØ2Ø   PSTC   A ØØ13   PTPRY  A ØØØ1   PTRADV A ØØ4Ø
PTRDY  A ØØØ1   PTRREV A ØØ6Ø   PTTY   A ØØØØ   PUNC   A ØØ12   PUSE1  A ØØ2Ø   PUSE2  A ØØ3Ø   QØ     A FA21
Q1     A FA3D   Q2     A FA48   QUERY  A FA14   R16X   A ØØØ2   R1LOC  A EAEE   R1X    A ØØØ1   R2LOC  A EAF1
R64X   A ØØØ3   RADCT  A ØØ28   RDBC   A ØØ19   RDBCC  A ØØ1A   RDRC   A ØØ1Ø   RDSTS  A ØØ1C   READ   A FA52
REDØ   A FA59   RED1   A FA7B   RED2   A FA93   RED3   A FA9E   RED4   A FAB3   RESET  A ØØØØ   RESTAR A FED4
RFR    A ØØ2Ø   RI     A FCØF   RIØ    A FC1F   RI1    A FC2C   RI2    A FC39   RI3    A FC47   RI4    A FC4C
RI4B   A FC4F   RI5    A FC58   RI6    A FC65   RI7    A FC79   RI8    A FC82   RIX    A FF58   RLLB   A ØØ1Ø
RLLM   A ØØ3Ø   RLMB   A ØØ2Ø   RMSK   A ØØF3   ROV    A ØØ1Ø   RPAR   A ØØØ8   RPPC   A ØØ17   RPSTC  A ØØ18
RPTR   A ØØØ4   RRDY   A ØØØ2   RRSTS  A ØØ1B   RSTØ   A FEF7   RST1   A FF25   RST2   A FF42   RST3   A FF5Ø
RSTA   A FF13   RSTB   A FF2Ø   RSTC   A ØØ11   RSTS   A ØØ7B   RTCC   A Ø4CD   RTOCT  A ØØFA   RTS    A ØØ2Ø
RTTY   A ØØØØ   RUSE1  A ØØØ8   RUSE2  A ØØØC   RXEN   A ØØØ4   SBCH   A ØØØ8   SICPØ  A ØØFD   SICP1  A ØØFC
SINT   A ØØØA   SLOC   A EAD1   SOCPØ  A ØØFD   SOCP1  A ØØFC   SRQ    A ØØØ6   SRQACK A ØØØ5   SRQDAK A ØØØ4
ST1    A ØØ4Ø   ST15   A ØØ8Ø   ST2    A ØØCØ   START  A F855   STARTØ A F851   SUØ    A FAC3   SU1    A FAD9
SUBS   A FABF   SYNC   A ØØØØ   SYND   A ØØ4Ø   SYSTAT A ØØØ2   TADV   A ØØ27   TI     A FF61   TLOC   A EAE2
TOS    A EAC8   TOUT   A ØØFA   TRAM   A ØØØ9   TRDY   A ØØØ1   TRKØ   A 3ØØØ   TRKL   A ØDØØ   TRUE   A FFFF
TTYC   A ØØF5   TTYI   A ØØF4   TTYIN  A FBC6   TTYO   A ØØF4   TTYOUT A FCA7   TTYS   A ØØF5   TXBE   A ØØØ4
TXEN   A ØØØ1   UC     A FF76   UCI    A ØØØØ   UCO    A ØØØ1   UCS    A ØØØ7   UI     A FDAD   UL1    A ØØØ6
UO     A FDBE   UP1    A ØØØ4   UP2    A ØØØ5   UPPS   A FDCE   UR1    A ØØØ2   UR2    A ØØØ3   USCC   A ØØF7
USCI   A ØØF6   USCO   A ØØF6   USCS   A ØØF7   USER   A EACØ   USRST  A ØØ4Ø   VER    A ØØØD   VERH   A ØØ13
VERS   A EA3B   WDBC   A ØØ17   WDBCC  A ØØ18   WPBC   A ØØ15   WPBCC  A ØØ16   WPPC   A ØØ16   WRØ    A FAE5
WR1    A FAED   WR2    A FAF8   WR3    A FBØ7   WRITE  A FADD   X      A FB26   XØ     A FB31   X1     A FB3F
X2     A FB42   X3     A FB5F   X4     A FB6Ø   X5     A FB6A   X6     A FB6D   XTBL   A EAE8   Z      A FBA6

ASSEMBLY COMPLETE,   NO ERRORS
```

```
@USER  1876   1981   1993#  2043   2135   2137   2164   2238
ACHRM   384#
ACT    1181   1195#
ACTBL  1710   1774#  1788
ALOC    882#  1775
ALT    1187   1210#
ALUP1  1139#  1142
ALUP2  1144#  1147
ALUP3  1160#  1163
APT    1185   1205#
ART    1183   1200#
AS0    1119#  1127
AS1    1122   1130#
AS2    1150#  1156
AS3    1153   1159#
ASSIGN 1080   1114#
B0110   212#   555
B2400   211#   544
B9600   210#
BASE    962#   963
BATCH   341#  1198   1873   2020   2039   2152   2233
BBASE   414#   415    870
BCDC    209#
BDLY    573    574    587    588    823#
BDLY1   825#   827
BEGIN   810    972#
BLK    1400   1724   1757   2015#
BLOC    598    651    716    751    876#  1776
BOVROF   87#   439
BOVRON   88#
BREAK  1885#  2021   2153
BS0     416    435#
BS1     475#   483
BS10    637#   640
BS11    614    649#
BS12    674#   679
BS13    681#   684
BS14    697#   701    708
BS1X    495#   503
BS2     488    511    516#
BS3     521#   526
BS4     568#   577
BS5     572    578#
BS6     582#   591
BS7     586    592#
BS8     576    590    596#
BS9     595    601#
BSX1    617    645    654    658    663    689    714#
BSX10   773    786    814#
BSX2    719    728#   740    745
BSX3    733    736#
BSX4    735    742#
BSX5    725    756#
BSX6    770    781#
BSX8    780    791#
BSX9    776    783    790    803#   818
BTCKSM  863#
```

```
BTDGOF   89#   796   1037
BTDGON   90#  1800
BYTE   1528   1534   1536   1538   1551   1558   1589   1594   1605   1606   2388#
CCRT    340#  1197   1837   2041   2200
CI      974   1822#  2904
CI0    1825   1836#
CI1    1851#  1854
CI2    1848   1859#
CI3    1861#  1865
CI4    1838   1872#
CILOC   911#   931   1875
CL5     144#
CL6     143#
CL7     142#
CL8     141#   549    560
CLERR   157#
CLOC    875#  1777
CMSK    334#  1180   1824   2019   2025   2151   2189
CNOTD   306#
CO      976   2023#  2910
CO0    2026   2038#
COLOC   913#   932   2042
COM    1469   1495   1762   2017#  2093   2586
COMC   1021   1042   1355   1470   1630   1727   1763   2085#  2431   2433   2852
COMD    178#  1923
CONC     78#
CONI     75#
CONO     76#
CONS     77#
CONV   2414#  2462   2465   2585   2680   2683
COP     394#   835
CPUC     96#   438    440    442    444    792    794   1038   1801   1909   1943   2958   2979   2991   3017   3025
       3041   3048   3059   3067
CPUS     95#
CR      328#   854    856    861    861   1045   1162   1441   1461   1694   1750   1798   2432   2595   2707
CRLF   1041   1382   1394   1467   1521   1653   1755   2430#
CRTC    233#  2072
CRTOT1 2058#  2061
CRTOT2 2055   2067#
CRTOUT 2047#  2161
CRTS    234#   579
CS0    2190   2199#
CS1    2209   2218#
CS2    2196   2215   2225#
CS3    2201   2232#
CSLOC   925#   938   2236
CSMEM   296#
CSTS    979   1886   2187#
CTBL   1057   1079#  1106
CTR0P   216#   557    559
CTR0S   196#   553
CTR1P   217#   546    548
CTR1S   197#   542
CTR2P   218#   459    461
CTR2S   198#   455
CTTY    339#  1196
CUSE    343#  1199
```

```
DADR    1232   2443#
DATE      56#   423    985
DBYTE   1237   2445   2456#
DECHO    295#
DELAY   1920   1930   1964   2474#
DIØ     1230#  1243
DI1     1233#  1242
DI2     1239   1244#
DIAGBT   764    943#
DIAGMN   941#  1802
DISABL    82#  1908   2990   3016   3040   3058
DISAXP    84#
DISP    1083   1226#
DLOC     874#  1778
DLY1    2476#  2478
DPRNT    391#   657
DREG    1726   1765   2489#
DSR      171#
DSTAT    288#
DSTS     315#   611    615    638    774
DTR      154#   177    551
ELOC     873#  1779
ENABL     83#   441   1942   2957   2978   3024   3047   3066
ENAXP     86#   443
ENDX     927#
ENHM     160#
EOF     1084   1261#
EOI      103#  2828
ERESET   286#
ERMSG    815    861#   862
ERROR   1017#  1048   1059   1081   1082   1088   1089   1090   1091   1094   1095   1099   1100   1101   1104   1128
        1157   1364   1442   1462   1563   1607   1722   1796   1799   2296   2520   2524   2609   2628   2890   2908
ETX      330#  2907
EXØ     2519   2522#
EXIT     886#  1335   2768
EXPR    1227   1263   1287   1393   1420   1520   1652   2513#  2523
FØ       305#   570    584    699   2952   2974   2994   3019   3043   3062
FALSE    320#   321   2226
FDOC     383#
FIØ     1291#  1294
FILL    1085   1285#
FLOC     881#  1780
FRDY     250#
FSTOP    381#   486    513
FSTP     382#
GOØ     1343   1351#
GO1     1354#  1362
GO2     1360   1363#
GO3     1368#  1380
GO4     1352   1381#
GOTO    1086   1333#
HEXN    1087   1392#
HI       313#   636
HILO    1238   1293   1428   1666   1697   2533#
HLOC     898#  1781   1784
HMSK     322#   527    883   1775   1776   1777   1778   1779   1780   1781   1782   1783   1784   1785   1786   1875
        1980   1982
```

```
HXD     2572   2584#
IBF      304#   570    584    699   2952   2974   2994   3019   3043   3062
ICFG     387#   724    737
ICNP     388#   599    752
ICRTI    374#
ICRTO    373#
ICW1     100#   445
ICW2     101#   448
IICP0    362#   447
IICP1    363#   450
ILOC     880#  1782   1841   2050   2204   2756
ILPT     375#
INIT     417#
INITIO   408#   759
INT0     107#   451    880   2765
INT1     108#
INT2     109#
INT3     110#
INT4     111#
INT5     112#
INT6     113#
INT7     114#
INTA     115#   453
IOBYT    404#   757   1164   1169   1482   1823   1902   2018   2024   2113   2150   2157   2188   2248   2260
IOCC     229#   580   2997
IOCCOM   991   2947   2970   2989#
IOCDP1   395#   656    660    682    686    693    721    788
IOCDP2   396#   671    677
IOCDR1   984   1863   1867   2221   2946#
IOCDR2   992   2073   2969#
IOCHK    980   2247#
IOCI     226#   593    702   2955
IOCO     227#  2977
IOCP0    364#
IOCP1    365#   454
IOCS     228#   569    583    698   2951   2973   2993
IOCXXX  2950#  2954
IOCYYY  2972#  2975
IOCZZZ  2992#  2995
IODEF    983   2289#
IOPB     623    668    844#
IOSET    981   2258#
IPTP     371#
IPTR     372#
ITCP     219#   456    543    554
ITIMO    385#   567    581
ITTYI    370#
ITTYO    369#
KEYC     235#  1866
KINT     237#
KRDY     249#  1864   2222
KSTS     236#   720   1862   2220
L1LOC    923#   937   2163
LADR    1399   1408   2552#  2855
LBMK     386#   598    651    716    751
LBYTE   1629   2500   2505   2554   2566#
LCRLF   1231   1245   2594#
```

```
LCRT    356#  1212   2160
LCT    1058   1106#
LCTR    199#
LERM    816    862#
LF      329#   854    856    861    861   1696   2434   2597
LLOC    897#  1783   2839
LLPT    357#  1213
LMSK    337#  1186   2158
LO      978   2040   2156#
LOM    1235   1247   2149#  2463   2466   2596   2598
LOWW    312#   634
LP0    2168#  2172
LPTC    270#  2173
LPTRY   278#  2171
LSTC    271#  2169
LSTE    389#
LTBL   1116   1179#  1465
LTTY    355#  1211
LUSE    358#  1214   2162
LVER    805    857#
MEMCHK  982   2270#
MEMTOP  406#   517    597    650    715    750   1018   1334   1346   1365   1598   1840   1995   2049   2203   2271
       2292   2492   2755   2767   2804   2838   2858
MENB    376#
MLP     625#   631
MNCKSM 3071#
MODE0   203#
MODE1   204#
MODE2   205#
MODE3   206#   455    542    553
MODE4   207#
MODE5   208#
MOVBOT   91#   437
MOVE   1092   1418#
MV0    1424#  1429
NI0    2646   2650#
NIBBLE 2391   2398   2614   2640#
NLEADX 1445#  1449
NREGS  1713   1788#
NU0    1276   1443#
NULL   1093   1439#
OBF     303#   570    584    585    699    700   2952   2953   2974   2994   3019   3020   3043   3062
OCW3    102#
ONEMS   326#   824   2475
OPCPL   311#   639    683
P1LOC   919#   935   2134
P2C    2627   2702#
P2LOC   921#   936   2136
PA0    1344   1636   1734   2610#
PA1    2612#  2624
PA2    2616   2625#
PACIFY  285#
PADR   1270   1675   2661#
PARAM  1357   1625   2514   2607#
PARML   393#   672
PBYTE  1268   1272   1275   1674   1677   1681   1687   2663   2674#
PCHK   1342   1632   1711   1729   2608   2700#
```

```
PCOMP   126#
PENB    145#
PEVEN   146#
PGRDY   127#
PIOC    260#  3065
PIOCOM 1957   3012   3038   3057#
PIODR1 1961   1970   2123   2170   2370   3011#
PIODR2 2333   2372   3015#  3021
PIODR3 2127   2174   2326   2351   3037#
PIODR4 2331   2354   2357   3039#  3044
PIOI    257#  3022
PIOO    258#  3046
PIOS    259#  3018   3042   3061
PIOZZZ 3060#  3063
PLOC    903#  1347   1599   1785   2844
PMSK    336#  1184   2114
PNIB    129#
PO      977   2112#  2681   2684
PO0    2121#  2125
PO1    2117   2132#
POC    1264   1446   1657   1693   1695   2105#
PPTP    351#  1207   2116
PRTM    806    817    833#   838
PSOCK   128#
PSTC    269#  2122
PTPRY   280#  2124
PTRADV  266#  1956
PTRDY   279#  1962
PTRREV  265#
PTTY    350#  1206
PUNC    268#  2126
PUSE1   352#  1208   2133
PUSE2   353#  1209
Q0     1466#  1499
Q1     1486#  1493
Q2     1490   1494#
QUERY  1096   1459#
R16X    138#   549    560
R1LOC   915#   933   1980
R1X     139#
R2LOC   917#   934   1982
R64X    137#
RADCT   175#  1918
RDBC    242#   692
RDBCC   243#
RDRC    264#  1956   1969
RDSTS   245#   655    659    680    787
READ   1097   1518#
RED0   1522#  1525   1585
RED1   1550#  1557
RED2   1577#  1584
RED3   1531   1587#
RED4   1597   1603#
RESET   402#   532    534
RESTAR  531    533   2744#
RFR     169#
RI      975   1874   1900#  2889
```

```
RIØ     1912#  1915
RI1     1919#  1922
RI2     1926#  1932
RI3     1933#  1967
RI4     1929   1937#
RI4B    1936   194Ø#
RI5     19Ø4   195Ø#
RI6     1959#  1966
RI7     1963   1968#
RI8     1952   1977#
RIX     1523   239Ø   2397   2888#
RLLB     2Ø1#
RLLM     2Ø2#   455    542    553
RLMB     2ØØ#
RMSK     335#  1182   19Ø3
ROV      168#
RPAR     167#
RPPC     273#  2324
RPSTC    274#  2369
RPTR     346#  12Ø2   1951
RRDY     165#   732    739   183Ø   1853   1928   2194   2213
RRSTS    244#   685
RSTØ    2782#  2793
RST1    2815   2826   2836#
RST2    2861#  2879
RST3    2871   2876#
RSTA    2812   2816#
RSTB    2823   2827#
RSTC     267#  196Ø
RSTS     314#   771
RTCC     39Ø#   457
RTOCT    176#  1925
RTS      158#   177    178    551    562
RTTY     345#  12Ø1
RUSE1    347#  12Ø3   1979
RUSE2    348#  12Ø4
RXEN     155#   177    178    551    562
SBCH     156#
SICPØ    119#   446
SICP1    12Ø#   449
SINT     298#
SLOC     527    884#  1786
SOCPØ    121#  2829
SOCP1    122#   452    891   2759   2766
SRQ      291#
SRQACK   29Ø#
SRQDAK   289#
ST1      149#
ST15     148#
ST2      147#   549    56Ø
START   1Ø39#  1Ø46   1Ø5Ø   288Ø
STARTØ   973   1Ø36#
SUØ     1627#  1642
SU1     1634   164Ø#
SUBS    1Ø98   1624#
SYNC     14Ø#
SYND     17Ø#
```

```
SYSTAT   287#
TADV     177#  1916
TI      1044   1115   1140   1145   1161   1440   1460   1794   1797   1889   2623   2701   2902#
TLOC     904#  1366   2805   2859
TOS      518    871#   872   1019
TOUT     327#  1958
TRAM     297#
TRDY     164#  2031   2060
TRK0     316#   687    690    784    793    795    797    850
TRKL     392#   691
TRUE     321#  2234
TTYC     185#   561    563   1917   1924
TTYI     182#   734   1832   1910   1938
TTYIN   1828#  1831
TTYO     183#  2034
TTYOUT  2029#  2032   2115   2159
TTYS     184#   731   1829   1913   1927   2030   2193
TXBE     166#  1914
TXEN     153#   177    178    551    562
UC      2906   2921#
UCI      931#
UCO      932#
UCS      938#  2295
UI       986   2320#
UL1      937#
UO       987   2345#
UP1      935#
UP2      936#
UPPS     988   2367#
UR1      933#
UR2      934#
USCC     192#   550    552
USCI     189#   741   1855
USCO     191#  2063
USCS     190#   738   1852   2059   2212
USER     872#   883   2276
USRST    159#
VER       54#   855    855
VERH      55#   989
VERS     804    854#   857
WDBC     240#
WDBCC    241#
WPBC     238#   669
WPBCC    239#   673
WPPC     272#  2349
WR0     1656#  1698
WR1     1662#  1667
WR2     1665   1669#
WR3     1679#  1684
WRITE   1102   1651#
X       1103   1709#
X0      1714#  1721
X1      1716   1723#
X2      1725#  1752
X3      1739   1743#
X4      1731   1745#
X5      1712   1754#
```

```
X6      1756#  1766
XTBL     908#   931    932    933    934    935    936    937    938   2293
Z       1105   1793#

CROSS REFERENCE COMPLETE
```

```
 LOC  OBJ         SEQ         SOURCE STATEMENT

                    1 $       TITLE   ('INTELLEC SERIES II IPB DIAGNOSTIC')
                    2
                    3
                    4
                    5
                    6 ;       EQUATES
                    7
                    8
                    9 ;       CHECKSUM DEFINITIONS
                   10
 F800              11 MONORG  EQU     0F800H          ;MONITOR ROM ORIGIN
 0800              12 MONLEN  EQU     0800H           ;MONITOR ROM LENGTH
 001E              13 MONCHK  EQU     01EH            ;MONITOR ROM CHECKSUM
                   14
 E800              15 BOOORG  EQU     0E800H          ;BOOT ROM ORIGIN
 0800              16 BOOLEN  EQU     0800H           ;BOOT ROM LENGTH
 0055              17 BOOCHK  EQU     055H            ;BOOT ROM CHECKSUM
 00A3              18 BOOSUM  EQU     0A3H            ;BOOT ROM CHECKSUM BYTE CONTENTS (TO MAKE 55H)
                   19
                   20 ;       GENERAL DEFINITIONS
                   21
 000D              22 CR      EQU     0DH             ;CARRIAGE RETURN CHARACTER
 000A              23 LF      EQU     0AH             ;LINE FEED CHARACTER
 0038              24 INT7V   EQU     038H            ;INTERRUPT 7 VECTOR LOCATION
                   25
                   26 ;       PIC DEFINITIONS
                   27
 00FC              28 SPICMR  EQU     0FCH            ;SYSTEM PIC MASK REGISTER
 00FA              29 IPICMR  EQU     0FAH            ;IO PIC MASK REGISTER
 00FD              30 SPICCR  EQU     0FDH            ;SYSTEM PIC COMMAND REGISTER
 00FB              31 IPICCR  EQU     0FBH            ;IO PIC COMMAND REGISTER
 0020              32 EOI     EQU     20H             ;END OF INTERRUPT COMMAND
 000C              33 POLL    EQU     0CH             ;POLL COMMAND
                   34
                   35 ;       IOC AND PIO COMMAND DEFFINITIONS
                   36
 0008              37 CSMEM   EQU     01000B          ;CHECKSUM MEMORY COMMAND
 0007              38 DECHO   EQU     00111B          ;DATA ECHO COMMAND
 0006              39 SRQ     EQU     00110B          ;GENERATE INTERRUPT COMMAND
 0005              40 SRQACK  EQU     00101B          ;INTERRUPT ACKNOWLEDGE COMMAND
 0009              41 TRAM    EQU     01001B          ;TEST RAM COMMAND
                   42
                   43 ;       THINGS ALREADY DEFINED IN THE MONITOR
                   44
 0001              45 OBF     EQU     00000001B       ;SLAVE OUTPUT BUFFER IS FULL
 0002              46 IBF     EQU     00000010B       ;SLAVE INPUT BUFFER IS FULL
 0004              47 F0      EQU     00000100B       ;FLAG 0 - SAVE BUSY, MASTER LOCKED OUT
                   48
 00C0              49 IOCI    EQU     0C0H            ;IOC INPUT DATA (FROM DBB) PORT
 00C0              50 IOCO    EQU     0C0H            ;IOC OUTPUT DATA (TO DBB) PORT
 00C1              51 IOCS    EQU     0C1H            ;IOC STATUS PORT
 00C1              52 IOCC    EQU     0C1H            ;IOC COMMAND PORT
```

```
  LOC  OBJ          SEQ         SOURCE STATEMENT

                      53
  ØØF8                54 PIOI     EQU     ØF8H            ;PIO INPUT DATA (FROM DBB) PORT
  ØØF8                55 PIOO     EQU     ØF8H            ;PIO OUTPUT DATA (TO DBB) PORT
  ØØF9                56 PIOS     EQU     ØF9H            ;PIO STATUS PORT
  ØØF9                57 PIOC     EQU     ØF9H            ;PIO COMMAND PORT
                      58
  ØØ2Ø                59 INT5     EQU     ØØ1ØØØØØB       ;INTERRUPT LEVEL 5
  ØØ4Ø                6Ø INT6     EQU     Ø1ØØØØØØB       ;INTERRUPT LEVEL 6
  ØØ8Ø                61 INT7     EQU     1ØØØØØØØB       ;INTERRUPT LEVEL 7
                      62
  F8Ø9                63 CO       EQU     ØF8Ø9H          ;CO MONITOR FUNCTION
  F81B                64 MEMCHK   EQU     ØF81BH          ;MEMCHK MONITOR FUNCTION
                      65
                      66
                      67
                      68
                      69 ;        GLOBALS
                      7Ø
                      71
  ØØ1Ø                72 FFLAG    EQU     Ø1ØH            ;MAJOR TEST FAILURE FLAG
                      73                                  ;  Ø = NO FAILURES IN TEST
                      74                                  ;  ØFFH = TEST HAS FAILED
  ØØ11                75 TOFLAG   EQU     Ø11H            ;TIMEOUT FLAG
                      76                                  ;  Ø = NO TIMEOUT
                      77                                  ;  ØFFH = TIMEOUT HAS OCCURRED
                      78
                      79
                      8Ø
                      81
                      82 ;        ENTRY POINTS
                      83
                      84
  EBØØ                85          ORG     ØEBØØH          ;BEGINNING OF DIAGNOSTIC
  EBØØ C324EB         86          JMP     MIMODE          ;MONITOR'S ENTRY POINT
  EBØ3 C3Ø6EB         87          JMP     BIMODE          ;BOOT ENTRY POINT
                      88
                      89 $        EJECT
  LOC  OBJ         SEQ         SOURCE STATEMENT

                       ;;;     BIMODE - BOOT INVOKED MODE CONTROL ROUTINE


  EB06 CD71ED          BIMODE: CALL    INIT            ;SAVE ENVIRONMENT
  EB09 CD67EB                  CALL    CHKSUM          ;CHECKSUM ROMS

  EB0C 0E55                    MVI     C,55H           ;CHECK FOR IOC PRESENCE
  EB0E CDE6EB                  CALL    IOCDRA
  EB11 211100                  LXI     H,TOFLAG        ;TEST TIME OUT FLAG
  EB14 B6                      ORA     M               ;AND RESULT FLAG
  EB15 CC9FEB                  CZ      IOCTST          ;RUN THE IOC TEST IF IOC PRESENT
  EB18 3E00                    MVI     A,0             ;RESET TIMEOUT FLAG
  EB1A 321100                  STA     TOFLAG

  EB1D CD2DEC                  CALL    PIOTST          ;RUN THE PIO TEST

  EB20 CDA8ED                  CALL    RESTOR          ;RESTORE THE ENVIRONMENT
  EB23 C9                      RET                     ;RETURN TO THE BOOT




                       ;;;     MIMODE - MONITOR INVOKED MODE CONTROL ROUTINE


  EB24 CD71ED          MIMODE: CALL    INIT            ;SAVE ENVIRONMENT
  EB27 014CEE                  LXI     B,SIGNON        ;PRINT SIGN ON MESSAGE
  EB2A CD9EED                  CALL    PRINTL

                       ;       CHECKSUM ROMS

  EB2D 0180EE                  LXI     B,MIM1          ;'CHECKSUM TEST' MESSAGE
  EB30 CDBFED                  CALL    SETUP           ;PRINT MESSAGE AND INITIALIZE FFLAG
  EB33 CD67EB                  CALL    CHKSUM          ;CHECKSUM ROMS
  EB36 CD5AED                  CALL    FINISH          ;CHECK FFLAG

                       ;       TEST IOC

  EB39 018AEE                  LXI     B,MIM2          ;'IOC TEST' MESSAGE
  EB3C CDBFED                  CALL    SETUP           ;PRINT MESSAGE AND INITIALIZE FFLAG
  EB3F CD9FEB                  CALL    IOCTST          ;TEST IOC
  EB42 CD5AED                  CALL    FINISH          ;CHECK FFLAG

                       ;       TEST PIO

  EB45 018EEE                  LXI     B,MIM3          ;'PIO TEST' MESSAGE
  EB48 CDBFED                  CALL    SETUP           ;PRINT MESSAGE AND INITIALIZE FFLAG
  EB4B CD2DEC                  CALL    PIOTST          ;TEST PIO
  EB4E CD5AED                  CALL    FINISH          ;CHECK FFLAG

                       ;       TEST RAM

  EB51 0192EE                  LXI     B,MIM4          ;'RAM TEST' MESSAGE
  EB54 CDBFED                  CALL    SETUP           ;PRINT MESSAGE AND INITIALIZE FFLAG
  EB57 CDBBEC                  CALL    RAMTST          ;TEST RAM
  LOC  OBJ         SEQ        SOURCE STATEMENT

  EB5A CD5AED       145         CALL    FINISH          ;CHECK FFLAG
                    146
                    147 ;       RETURN TO MONITOR
                    148
  EB5D 0171EE       149         LXI     B,SGNOFF        ;SIGNOFF MESSAGE
  EB60 CD9EED       150         CALL    PRINTL
  EB63 CDA8ED       151         CALL    RESTOR          ;RESTORE ENVIRONMENT
  EB66 C9           152         RET                     ;RETURN TO MONITOR
                    153
                    154
                    155 $       TITLE   ('CHKSUM - CHECKSUM TEST')
                    156 $       EJECT
  LOC  OBJ          SEQ         SOURCE STATEMENT

                     157 ;;;     CHKSUM - CHECKSUM ROMS
                     158
                     159
  EB67 21ØØF8        16Ø CHKSUM: LXI     H,MONORG        ;SET UP TO CHECKSUM MONITOR
  EB6A 11ØØØ8        161         LXI     D,MONLEN
  EB6D 3E1E          162         MVI     A,MONCHK
  EB6F CD8AEB        163         CALL    SUM             ;CHECKSUM MONITOR
  EB72 Ø196EE        164         LXI     B,CHKM1         ;'MONITOR CHECKSUM' MESSAGE
  EB75 CDDBED        165         CALL    TEST
                     166
  EB78 21ØØE8        167         LXI     H,BOOORG        ;SET UP TO CHECKSUM BOOT
  EB7B 11ØØØ8        168         LXI     D,BOOLEN
  EB7E 3E55          169         MVI     A,BOOCHK
  EB8Ø CD8AEB        17Ø         CALL    SUM             ;CHECKSUM BOOT
  EB83 Ø1A7EE        171         LXI     B,CHKM2         ;'BOOT CHECKSUM' MESSAGE
  EB86 CDDBED        172         CALL    TEST
  EB89 C9            173         RET
                     174
                     175
                     176
                     177
                     178 ;;;     SUM - CHECKSUM MEMORY
                     179 ;
                     18Ø ;       PARAMETERS:
                     181 ;          HL = ORIGIN OF ROM TO BE CHECKSUMMED
                     182 ;          DE = LENGTH OF ROM
                     183 ;          A = EXPECTED CHECKSUM
                     184 ;
                     185 ;       RETURNS:
                     186 ;          A = SUCCESS FLAG
                     187 ;              Ø = CHECKSUM OK
                     188 ;              OFFH = CHECKSUM FAILED
                     189
                     19Ø
  EB8A 2F            191 SUM:    CMA                     ;TAKE TWO'S COMPLEMENT OF EXPECTED CHECKSUM, SO
  EB8B 3C            192         INR     A               ;WHEN ADDED TO CHECKSUM THE TOTAL WILL BE ZERO
                     193
  EB8C 47            194 SUM1:   MOV     B,A             ;SAVE SUM DURING TEST
  EB8D 7B            195         MOV     A,E             ;TEST FOR NONE LEFT
  EB8E B2            196         ORA     D
  EB8F CA99EB        197         JZ      SUM2            ;IF NONE LEFT
  EB92 78            198         MOV     A,B             ;PUT COUNT BACK
  EB93 86            199         ADD     M               ;ACCUMULATE SUM
  EB94 23            2ØØ         INX     H               ;STEP TO NEXT WORD
  EB95 1B            2Ø1         DCX     D               ;DECREMENT COUNT
  EB96 C38CEB        2Ø2         JMP     SUM1            ;LOOP
                     2Ø3
  EB99 78            2Ø4 SUM2:   MOV     A,B             ;GET SUM
  EB9A B7            2Ø5         ORA     A               ;TEST FOR ZERO SUM
  EB9B C8            2Ø6         RZ                      ;IF ZERO, RETURN SUCCESS
                     2Ø7
  EB9C 3EFF          2Ø8         MVI     A,ØFFH          ;RETURN FAILURE
  EB9E C9            2Ø9         RET
                     21Ø
                     211
  LOC  OBJ         SEQ         SOURCE STATEMENT

                    212 $       TITLE   ('IOCTST - IOC TEST')
                    213 $       EJECT
  LOC  OBJ         SEQ         SOURCE STATEMENT

                    214 ;;;     IOCTST - IOC TEST
                    215
                    216
                    217 IOCTST:
                    218
                    219 ;       ECHO TEST
                    22Ø
  EB9F ØE55         221         MVI     C,55H           ;TRY TO ECHO A 55H
  EBA1 CDE6EB       222         CALL    IOCDRA
  EBA4 2111ØØ       223         LXI     H,TOFLAG        ;TEST THE RESULT AND TIME-OUT FLAGS
  EBA7 B6           224         ORA     M
  EBA8 CAB2EB       225         JZ      IOC1            ;JUMP IF OK
  EBAB Ø1B5EE       226         LXI     B,IOCM1         ;FAILURE MESSAGE
  EBAE CDDBED       227         CALL    TEST
  EBB1 C9           228         RET                     ;DO NOT DO OTHER TESTS IF NOT PRESENT
                    229
                    23Ø ;       IOC CHECKSUM TEST
                    231
  EBB2 Ø6Ø8         232 IOC1:   MVI     B,CSMEM         ;CHECKSUM COMMAND
  EBB4 CDF7EB       233         CALL    IOCDRB
  EBB7 Ø1D2EE       234         LXI     B,IOCM2
  EBBA CDDBED       235         CALL    TEST
                    236
                    237 ;       IOC RAM TEST
                    238
  EBBD Ø6Ø9         239         MVI     B,TRAM          ;TEST RAM COMMAND
  EBBF CDF7EB       24Ø         CALL    IOCDRB
  EBC2 Ø1DFEE       241         LXI     B,IOCM3
  EBC5 CDDBED       242         CALL    TEST
                    243
                    244 ;       IOC INTERRUPT TEST
                    245
  EBC8 ØEBF         246         MVI     C,NOT INT6      ;SET UP MASKS
  EBCA CD4ØEE       247         CALL    SETINT
  EBCD Ø6Ø6         248         MVI     B,SRQ           ;TURN ON IOC INTERRUPT
  EBCF CDFEEB       249         CALL    IOCDRC
  EBD2 CD13EE       25Ø         CALL    CHKINT          ;CHECK INTERRUPTS
  EBD5 F5           251         PUSH    PSW             ;SAVE RESULT
  EBD6 Ø6Ø5         252         MVI     B,SRQACK        ;RESET IOC INTERRUPT
  EBD8 CDFEEB       253         CALL    IOCDRC
  EBDB CD2EEE       254         CALL    RESET           ;RESTORE INTERRRUPTS TO NORMAL
                    255
  EBDE F1           256         POP     PSW             ;GET RESULT FLAG
  EBDF Ø1E7EE       257         LXI     B,IOCM4         ;LOAD MESSAGE POINTER
  EBE2 CDDBED       258         CALL    TEST
  EBE5 C9           259         RET
                    26Ø
                    261
                    262
                    263
                    264 ;;;     IOCDRA - ECHO TEST DRIVER
                    265 ;
                    266 ;          IOCDRA RUNS AN ECHO TEST WITH THE VALUE SUPPLIED, AND RETURNS THE
                    267 ;       SUCCESS FLAG IN A.
                    268 ;
  LOC  OBJ         SEQ         SOURCE STATEMENT

                    269 ;       PARAMETER:
                    27Ø ;          C = DATA TO BE ECHOED
                    271 ;
                    272 ;       RETURNS:
                    273 ;          A = SUCCESS FLAG
                    274 ;              Ø = PASSED
                    275 ;              ØFFH - FAILED
                    276 ;
                    277 ;       CALLS:
                    278 ;          IOCCOD
                    279 ;          IOCDID
                    28Ø ;          IOCDOD
                    281
                    282
  EBE6 Ø6Ø7         283 IOCDRA: MVI     B,DECHO          ;PUT OUT DATA ECHO COMMAND
  EBE8 CDFEEB       284         CALL    IOCCOD
  EBEB CDØFEC       285         CALL    IOCDOD
  EBEE CDØ7EC       286         CALL    IOCDID           ;READ BACK DATA
  EBF1 2F           287         CMA                      ;RETURNS COMPLEMENT
  EBF2 91           288         SUB     C                ;CHECK IF ECHO EQUALS ORIGINAL
  EBF3 C8           289         RZ                       ;RETURN IF OK
  EBF4 3EFF         29Ø         MVI     A,ØFFH           ;OTHERWISE,RETURN FAILURE
  EBF6 C9           291         RET
                    292
                    293
                    294
                    295
                    296 ;;;     IOCDRB - ISSUE COMMAND AND READ DATA
                    297 ;
                    298 ;       PARAMETER:
                    299 ;          B = COMMAND
                    3ØØ ;
                    3Ø1 ;       RETURNS:
                    3Ø2 ;          A = DATA
                    3Ø3 ;
                    3Ø4 ;       CALLS:
                    3Ø5 ;          IOCCOD
                    3Ø6 ;          IOCDID
                    3Ø7
                    3Ø8
  EBF7 CDFEEB       3Ø9 IOCDRB: CALL    IOCCOD           ;PUT OUT COMMAND
  EBFA CDØ7EC       31Ø         CALL    IOCDID           ;READ DATA
  EBFD C9           311         RET
                    312
                    313
                    314
                    315
                    316 ;;;     IOCDRC - ISSUE COMMAND
                    317 ;
                    318 ;       PARAMETER:
                    319 ;          B = COMMAND
                    32Ø ;
                    321 ;       CALLS:
                    322 ;          IOCCOD
                    323
  LOC  OBJ         SEQ         SOURCE STATEMENT

                    324 ;IOCDRC EQU     IOCCOD          ;THIS IS JUST THE COMMAND OUT DRIVER
                    325                                 ;NOTE: ACTUAL DEFINITION FOLLOWS IOCCOD
                    326
                    327
                    328
                    329
                    33Ø ;;      IOCCOD - IOC COMMAND OUT DRIVER
                    331 ;
                    332 ;       PARAMETER:
                    333 ;          B = COMMAND
                    334 ;
                    335 ;       MODIFIES A,B,E,HL
                    336
  EBFE 1EØØ         337 IOCCOD: MVI     E,Ø             ;TEST FOR ZERO STATUS
  ECØØ CD18EC       338         CALL    IOCWT           ;WAIT FOR STATUS OR TIMEOUT
  ECØ3 78           339         MOV     A,B             ;OUTPUT COMMAND
  ECØ4 D3C1         34Ø         OUT     IOCC
  ECØ6 C9           341         RET
                    342
                    343
  EBFE              344 IOCDRC  EQU     IOCCOD          ;DEFINITION HERE DUE TO FORWARD REFERENCE
                    345
                    346
                    347
                    348
                    349 ;;      IOCDID - IOC DATA IN DRIVER
                    35Ø ;
                    351 ;       RETURNS:
                    352 ;          A = DATA
                    353 ;
                    354 ;       MODIFIES A,E,HL
                    355
  ECØ7 1EØ1         356 IOCDID: MVI     E,OBF           ;TEST FOR OBF STATUS
  ECØ9 CD18EC       357         CALL    IOCWT
  ECØC DBCØ         358         IN      IOCI            ;READ DATA
  ECØE C9           359         RET
                    36Ø
                    361
                    362
                    363
                    364 ;;      IOCDOD - IOC DATA OUT DRIVER
                    365 ;
                    366 ;       PARAMETER:
                    367 ;          C = DATA
                    368 ;
                    369 ;       MODIFIES A,E,HL
                    37Ø
  ECØF 1EØØ         371 IOCDOD: MVI     E,Ø             ;TEST FOR ZERO STATUS
  EC11 CD18EC       372         CALL    IOCWT           ;WAIT FOR READY STATUS
  EC14 79           373         MOV     A,C             ;WRITE DATA
  EC15 D3CØ         374         OUT     IOCO
  EC17 C9           375         RET
                    376
                    377
                    378
  LOC  OBJ          SEQ         SOURCE STATEMENT

                     379
                     380 ;;;     IOCWT - IOC WAIT
                     381 ;          IOCWT WILL WAIT FOR THE IOC STATUS TO BE EQUAL TO REG. E,
                     382 ;       OR TO TIMEOUT
                     383 ;
                     384 ;       PARAMETERS:
                     385 ;          E = STATUS DESIRED
                     386 ;
                     387 ;       RETURNS:
                     388 ;          TOFLAG UPDATED TO 0FFH IF TIMEOUT HAS OCCURED
                     389
                     390
  EC18 210010        391 IOCWT:  LXI     H,01000H                ;WAIT COUNT
  EC1B DBC1          392 IOCWT1: IN      IOCS            ;CHECK STATUS
  EC1D E607          393         ANI     F0 OR IBF OR OBF
  EC1F AB            394         XRA     E               ;CHECK FOR DESIRED STATUS
  EC20 C8            395         RZ                      ;RETURN IF OK
  EC21 2B            396         DCX     H               ;DECREMENT TIMEOUT
  EC22 7D            397         MOV     A,L             ;TEST FOR TIMED OUT
  EC23 B4            398         ORA     H
  EC24 C21BEC        399         JNZ     IOCWT1          ;IF NOT TIMED OUT
  EC27 3EFF          400         MVI     A,0FFH          ;UPDATE TOFLAG TO FAILURE STATUS
  EC29 321100        401         STA     TOFLAG
  EC2C C9            402         RET                     ;RETURN
                     403
                     404
                     405
                     406
                     407 $       TITLE   ('PIOTST - PIO TEST')
                     408 $       EJECT
  LOC  OBJ          SEQ          SOURCE STATEMENT

                     4Ø9 ;;;      PIOTST - PIO TEST
                     41Ø
                     411
                     412 PIOTST:
                     413
                     414 ;        ECHO TEST
                     415
  EC2D ØE55          416          MVI     C,55H           ;TRY TO ECHO A 55H
  EC2F CD74EC        417          CALL    PIODRA
  EC32 2111ØØ        418          LXI     H,TOFLAG        ;TEST TIME-OUT AND RESULT FLAGS
  EC35 B6            419          ORA     M
  EC36 CA4ØEC        42Ø          JZ      PIO1            ;IF RESULT OK
  EC39 Ø1F6EE        421          LXI     B,PIOM1         ;FAILURE MESSAGE
  EC3C CDDBED        422          CALL    TEST
  EC3F C9            423          RET                     ;DO NOT DO OTHER TESTS IF NOT PRESENT
                     424
                     425 ;        PIO CHECKSUM TEST
                     426
  EC4Ø Ø6Ø8          427 PIO1:    MVI     B,CSMEM         ;CHECKSUM COMMAND
  EC42 CD85EC        428          CALL    PIODRB
  EC45 Ø1Ø9EF        429          LXI     B,PIOM2
  EC48 CDDBED        43Ø          CALL    TEST
                     431
                     432 ;        PIO RAM TEST
                     433
  EC4B Ø6Ø9          434          MVI     B,TRAM          ;TEST RAM COMMAND
  EC4D CD85EC        435          CALL    PIODRB
  EC5Ø Ø116EF        436          LXI     B,PIOM3
  EC53 CDDBED        437          CALL    TEST
                     438
                     439 ;        PIO INTERRUPT TEST
                     44Ø
  EC56 ØEDF          441          MVI     C,NOT INT5      ;SET UP MASKS
  EC58 CD4ØEE        442          CALL    SETINT
  EC5B Ø6Ø6          443          MVI     B,SRQ           ;TURN ON PIO INTERRUPT
  EC5D CD8CEC        444          CALL    PIODRC
  EC6Ø CD13EE        445          CALL    CHKINT          ;CHECK INTERRUPTS
  EC63 F5            446          PUSH    PSW             ;SAVE RESULT FLAG
  EC64 Ø6Ø5          447          MVI     B,SRQACK        ;RESET PIO INTERRUPT
  EC66 CD8CEC        448          CALL    PIODRC
  EC69 CD2EEE        449          CALL    RESET           ;RESTORE INTERRRUPTS TO NORMAL
                     45Ø
  EC6C F1            451          POP     PSW             ;GET RESULT FLAG
  EC6D Ø11EEF        452          LXI     B,PIOM4         ;LOAD MESSAGE POINTER
  EC7Ø CDDBED        453          CALL    TEST
                     454
  EC73 C9            455          RET
                     456
                     457
                     458
                     459
                     46Ø ;;;      PIODRA - ECHO TEST DRIVER
                     461 ;
                     462 ;           PIODRA RUNS AN ECHO TEST WITH THE VALUE SUPPLIED, AND RETURNS THE
                     463 ;        SUCCESS FLAG IN A.
  LOC  OBJ         SEQ         SOURCE STATEMENT

                    464 ;
                    465 ;       PARAMETER:
                    466 ;          C = DATA TO BE ECHOED
                    467 ;
                    468 ;       RETURNS:
                    469 ;          A = SUCCESS FLAG
                    470 ;              Ø = PASSED
                    471 ;              ØFFH - FAILED
                    472 ;
                    473 ;       CALLS:
                    474 ;          PIOCOD
                    475 ;          PIODID
                    476 ;          PIODOD
                    477
                    478
  EC74 Ø6Ø7         479 PIODRA: MVI     B,DECHO         ;PUT OUT DATA ECHO COMMAND
  EC76 CD8CEC       48Ø         CALL    PIOCOD
  EC79 CD9DEC       481         CALL    PIODOD
  EC7C CD95EC       482         CALL    PIODID          ;READ BACK DATA
  EC7F 2F           483         CMA                     ;PIO RETURNS COMPLEMENTED DATA
  EC8Ø 91           484         SUB     C               ;CHECK IF ECHO EQUALS ORIGINAL
  EC81 C8           485         RZ                      ;RETURN IF OK
  EC82 3EFF         486         MVI     A,ØFFH          ;OTHERWISE,RETURN FAILURE
  EC84 C9           487         RET
                    488
                    489
                    49Ø
                    491
                    492 ;;;     PIODRB - ISSUE COMMAND AND READ DATA
                    493 ;
                    494 ;       PARAMETER:
                    495 ;          B = COMMAND
                    496 ;
                    497 ;       RETURNS:
                    498 ;          A = DATA
                    499 ;
                    5ØØ ;       CALLS:
                    5Ø1 ;          PIOCOD
                    5Ø2 ;          PIODID
                    5Ø3
                    5Ø4
  EC85 CD8CEC       5Ø5 PIODRB: CALL    PIOCOD          ;PUT OUT COMMAND
  EC88 CD95EC       5Ø6         CALL    PIODID          ;READ DATA
  EC8B C9           5Ø7         RET
                    5Ø8
                    5Ø9
                    51Ø
                    511
                    512 ;;;     PIODRC - ISSUE COMMAND
                    513 ;
                    514 ;       PARAMETER:
                    515 ;          B = COMMAND
                    516 ;
                    517 ;       CALLS:
                    518 ;          PIOCOD
  LOC  OBJ         SEQ         SOURCE STATEMENT

                       ;PIODRC EQU     PIOCOD          ;THIS IS JUST THE COMMAND OUT DRIVER
                                                       ;ACTUAL DEFINITION FOLLOWS PIOCOD




                       ;;      PIOCOD - PIO COMMAND OUT DRIVER
                       ;
                       ;       PARAMETER:
                       ;          B = COMMAND
                       ;
                       ;       MODIFIES A,B,E,HL

  EC8C 1E00            PIOCOD: MVI     E,0             ;WAIT FOR 0 STATUS
  EC8E CDA6EC                  CALL    PIOWT           ;CALL WAIT ROUTINE
  EC91 78                      MOV     A,B             ;OUTPUT COMMAND
  EC92 D3F9                    OUT     PIOC
  EC94 C9                      RET


  EC8C                 PIODRC  EQU     PIOCOD          ;DEFINITION HERE DUE TO FORWARD REFERENCE




                       ;;      PIODID - PIO DATA IN DRIVER
                       ;
                       ;       RETURNS:
                       ;          A = DATA
                       ;
                       ;       MODIFIES A,E,HL

  EC95 1E01            PIODID: MVI     E,OBF           ;WAIT FOR OBF STATUS
  EC97 CDA6EC                  CALL    PIOWT           ;WAIT
  EC9A DBF8                    IN      PIOI            ;READ DATA
  EC9C C9                      RET




                       ;;      PIODOD - PIO DATA OUT DRIVER
                       ;
                       ;       PARAMETER:
                       ;          C = DATA
                       ;
                       ;       MODIFIES A,E,HL

  EC9D 1E00            PIODOD: MVI     E,0             ;WAIT FOR 0 STATUS
  EC9F CDA6EC                  CALL    PIOWT
  ECA2 79                      MOV     A,C             ;WRITE DATA
  ECA3 D3F8                    OUT     PIOO
  ECA5 C9                      RET
  LOC  OBJ         SEQ         SOURCE STATEMENT

                    574
                    575
                    576 ;;;     PIOWT - PIO WAIT
                    577 ;          PIOWT WAITS FOR THE PIO STATUS TO BE EQUAL TO E, OR A TIMEOUT.
                    578 ;
                    579 ;       PARAMETER:
                    58Ø ;          E = STATUS TO WAIT FOR
                    581 ;
                    582 ;       RETURNS:
                    583 ;          TOFLAG UPDATED TO ØFFH IF A TIMEOUT OCCURS
                    584 ;
                    585 ;       MODIFIES:
                    586 ;          TOFLAG,HL
                    587
                    588
  ECA6 21ØØ1Ø       589 PIOWT:  LXI     H,Ø1ØØØH        ;WAIT COUNT
  ECA9 DBF9         59Ø PIOWT1: IN      PIOS            ;CHECK STATUS
  ECAB E6Ø7         591         ANI     FØ OR IBF OR OBF
  ECAD AB           592         XRA     E               ;CHECK IF EQUAL TO DESIRED
  ECAE C8           593         RZ                      ;IF OK
  ECAF 2B           594         DCX     H               ;DECREMENT TIMER
  ECBØ 7D           595         MOV     A,L             ;TEST FOR TIMED OUT
  ECB1 B4           596         ORA     H
  ECB2 C2A9EC       597         JNZ     PIOWT1          ;IF NOT TIMED OUT
  ECB5 3EFF         598         MVI     A,ØFFH          ;TIMED OUT; UPDATE TOFLAG
  ECB7 3211ØØ       599         STA     TOFLAG
  ECBA C9           6ØØ         RET
                    6Ø1
                    6Ø2
                    6Ø3
                    6Ø4
                    6Ø5 $       TITLE   ('RAMTST - RAM TEST')
                    6Ø6 $       EJECT
  LOC  OBJ         SEQ        SOURCE STATEMENT

                    6Ø7 ;;;     RAMTST - TAM TEST
                    6Ø8
                    6Ø9
  ECBB 2112ØØ       61Ø RAMTST: LXI     H,Ø12H          ;FIRST WORD TO FILL
  ECBE 11FFE7       611         LXI     D,ØE7FFH        ;BOTTOM OF BOOT ROM
  ECC1 CD2DED       612         CALL    FILL
  ECC4 21ØØFØ       613         LXI     H,ØFØØØH        ;TOP OF BOOT/DIAGNOSTIC ROM
  ECC7 11FFF7       614         LXI     D,ØF7FFH        ;BOTTOM OF MONITOR ROM
  ECCA CD2DED       615         CALL    FILL
                    616
                    617 ;       TEST BANK Ø-32K
                    618
  ECCD 2112ØØ       619         LXI     H,Ø12H          ;FIRST WORD TO TEST
  ECDØ 11FF7F       62Ø         LXI     D,Ø7FFFH        ;LAST WORD TO TEST
  ECD3 CDØ8ED       621         CALL    CHECK
  ECD6 Ø12DEF       622         LXI     B,RAMM1         ;'BANK Ø-23K FAILURE' MESSAGE
  ECD9 CDDBED       623         CALL    TEST
                    624
                    625 ;       TEST BANK 32-48K
                    626
  ECDC 21ØØ8Ø       627         LXI     H,Ø8ØØØH        ;FIRST WORD TO TEST
  ECDF 11FFBF       628         LXI     D,ØBFFFH        ;LAST WORD TO TEST
  ECE2 CDØ8ED       629         CALL    CHECK
  ECE5 Ø13CEF       63Ø         LXI     B,RAMM2         ;'BANK 32-48K FAILURE' MESSAGE
  ECE8 CDDBED       631         CALL    TEST
                    632
                    633 ;       TEST BANK 48-62K
                    634
  ECEB 21ØØCØ       635         LXI     H,ØCØØØH        ;FIRST WORD TO TEST
  ECEE 11FFE7       636         LXI     D,ØE7FFH        ;BOTTOM OF BOOT/DIAGNOSTIC ROM
  ECF1 CDØ8ED       637         CALL    CHECK
  ECF4 B7           638         ORA     A               ;TEST FOR FAILURE
  ECF5 C2Ø1ED       639         JNZ     RAM1            ;IF A FAILURE
  ECF8 21ØØFØ       64Ø         LXI     H,ØFØØØH        ;TOP OF BOOT/DIAGNOSTIC ROM
  ECFB 11FFF7       641         LXI     D,ØF7FFH        ;BOTTOM OF MONITOR ROM
  ECFE CDØ8ED       642         CALL    CHECK
  EDØ1 Ø14CEF       643 RAM1:   LXI     B,RAMM3         ;'BANK 48-62K FAILURE' MESSAGE
  EDØ4 CDDBED       644         CALL    TEST
                    645
  EDØ7 C9           646         RET
                    647
                    648
                    649
                    65Ø
                    651 ;;;     CHECK - CHECK SECTION OF MEMORY
                    652 ;
                    653 ;       PARAMETERS:
                    654 ;          DC = FIRST WORD ADDRESS OF BLOCK TO TEST
                    655 ;          HL = LAST WORD ADDRESS OF BLOCK TO TEST
                    656 ;
                    657 ;       RETURNS:
                    658 ;          A = Ø IF TEST SUCCESSFUL
                    659 ;          A = ØFFH AT FIRST FAILURE
                    66Ø ;
                    661 ;       CALLS:
```

```
  LOC  OBJ          SEQ         SOURCE STATEMENT

                     662 ;          SETLIM
                     663
                     664
  EDØ8 CD3BED        665 CHECK:  CALL    SETLIM          ;SET UP LIMITS TO TAKE MEMCHK INTO ACCOUNT
                     666
  EDØB 7A            667 CHECK1: MOV     A,D             ;CHECK IF ALREADY DONE
  EDØC B3            668         ORA     E
  EDØD CA27ED        669         JZ      CHECK2          ;IF ALREADY DONE
                     67Ø
  ED1Ø 7C            671         MOV     A,H             ;GENERATE PATTERN
  ED11 AD            672         XRA     L
  ED12 BE            673         CMP     M               ;CHECK IF PATTERN STILL IN MEMORY
  ED13 C22AED        674         JNZ     CHECK3
                     675
  ED16 2F            676         CMA                     ;STORE AND VERIFY COMPLEMENT
  ED17 77            677         MOV     M,A
  ED18 BE            678         CMP     M               ;VERIFY ØFFH IS THERE
  ED19 C22AED        679         JNZ     CHECK3
                     68Ø
  ED1C 2F            681         CMA                     ;PUT PATTERN BACK
  ED1D 77            682         MOV     M,A
  ED1E BE            683         CMP     M               ;VERIFY PATTERN
  ED1F C22AED        684         JNZ     CHECK3
                     685
  ED22 23            686         INX     H               ;ADVANCE ADDRESS
  ED23 1B            687         DCX     D               ;DECREMENT COUNT
  ED24 C3ØBED        688         JMP     CHECK1          ;LOOP
                     689
  ED27 3EØØ          69Ø CHECK2: MVI     A,Ø             ;RETURN SUCCESS
  ED29 C9            691         RET
                     692
  ED2A 3EFF          693 CHECK3: MVI     A,ØFFH          ;RETURN FAILURE
  ED2C C9            694         RET
                     695
                     696
                     697
                     698
                     699 ;;;     FILL - FILL MEMORY WITH BACKGROUND, TAKING MEMCHK INTO ACCOUNT.
                     7ØØ ;
                     7Ø1 ;          FILL WILL PUT BACKGROUND INTO MEMORY STARTING AT FIRST WORD ADDRESS
                     7Ø2 ;       AND ENDING AT MEMCHK OR LAST MEMORY ADDRESS, WHICHEVER IS ENCOUNTERED
                     7Ø3 ;       FIRST
                     7Ø4 ;
                     7Ø5 ;       PARAMETERS:
                     7Ø6 ;          HL = FIRST WORD ADDRESS
                     7Ø7 ;          DE = LAST WORD ADDRESS
                     7Ø8 ;
                     7Ø9 ;       CALLS:
                     71Ø ;          SETLIM
                     711
                     712
  ED2D CD3BED        713 FILL:   CALL SETLIM
                     714
  ED3Ø 7A            715 FILL1:  MOV     A,D             ;CHECK IF COUNT=Ø
  ED31 B3            716         ORA     E
  LOC  OBJ          SEQ        SOURCE STATEMENT

  ED32 C8                        RZ
  ED33 7C                        MOV     A,H             ;GENERATE PATTERN
  ED34 AD                        XRA     L
  ED35 77                        MOV     M,A             ;STORE PATTERN
  ED36 23                        INX     H               ;INCREMENT POINTER TO NEXT LOCATION
  ED37 1B                        DCX     D               ;DECREMENT COUNTER
  ED38 C330ED                    JMP     FILL1           ;LOOP




                         ;;;     SETLIM - SET LIMITS
                         ;
                         ;          SETLIM PERFORMS THE PLM FUNCTION:
                         ;
                         ;       IF FWA <= MEMCHK AND MEMCHK <= LWA
                         ;       THEM COUNT = MEMCHK - FWA + 1;
                         ;       ELSE COUNT = LWA - FWA + 1;
                         ;
                         ;       PARAMETERS:
                         ;          HL = FIRST WORD ADDRESS (FWA)
                         ;          DE = LAST WORD ADDRESS (LWA)
                         ;
                         ;       RETURNS:
                         ;          HL = FIRST WORD ADDRESS
                         ;          DE = COUNT
                         ;
                         ;       CALLS:
                         ;          MEMCHK


  ED3B E5                SETLIM: PUSH    H               ;SAVE HL AND DE DURING CALL TO MEMCHK
  ED3C D5                        PUSH    D

  ED3D CD1BF8                    CALL    MEMCHK
  ED40 4F                        MOV     C,A             ;BC=MEMCHK

  ED41 D1                        POP     D               ;RESTORE HL AND DE
  ED42 E1                        POP     H

  ED43 95                        SUB     L               ;SUBTRACT FWA FROM MEMCHK
  ED44 78                        MOV     A,B
  ED45 9C                        SBB     H
  ED46 DA52ED                    JC      SETLM1          ;JUMP IF MEMCHK < FWA

  ED49 7B                        MOV     A,E             ;SUBTRACT MEMCHK FROM LWA
  ED4A 91                        SUB     C
  ED4B 7A                        MOV     A,D
  ED4C 98                        SBB     B
  ED4D DA52ED                    JC      SETLM1          ;JUMP IF LWA < MEMCHK

  ED50 50                        MOV     D,B             ;MEMCHK IS WITHIN RANGE; USE IT AS LWA
  ED51 59                        MOV     E,C

  ED52 7B                SETLM1: MOV     A,E             ;SUBTRACT FWA FROM MEMCHK OR LWA,
  LOC  OBJ          SEQ         SOURCE STATEMENT

  ED53 95            772          SUB     L                ;  AS THE CASE MAY BE
  ED54 5F            773          MOV     E,A
  ED55 7A            774          MOV     A,D
  ED56 9C            775          SBB     H
  ED57 57            776          MOV     D,A
                     777
  ED58 13            778          INX     D                ;ADD 1
                     779
  ED59 C9            780          RET
                     781
                     782
                     783 $        TITLE   ('UTILITY ROUTINES')
                     784 $        EJECT
```

```
 LOC  OBJ         SEQ         SOURCE STATEMENT

                   785 ;;;     FINISH - PRINT ' -- PASSED' IF FFLAG Ø
                   786 ;
                   787 ;       ACCESSES:
                   788 ;          FFLAG
                   789 ;
                   79Ø ;       CALLS:
                   791 ;          PRINT
                   792
                   793
 ED5A 3A1ØØØ       794 FINISH: LDA     FFLAG           ;TEST FFLAG
 ED5D B7           795         ORA     A
 ED5E CØ           796         RNZ                     ;RETURN IF TEST FAILED
 ED5F Ø166ED       797         LXI     B,FINA          ;PRINT ' -- PASSED' MESSAGE
 ED62 CD9EED       798         CALL    PRINTL
 ED65 C9           799         RET
                   8ØØ
                   8Ø1
 ED66 2Ø2D2D2Ø     8Ø2 FINA:   DB      ' -- PASSED',Ø
 ED6A 5Ø415353
 ED6E 4544
 ED7Ø ØØ
                   8Ø3
                   8Ø4
                   8Ø5
                   8Ø6
                   8Ø7 ;;;     INIT - SAVE INVIRONMENT ON STACK
                   8Ø8 ;          INIT SAVES THE INTERRUPT MASKS OF BOTH 8257'S AND THE
                   8Ø9 ;       CONTENTS OF FFLAG.  IT IS INTENDED TO BE USED WITH RESTOR,
                   81Ø ;       AND MUST BE CALLED AT THE SAME NEST LEVEL AS RESTOR
                   811
                   812
 ED71 D1           813 INIT:   POP     D               ;SAVE RETURN SINCE STACK TO BE MODIFIED
 ED72 F5           814         PUSH    PSW             ;SAVE A AND FLAGS
 ED73 2A38ØØ       815         LHLD    INT7V           ;SAVE INTERRUPT 7 VECTOR
 ED76 E5           816         PUSH    H
 ED77 2A3AØØ       817         LHLD    INT7V+2         ;SAVE REST OF VECTOR
 ED7A E5           818         PUSH    H
 ED7B 2A1ØØØ       819         LHLD    Ø1ØH            ;SAVE TOFLAG AND FFLAG
 ED7E E5           82Ø         PUSH    H
 ED7F DBFA         821         IN      IPICMR          ;READ IO PIO MASK REGISTER
 ED81 47           822         MOV     B,A
 ED82 DBFC         823         IN      SPICMR          ;READ SYSTEM PIO MASK REGISTER
 ED84 4F           824         MOV     C,A
 ED85 C5           825         PUSH    B               ;SAVE THE MASKS IN THE STACK
 ED86 3EØØ         826         MVI     A,Ø             ;INITIALIZE FFLAG AND TOFLAG
 ED88 321ØØØ       827         STA     FFLAG
 ED8B 3211ØØ       828         STA     TOFLAG
 ED8E D5           829         PUSH    D               ;RETURN
 ED8F C9           83Ø         RET
                   831
                   832
                   833
                   834
                   835 ;;;     PRINT - PRINT STRING
                   836 ;
```

```
  LOC  OBJ          SOURCE STATEMENT

                    ;       PARAMETER:
                    ;          BC = POINTER TO STRING TERMINATED WITH A NULL.
                    ;
                    ;       CALLS:
                    ;          CO


  ED9Ø C5           PRINT:  PUSH    B               ;SAVE POINTER ON THE STACK

  ED91 E1           PRINT1: POP     H               ;LOAD POINTER INTO HL
  ED92 4E                   MOV     C,M             ;READ NEXT CHARACTER
  ED93 79                   MOV     A,C             ;PREPARE FOR TERMINATOR CHECK
  ED94 B7                   ORA     A               ;CHECK FOR STRING TERMINATOR
  ED95 C8                   RZ                      ;RETURN IF NULL
  ED96 23                   INX     H               ;INCREMENT POINTER
  ED97 E5                   PUSH    H               ;RESTORE ON STACK
  ED98 CDØ9F8               CALL    CO              ;PRINT CHARACTER
  ED9B C391ED               JMP     PRINT1          ;LOOP UNTIL DONE




                    ;;;     PRINTL - PRINT MESSAGE WITH A CR-LF ADDED AT THE END
                    ;
                    ;       PARAMETER:
                    ;          BC = POINTER TO STRING TERMINATED WITH A NULL
                    ;
                    ;       CALLS:
                    ;          PRINT


  ED9E CD9ØED       PRINTL: CALL    PRINT           ;PRINT ORIGINAL STRING
  EDA1 Ø149EE               LXI     B,CRLF          ;PRINT CR-LF
  EDA4 CD9ØED               CALL    PRINT
  EDA7 C9                   RET




                    ;;;     RESTOR - RESTOR ENVIRONMENT
                    ;          RESTOR IS THE COMPLEMENT OF INIT


  EDA8 D1           RESTOR: POP     D               ;SAVE RETURN ADDR WHILE PLAYING WITH STACK
  EDA9 C1                   POP     B               ;READ INTERRUPT MASKS
  EDAA 79                   MOV     A,C             ;RESTORE SYSTEM INTERRUPT MASK
  EDAB D3FC                 OUT     SPICMR
  EDAD 78                   MOV     A,B             ;RESTORE IO INTERRUPT MASK
  EDAE D3FA                 OUT     IPICMR
  EDBØ E1                   POP     H               ;RESTORE TOFLAG AND FFLAG
  EDB1 221ØØØ               SHLD    Ø1ØH
  EDB4 E1                   POP     H               ;RESTORE INTERRUPT VECTOR
  EDB5 223AØØ               SHLD    INT7V+2
  EDB8 E1                   POP     H
  EDB9 2238ØØ               SHLD    INT7V
  LOC  OBJ         SEQ         SOURCE STATEMENT

  EDBC F1                     POP     PSW               ;RESTORE A AND FLAGS
  EDBD D5                     PUSH    D                 ;RETURN
  EDBE C9                     RET



                        ;;;   SETUP - SET UP FOR TEST
                        ;
                        ;        SETUP PRINTS OUT THE START MESSAGE FOR A TEST AND INITIALIZES
                        ;     FFLAG TO 0.
                        ;
                        ;     PARAMETER:
                        ;        BC = POINTER TO START MESSAGE
                        ;
                        ;     MODIFIES:
                        ;        FFLAG
                        ;
                        ;     CALLS:
                        ;        PRINT


  EDBF C5               SETUP: PUSH   B                 ;SAVE MESSAGE POINTER
  EDC0 01D0ED                 LXI     B,SETA            ;PRINT 'TESTING '
  EDC3 CD90ED                 CALL    PRINT
  EDC6 C1                     POP     B                 ;PRINT MESSAGE
  EDC7 CD90ED                 CALL    PRINT
  EDCA 3E00                   MVI     A,0               ;ZERO OUT FFLAG
  EDCC 321000                 STA     FFLAG
  EDCF C9                     RET


  EDD0 20205445         SETA:  DB      '  TESTING ',0
  EDD4 5354494E
  EDD8 4720
  EDDA 00




                        ;;;   TEST - TEST RESULT FLAG OF A TEST
                        ;
                        ;        THIS ROUTINE TESTS THE RESULT OF A TEST AND PRINTS A
                        ;     FAILURE MESSAGE IF JUSTIFIED.  THE FLAG 'FFLAG' ARE TESTED,
                        ;     AND IF THERE HAVE BEEN NO FAILURES IN THE TEST TO THAT POINT,
                        ;     A CR-LF PAIR IS OUTPUT.  FFLAG IS UPDATED TO TO REFLECT THE
                        ;     FAILURE.  TOFLAG IS RESET TO 0.
                        ;
                        ;     PARAMETERS:
                        ;        A = RESULT FLAG
                        ;            0 => TEST PASSED
                        ;            0FFH => TEST FAILED
                        ;        BC = MESSAGE ADDRESS
                        ;
                        ;     MODIFIES:
```

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V2.0         MODULE    PAGE    22
UTILITY ROUTINES

  LOC  OBJ          SEQ         SOURCE STATEMENT

                     944 ;          FFLAG
                     945 ;          TOFLAG
                     946 ;
                     947 ;       CALLS:
                     948 ;          PRINT
                     949
                     950
  EDDB 211100        951 TEST:   LXI     H,TOFLAG        ;CHECK TIMEOUT AND RESULT FLAGS
  EDDE B6            952         ORA     M               ;CHECK FLAGS
  EDDF C8            953         RZ                      ;RETURN IF PASSED
  EDE0 C5            954         PUSH    B               ;SAVE MESSAGE POINTER
  EDE1 3A1000        955         LDA     FFLAG           ;CHECK FAILURE FLAG
  EDE4 B7            956         ORA     A
  EDE5 C2F3ED        957         JNZ     TEST1           ;IF THERE HAS ALREADY BEEN A FAILURE
                     958
  EDE8 0149EE        959         LXI     B,CRLF          ;CR-LF MESSAGE
  EDEB CD90ED        960         CALL    PRINT
  EDEE 3EFF          961         MVI     A,0FFH          ;SET FFLAG TO FAILED
  EDF0 321000        962         STA     FFLAG
                     963
  EDF3 0103EE        964 TEST1:  LXI     B,TESTA         ;'FAILURE -- ' MESSAGE
  EDF6 CD90ED        965         CALL    PRINT
                     966
  EDF9 C1            967         POP     B               ;POP ERROR MESSAGE POINTER
  EDFA CD9EED        968         CALL    PRINTL          ;PRINT ERROR MESSAGE
                     969
  EDFD 3E00          970         MVI     A,0             ;RESET TOFLAG
  EDFF 321100        971         STA     TOFLAG
  EE02 C9            972         RET
                     973
                     974
  EE03 20202020      975 TESTA:  DB      '    FAILURE -- ',0
  EE07 4641494C
  EE0B 55524520
  EE0F 2D2D20
  EE12 00
                     976
                     977
                     978 $       TITLE   ('INTERRUPT UTILITIES')
                     979 $       EJECT
```

```
 LOC  OBJ         SEQ         SOURCE STATEMENT

                   980 ;;;     CHKINT - CHECK INTERRUPTS
                   981 ;
                   982 ;       ENTRY CONDITIONS:
                   983 ;          ALL UNDESIRED INTERRUPTS MASKED OUT
                   984 ;          DESIRED INTERRUPT LINE ON
                   985 ;          8080 INTERRUPTS DISABLED
                   986 ;
                   987 ;       RETURNS:
                   988 ;          A = SUCCCESS FLAG
                   989 ;              0 = TEST PASSED
                   990 ;              0FFH = TEST FAILED
                   991 ;          8080 INTERRUPTS TURNED OFF
                   992
                   993
 EE13 3EC3         994 CHKINT: MVI     A,0C3H          ;STORE A JUMP TO CHK2
 EE15 323800       995         STA     INT7V           ;  INTO INT7V
 EE18 2129EE       996         LXI     H,CHK2
 EE1B 223900       997         SHLD    INT7V+1
                   998
 EE1E FB           999         EI                      ;TURN THE INTERRUPTS ON
 EE1F 06FF        1000         MVI     B,255           ;WAIT A WHILE
 EE21 05          1001 CHK1:   DCR     B               ;DECREMENT THE COUNTER
 EE22 C221EE      1002         JNZ     CHK1            ;IF NOT COUNTED OUT YET
 EE25 F3          1003         DI                      ;TEST DONE; FAILED
 EE26 3EFF        1004         MVI     A,0FFH          ;RETURN FAILURE
 EE28 C9          1005         RET
                  1006
 EE29 F3          1007 CHK2:   DI                      ;TURN OFF INTERRUPTS
 EE2A E1          1008         POP     H               ;GET RID OF EXTRA RETURN ADDRESS
 EE2B 3E00        1009         MVI     A,0             ;RETURN SUCCESS
 EE2D C9          1010         RET
                  1011
                  1012
                  1013
                  1014
                  1015 ;;;     RESET - TRANSMIT EOI'S TO INTERRUPT CONTROLLERS
                  1016 ;
                  1017 ;       ENTRY CONDITIONS:
                  1018 ;          INTERRUPTS TURNED OFF
                  1019 ;
                  1020 ;       EXIT CONDITIONS:
                  1021 ;          INTERRUPTS TURNED ON
                  1022
                  1023
 EE2E 3E0C        1024 RESET:  MVI     A,POLL          ;SEND A POLL TO PICS
 EE30 D3FD        1025         OUT     SPICCR
 EE32 D3FB        1026         OUT     IPICCR
 EE34 DBFD        1027         IN      SPICCR          ;READ AND IGNORE POLL DATA
 EE36 DBFB        1028         IN      IPICCR
 EE38 3E20        1029         MVI     A,EOI
 EE3A D3FB        1030         OUT     IPICCR          ;OUTPUT EOI TO IO PIC COMMAND REGISTER
 EE3C D3FD        1031         OUT     SPICCR          ;OUTPUT EOI TO SYSTEM PIC COMMAND REGISTER
 EE3E FB          1032         EI
 EE3F C9          1033         RET
                  1034
```

```
  LOC  OBJ         SEQ         SOURCE STATEMENT

                   1Ø35
                   1Ø36
                   1Ø37
                   1Ø38 ;;;      SETINT - SET UP FOR INTERRUPT TEST
                   1Ø39 ;
                   1Ø4Ø ;        PARAMETER:
                   1Ø41 ;           C = MASK FOR IO PIC
                   1Ø42 ;
                   1Ø43 ;        EXIT CONDITIONS:
                   1Ø44 ;           PIC'S SET UP SO THIS IS THE ONLY INTERRUPT ENABLEABLE
                   1Ø45
                   1Ø46
  EE4Ø F3          1Ø47 SETINT: DI
  EE41 3E7F        1Ø48         MVI     A,NOT INT7       ;INITIALIZE SYSTEM PIC FOR ONLY IO INTERRUPTS
  EE43 D3FC        1Ø49         OUT     SPICMR
  EE45 79          1Ø5Ø         MOV     A,C              ;SET UP IO PIC
  EE46 D3FA        1Ø51         OUT     IPICMR
  EE48 C9          1Ø52         RET
                   1Ø53
                   1Ø54
                   1Ø55 $       TITLE   ('MESSAGES')
                   1Ø56 $       EJECT
  LOC  OBJ          SEQ         SOURCE STATEMENT

  EE49 0D           1057 CRLF:   DB      CR,LF,0
  EE4A 0A
  EE4B 00
  EE4C 0D           1058 SIGNON: DB      CR,LF,'INTELLEC SERIES II DIAGNOSTIC V1.0',0
  EE4D 0A
  EE4E 494E5445
  EE52 4C4C4543
  EE56 20534552
  EE5A 49455320
  EE5E 49492044
  EE62 4941474E
  EE66 4F535449
  EE6A 43205631
  EE6E 2E30
  EE70 00
  EE71 454E4420     1059 SGNOFF: DB      'END DIAGNOSTIC',0
  EE75 44494147
  EE79 4E4F5354
  EE7D 4943
  EE7F 00
                    1060
  EE80 43484543     1061 MIM1:   DB      'CHECKSUMS',0
  EE84 4B53554D
  EE88 53
  EE89 00
  EE8A 494F43       1062 MIM2:   DB      'IOC',0
  EE8D 00
  EE8E 50494F       1063 MIM3:   DB      'PIO',0
  EE91 00
  EE92 52414D       1064 MIM4:   DB      'RAM',0
  EE95 00
                    1065
  EE96 4D4F4E49     1066 CHKM1:  DB      'MONITOR CHECKSUM',0
  EE9A 544F5220
  EE9E 43484543
  EEA2 4B53554D
  EEA6 00
  EEA7 424F4F54     1067 CHKM2:  DB      'BOOT CHECKSUM',0
  EEAB 20434845
  EEAF 434B5355
  EEB3 4D
  EEB4 00
                    1068
  EEB5 494F4320     1069 IOCM1:  DB      'IOC NOT RESPONDING (N/A 210)',0
  EEB9 4E4F5420
  EEBD 52455350
  EEC1 4F4E4449
  EEC5 4E472028
  EEC9 4E2F4120
  EECD 32313029
  EED1 00
  EED2 494F4320     1070 IOCM2:  DB      'IOC CHECKSUM',0
  EED6 43484543
  EEDA 4B53554D
  EEDE 00
  LOC  OBJ          SEQ          SOURCE STATEMENT

  EEDF 494F4320     1071 IOCM3:  DB      'IOC RAM',0
  EEE3 52414D
  EEE6 00
  EEE7 494F4320     1072 IOCM4:  DB      'IOC INTERRUPTS',0
  EEEB 494E5445
  EEEF 52525550
  EEF3 5453
  EEF5 00
                    1073
  EEF6 50494F20     1074 PIOM1:  DB      'PIO NOT RESPONDING',0
  EEFA 4E4F5420
  EEFE 52455350
  EF02 4F4E4449
  EF06 4E47
  EF08 00
  EF09 50494F20     1075 PIOM2:  DB      'PIO CHECKSUM',0
  EF0D 43484543
  EF11 4B53554D
  EF15 00
  EF16 50494F20     1076 PIOM3:  DB      'PIO RAM',0
  EF1A 52414D
  EF1D 00
  EF1E 50494F20     1077 PIOM4:  DB      'PIO INTERRUPTS',0
  EF22 494E5445
  EF26 52525550
  EF2A 5453
  EF2C 00
                    1078
  EF2D 52414D20     1079 RAMM1:  DB      'RAM BANK 0-32K',0
  EF31 42414E4B
  EF35 20302D33
  EF39 324B
  EF3B 00
  EF3C 52414D20     1080 RAMM2:  DB      'RAM BANK 32-48K',0
  EF40 42414E4B
  EF44 2033322D
  EF48 34384B
  EF4B 00
  EF4C 52414D20     1081 RAMM3:  DB      'RAM BANK 48-62K',0
  EF50 42414E4B
  EF54 2034382D
  EF58 36324B
  EF5B 00
                    1082
  EF5C A3           1083         DB      BOOSUM          ;NUMBER TO MAKE CHECKSUM COME OUT TO 055H
                    1084
                    1085         END

PUBLIC SYMBOLS


EXTERNAL SYMBOLS


USER SYMBOLS
```

```
MESSAGES

BIMODE A EBØ6   BOOCHK A ØØ55   BOOLEN A Ø8ØØ   BOOORG A E8ØØ   BOOSUM A ØØA3   CHECK  A EDØ8   CHECK1 A EDØB
CHECK2 A ED27   CHECK3 A ED2A   CHK1   A EE21   CHK2   A EE29   CHKINT A EE13   CHKM1  A EE96   CHKM2  A EEA7
CHKSUM A EB67   CO     A F8Ø9   CR     A ØØØD   CRLF   A EE49   CSMEM  A ØØØ8   DECHO  A ØØØ7   EOI    A ØØ2Ø
FØ     A ØØØ4   FFLAG  A ØØ1Ø   FILL   A ED2D   FILL1  A ED3Ø   FINA   A ED66   FINISH A ED5A   IBF    A ØØØ2
INIT   A ED71   INT5   A ØØ2Ø   INT6   A ØØ4Ø   INT7   A ØØ8Ø   INT7V  A ØØ38   IOC1   A EBB2   IOCC   A ØØC1
IOCCOD A EBFE   IOCDID A ECØ7   IOCDOD A ECØF   IOCDRA A EBE6   IOCDRB A EBF7   IOCDRC A EBFE   IOCI   A ØØCØ
IOCM1  A EEB5   IOCM2  A EED2   IOCM3  A EEDF   IOCM4  A EEE7   IOCO   A ØØCØ   IOCS   A ØØC1   IOCTST A EB9F
IOCWT  A EC18   IOCWT1 A EC1B   IPICCR A ØØFB   IPICMR A ØØFA   LF     A ØØØA   MEMCHK A F81B   MIM1   A EE8Ø
MIM2   A EE8A   MIM3   A EE8E   MIM4   A EE92   MIMODE A EB24   MONCHK A ØØ1E   MONLEN A Ø8ØØ   MONORG A F8ØØ
OBF    A ØØØ1   PIO1   A EC4Ø   PIOC   A ØØF9   PIOCOD A EC8C   PIODID A EC95   PIODOD A EC9D   PIODRA A EC74
PIODRB A EC85   PIODRC A EC8C   PIOI   A ØØF8   PIOM1  A EEF6   PIOM2  A EFØ9   PIOM3  A EF16   PIOM4  A EF1E
PIOO   A ØØF8   PIOS   A ØØF9   PIOTST A EC2D   PIOWT  A ECA6   PIOWT1 A ECA9   POLL   A ØØØC   PRINT  A ED9Ø
PRINT1 A ED91   PRINTL A ED9E   RAM1   A EDØ1   RAMM1  A EF2D   RAMM2  A EF3C   RAMM3  A EF4C   RAMTST A ECBB
RESET  A EE2E   RESTOR A EDA8   SETA   A EDDØ   SETINT A EE4Ø   SETLIM A ED3B   SETLM1 A ED52   SETUP  A EDBF
SGNOFF A EE71   SIGNON A EE4C   SPICCR A ØØFD   SPICMR A ØØFC   SRQ    A ØØØ6   SRQACK A ØØØ5   SUM    A EB8A
SUM1   A EB8C   SUM2   A EB99   TEST   A EDDB   TEST1  A EDF3   TESTA  A EEØ3   TOFLAG A ØØ11   TRAM   A ØØØ9

ASSEMBLY COMPLETE,  NO ERRORS
```

intel®

INTEL CORPORATION, 3065 Bowers Avenue, Santa Clara, California 95051 (408) **987-8080**

Printed in U.S.A.