

**INTELLEC<sup>®</sup> SERIES II  
MICROCOMPUTER  
DEVELOPMENT SYSTEM  
BOOT/MONITOR LISTING**

Manual Order Number: 9800605-02 Rev. B

Additional copies of this manual or other Intel literature may be obtained from:

Literature Department  
Intel Corporation  
3065 Bowers Avenue  
Santa Clara, CA 95051

The information in this document is subject to change without notice.

Intel Corporation makes no warranty of any kind with regard to this material, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Intel Corporation assumes no responsibility for any errors that may appear in this document. Intel Corporation makes no commitment to update nor to keep current the information contained in this document.

Intel Corporation assumes no responsibility for the use of any circuitry other than circuitry embodied in an Intel product. No other circuit patent licenses are implied.

Intel software products are copyrighted by and shall remain the property of Intel Corporation. Use, duplication or disclosure is subject to restrictions stated in Intel's software license, or as defined in ASPR 7-104.9(a)(9).

No part of this document may be copied or reproduced in any form or by any means without the prior written consent of Intel Corporation.

The following are trademarks of Intel Corporation and may be used only to identify Intel products:

i	iSBC	Multimodule
ICE	Library Manager	PROMPT
iCS	MCS	Promware
Insite	Megachassis	RMX
Intel	Micromap	UPI
Intelelevision	Multibus	μScope
Inteltec		

and the combination of ICE, iCS, iSBC, MCS, or RMX and a numerical suffix.



## 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
1
2 ;*****
3 ;*
4 ;*                INTELLEC SERIES II BOOT/MONITOR          *
5 ;*                VERSION 1.3                               *
6 ;*
7 ;*    COPYRIGHT (C) 1978, 1979 INTEL CORPORATION.  ALL RIGHTS *
8 ;*    RESERVED.  NO PART OF THIS PROGRAM OR PUBLICATION    *
9 ;*    MAY BE REPRODUCED, TRANSMITTED, TRANSCRIBED,        *
10 ;*    STORED IN A RETRIEVAL SYSTEM, OR TRANSLATED INTO    *
11 ;*    ANY LANGUAGE OR COMPUTER LANGUAGE, IN ANY FORM      *
12 ;*    OR BY ANY MEANS, ELECTRONIC, MECHANICAL, MAGNETIC,  *
13 ;*    OPTICAL, CHEMICAL, MANUAL OR OTHERWISE, WITHOUT     *
14 ;*    THE PRIOR WRITTEN PERMISSION OF INTEL CORPORATION,  *
15 ;*    3065 BOWERS AVENUE, SANTA CLARA, CALIFORNIA 95051.  *
16 ;*
17 ;*****
18 ; <LEGAL COMMAND> ::= <ASSIGN I/O COMMAND>
19 ;                   <DISPLAY MEMORY COMMAND>
20 ;                   <ENDFILE COMMAND>
21 ;                   <FILL MEMORY COMMAND>
22 ;                   <PROGRAM EXECUTE COMMAND>
23 ;                   <HEXADECIMAL ARITHMETIC COMMAND>
24 ;                   <MOVE MEMORY COMMAND>
25 ;                   <LEADER COMMAND>
26 ;                   <QUERY STATUS COMMAND>
27 ;                   <READ HEXADECIMAL FILE COMMAND>
28 ;                   <SUBSTITUTE MEMORY COMMAND>
29 ;                   <WRITE HEXADECIMAL RECORD COMMAND>
30 ;                   <REGISTER MODIFY COMMAND>
31 ;                   <TRANSFER CONTROL TO DIAGNOSTIC PROGRAM COMMAND>
32 ; <ASSIGN I/O COMMAND> ::= A<LOGICAL DEVICE>=<PHYSICAL DEVICE>
33 ; <DISPLAY MEMORY COMMAND> ::= D<NUMBER>,<NUMBER>
34 ; <ENDFILE COMMAND> ::= E<NUMBER>
35 ; <FILL MEMORY COMMAND> ::= F<NUMBER>,<NUMBER>,<NUMBER>
36 ; <PROGRAM EXECUTE COMMAND> ::= G<NUMBER>,<NUMBER>,<NUMBER>
37 ; <HEXADECIMAL ARITHMETIC COMMAND> ::= H<NUMBER>,<NUMBER>
38 ; <MOVE MEMORY COMMAND> ::= M<NUMBER>,<NUMBER>,<NUMBER>
39 ; <LEADER COMMAND> ::= N
40 ; <QUERY STATUS COMMAND> ::= Q
41 ; <READ HEXADECIMAL FILE COMMAND> ::= R<NUMBER>
42 ; <SUBSTITUTE MEMORY COMMAND> ::= S<NUMBER><COMMA>...
43 ; <WRITE HEXADECIMAL RECORD COMMAND> ::= W<NUMBER>,<NUMBER>
44 ; <REGISTER MODIFY COMMAND> ::= X<REGISTER IDENTIFIER><NUMBER>...
45 ; <TRANSFER CONTROL TO DIAGNOSTIC PROGRAM COMMAND> ::= Z$
46 ; <LOGICAL DEVICE> ::= LOCAL CONSOLE!READER!LIST!PUNCH
47 ; <PHYSICAL DEVICE> ::= CRT!TTY!PTR!PTP!LPT!BATCH!112
48 ; <REGISTER IDENTIFIER> ::= A!B!C!D!E!F!H!I!L!M!P!S
49 ; <NUMBER> ::= <HEX DIGIT>
50 ;                   <NUMBER><HEX DIGIT>
51 ; <HEX DIGIT> ::= 0!1!2!3!4!5!6!7!8!9!A!B!C!D!E!F
52 ;*****
53 $ TITLE (' INTELLEC SERIES II MONITOR, VERSION 1.3, 1 MARCH 1979 ')
54 VER EQU 13 ; VERSION 1.3

```

```

LOC  OBJ      LINE      SOURCE STATEMENT
0013          55 VERH    EQU     13H      ; STORAGE REPRESENTATION OF VERSION
0103          56 DATE    EQU     0103H    ; CREATION DATE, 01 MARCH 1979
          57 ; NOTE:
          58 ; THE DATE SHOWN ABOVE IS ENCODED IN A TWO BYTE FIELD IN BOTH THE BOOTSTRAP
          59 ; PROM AND THE MONITOR PROM IN ORDER TO CONTROL NEW RELEASES OF THIS PROGRAM.
          60 ; THE DATE CODE IS LOCATED AT ADDRESSES 0E804H AND 0E805H IN THE BOOTSTRAP
          61 ; AND AT ADDRESSES 0F824H AND 0F825H IN THE MONITOR.
          62 ; THE VERSION CODE IS LOCATED IN THE MONITOR ROM AT ADDRESS 0F82FH.
          63 ; WHEN A NEW RELEASE IS ISSUED, PLEASE CHANGE THE DATE AND VERSION CODES.
          64 ; THE COPYRIGHT NOTICE IS LOCATED IN THE MONITOR ROM BEGINNING AT 0F830H.
          65 ;*****
          66 ;*
          67 ;*                               SYMBOL DEFINITIONS                               *
          68 ;*
          69 ;*****
          70 ;
          71 ; INTELLEC SERIES II SYSTEM CONSTANTS
          72 ;
          73 ; INTEGRATED CONSOLE I/O PORTS
          74 ;
00C0          75 CONI    EQU     0C0H      ; CONSOLE INPUT DATA PORT
00C0          76 CONO    EQU     0C0H      ; CONSOLE OUTPUT DATA PORT
00C1          77 CONS    EQU     0C1H      ; CONSOLE STATUS PORT
00C1          78 CONC    EQU     0C1H      ; CONSOLE CONTROL PORT
          79 ;
          80 ; SYSTEM BOOTSTRAP CONSTANTS (ISSUED TO PORT CPUC)
          81 ;
000D          82 DISABL  EQU     0DH        ; DISABLE INTERRUPTS
0005          83 ENABL   EQU     05H        ; ENABLE INTERRUPTS
0000          84 DISAXP  EQU     00H        ; DISABLE AUXILIARY PROM
          85
0008          86 ENAXP   EQU     08H        ; ENABLE AUXILIARY PROM
0001          87 BOVROF  EQU     01H        ; TURN OFF BUS OVERRIDE
0009          88 BOVRON  EQU     09H        ; TURN ON BUS OVERRIDE
0004          89 BTDGOF  EQU     04H        ; TURN OFF BOOT/DIAGNOSTIC
000C          90 BTDGON  EQU     0CH        ; TURN ON BOOT/DIAGNOSTIC
0002          91 MOVBOT  EQU     02H        ; MOVE BOOT TO 0E800H
          92 ;
          93 ; SYSTEM I/O PORTS
          94 ;
00FE          95 CPUS    EQU     0FEH      ; CPU STATUS PORT
00FF          96 CPUC    EQU     0FFH      ; CPU CONTROL PORT (CONTROLS BOOT & AUX.PROM)
          97 ;
          98 ; SYSTEM INTERRUPT CONSTANTS
          99 ;
0012          100 ICW1   EQU     00010010B   ; INITIALIZATION COMMAND WORD 1
0000          101 ICW2   EQU     00000000B   ; INITIALIZATION COMMAND WORD 2
000B          102 OCW3   EQU     00001011B   ; OPERATION COMMAND WORD 3
0020          103 EOI    EQU     00100000B   ; END OF INTERRUPT
          104 ;
          105 ; SYSTEM INTERRUPT MASKS AND VALUES
          106 ;
0001          107 INT0   EQU     00000001B   ; MASK FOR INTERRUPT LEVEL 0
0002          108 INT1   EQU     00000010B
0004          109 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      00000001B      ; 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    0000001B       ; KEYBOARD READY WITH DATA
0001		250	FRDY    EQU    0000001B       ; 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	RDR    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	;
		319	;
		320	FALSE EQU 0
FFFF		321	TRUE EQU NOT FALSE
0FFF		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    1111100B            ; MASK FOR LOCAL CONSOLE I/O
00F3		335	RMSK    EQU    1110011B            ; 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
00FF		385	ITIMO EQU	0FFH ; IOC TIMEOUT CONSTANT
00FF		386	LBMK EQU	0FFH ; LOWER BYTE MASK
0041		387	ICFG EQU	041H ; CONSOLE CONFIGURATION STATUS
0001		388	ICNP EQU	001H ; INTEGRATED CONSOLE NOT PRESENT STATUS
0040		389	LSTE EQU	040H ; LIST DEVICE VALUE FOR CONSOLE
04CD		390	RTCC EQU	1229 ; REAL TIME CLOCK IMS CONSTANT
0008		391	DPRNT EQU	08H ; DISK READY MASK
0D00		392	TRKL EQU	26*128 ; TRACK LENGTH
0004		393	PARML EQU	4 ; PARAMETER LENGTH - 1
F809		394	COP EQU	0F809H ; ENTRY POINT FOR CONSOLE OUT
F821		395	IOCDP1 EQU	0F821H ; ENTRY POINT FOR IOC DRIVER 1
F844		396	IOCDP2 EQU	0F844H ; ENTRY POINT FOR IOC DRIVER 2
		397	;*****	
		398	;	
		399	; PAGE 0 DEDICATED RAM LOCATIONS, INITIALIZED BY BOOTSTRAP PROM CODE.	
		400	;	
0000		401	ORG	0
		402	RESET:	
0000		403	DS	3 ; TRAP TO MONITOR RESTART
		404	IOBYT:	
0003		405	DS	1 ; I/O SYSTEM STATUS BYTE
		406	MEMTOP:	
0004		407	DS	2 ; TOP OF RAM, ONLY H SAVED
		408	INITIO:	
0006		409	DS	1 ; INITIAL I/O CONFIGURATION
		410	;*****	
		411	;	
		412	; BOOTSTRAP PROM CODE	
		413	;	
E800		414	BBASE SET	0E800H
E800		415	ORG	BBASE
E800 C306E8		416	JMP	BS0 ; BRANCH AROUND DATE CODE BYTE
		417	INIT:	
E803 00		418	DB	0 ; INITIALLY
		419		; CONSOLE = TTY,
		420		; READER = TTY,
		421		; PUNCH = TTY,
		422		; LIST = TTY
E804 0301		423	DW	DATE ; DATE STAMP FOR BOOTSTRAP PROM
		424	;	
		425	; FUNCTIONS:	
		426	;	
		427	A.	INITIALIZE INTERRUPT SYSTEM AND REAL TIME CLOCK
		428		0. INITIALIZE PORT 0FFH (CPUC)
		429		1. PROGRAM SYSTEM INTERRUPTS (8259)
		430		2. MASK ALL SYSTEM INTERRUPTS BUT TRAP LOGIC
		431		3. PROGRAM I/O SUBSYSTEM INTERRUPTS (8259)
		432		4. MASK ALL I/O SUBSYSTEM INTERRUPTS
		433		5. PROGRAM REAL TIME CLOCK
		434	;	
		435	BS0:	
E806 F3		436	DI	; DISABLE INTERRUPT SYSTEM
E807 3E02		437	MVI	A,MOVBOT ; TURN ON RAM (ROM WILL NOW RESPOND ONLY TO ADDRESS E800H)
E809 D3FF		438	OUT	CPUC
E80B 3E01		439	MVI	A,BOVROF ; TURN OFF BUS OVERRIDE

LOC	OBJ	LINE	SOURCE STATEMENT
E80D	D3FF	440	OUT CPUC
E80F	3E05	441	MVI A,ENABL ; PSEUDO ENABLE OF INTERRUPTS
E811	D3FF	442	OUT CPUC
E813	3E08	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 SICP0 ; TO SYSTEM 8259
E81B	D3FB	447	OUT IICP0 ; TO I/O 8259
E81D	3E00	448	MVI A,ICW2 ; OUTPUT INITIALIZATION COMMAND WORD 2
E81F	D3FC	449	OUT SICP1 ; TO SYSTEM 8259
E821	D3FA	450	OUT IICP1 ; TO I/O 8259
E823	3EFE	451	MVI A,NOT INT0 ; 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 IMS REAL TIME CLOCK
E82D	D3F3	456	OUT ITPC
E82F	21CD04	457	LXI H,RTCC
E832	7D	458	MOV A,L
E833	D3F2	459	OUT CTR2P
E835	7C	460	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
		470 ;	USER STACK
		471 ;	MONITOR STACK
		472 ;	RESTART ROUTINE JUMP ADDRESS
		473 ;	
E838	210000	474	LXI H,0 ; INITIAL VALUE H:=0, L:=0
		475 BS1:	
E83B	24	476	INR H ; INCREMENT BY 256 BYTE PAGES, I.E.100H,200H,...,F800H
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	480	CMP M ; IS LOCATION READ/WRITE, I.E. EXISTING RAM
E840	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 0E7FFH
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 0E7FFH IN RAM
		489	; AT THIS POINT WE HAVE CONTIGUOUS RAM UP TO
		490	; 0E7FFH; SKIP OVER 0E800-EFFFH WHICH IS
		491	; SHADOWED BY BOOT ROM AND THEREFORE
		492	; INACCESSIBLE; CONTINUE TESTING RAM FROM
		493	; 0F000H
E84C	2100EF	494	LXI H,0EF00H

LOC	OBJ	LINE	SOURCE STATEMENT
		495	BS1X:
E84F	24	496	INR    H                    ; INCREMENT BY 256 BYTE PAGES
E850	7E	497	MOV    A,M                ; FETCH CONTENTS OF MEMORY
E851	2F	498	CMA                        ; INVERT IT
E852	77	499	MOV    M,A                ; ATTEMPT TO WRITE IT BACK INTO MEMORY
E853	BE	500	CMP    M                   ; IS LOCATION READ/WRITE, I.E. EXISTING RAM
E854	2F	501	CMA                        ; INVERT IT BACK AGAIN TO ORIGINAL VALUE
E855	77	502	MOV    M,A                ; WRITE ORIGINAL DATA VALUE BACK IN
E856	CA4FE8	503	JZ     BS1X                ; YES, CONTINUE (I.E. STILL CONTIGUOUS RAM)
E859	2B	504	DCX    H                   ; OTHERWISE HL POINT TO LAST CONTIGUOUS
		505	;    BYTE OF RAM
E85A	3EF0	506	MVI    A,0F0H
E85C	BC	507	CMP    H                   ; TEST IF H > 0F0H (I.E. THAT TOP OF MEMORY
		508	; IS AT LEAST 512 BYTES ABOVE SHADOW BOOT
		509	; ROM BECAUSE WE NEED SPACE FOR MONITOR
		510	; WORK TEMPLATE)
E85D	DA62E8	511	JC     BS2                ; IF H > 0F0H THEN CARRY=1 AND HL CONTAIN
		512	; TRUE TOP OF MEMORY
E860	26E7	513	MVI    H,FSTOP            ; OTHERWISE H <= 0F0H THEN CARRY=0, SO
		514	; SET TOP OF MEMORY TO 0E7FFH, WHICH IS
		515	; JUST BELOW THE START OF SHADOW BOOT ROM
		516	BS2:
E862	220400	517	SHLD   MEMTOP            ; STORE TOP OF MEMORY
E865	01C8EA	518	LXI    B,TOS              ; MOVE EXIT TEMPLATE TO RAM
E868	69	519	MOV    L,C
E869	F9	520	SPHL                      ; SET MONITOR'S STACK POINTER
		521	BS3:
E86A	0A	522	LDAX   B
E86B	77	523	MOV    M,A
E86C	0C	524	INR    C                   ; MOVE BOTH POINTERS
E86D	2C	525	INR    L
E86E	C26AE8	526	JNZ    BS3                ; END ON PAGE BOUNDARY
E871	2ED1	527	MVI    L,SLOC AND HMSK   ; SET UP INITIAL VALUE FOR USER STACK
E873	74	528	MOV    M,H                ; LOWER HALF OF STACK POINTER IS KNOWN
E874	35	529	DCR    M                   ; MERELY SET UPPER HALF
		530	; TRAP TO MONITOR (AT LOCATIONS 0,1,2)
E875	3EC3	531	MVI    A,(JMP RESTART)
E877	320000	532	STA    RESET
E87A	21D4FE	533	LXI    H,RESTART        ; SET UP RESTART 0 FOR BREAKPOINT
E87D	220100	534	SHLD   RESET+1           ; LOGIC
		535	;
		536	;
		537	C.    PROGRAM I/O DEVICES.
		538	1. BAUD RATE GENERATOR FOR CRT
		539	2. USART FOR CRT
		540	3. BAUD RATE GENERATOR FOR TTY
		541	4. USART FOR TTY
		542	;
E880	3E76	542	MVI    A,CTRLS OR MODE3 OR RLLM
E882	D3F3	543	OUT    ITCP
E884	212000	544	LXI    H,B2400            ; CRT BAUD RATE
E887	7D	545	MOV    A,L
E888	D3F1	546	OUT    CTRLP
E88A	7C	547	MOV    A,H
E88B	D3F1	548	OUT    CTRLP
E88D	3ECE	549	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	DAAE8	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	605	XRA	A	
E8EB	2F	606	CMA		; A-REG = 0FFH
E8EC	F5	607	PUSH	PSW	; THREE-VALUED FLAG:
		608			;   0FFH IF NEITHER FDCC NOR ISD SELECTED
		609			;   00H IF FDCC SELECTED
		610			;   01H IF ISD SELECTED
E8ED	DB78	611	IN	DSTS	; SAMPLE FDCC STATUS
		612			; STATUS = 00H IF NO CONTROLLER PRESENT
E8EF	E608	613	ANI	00001000B	; IS FDCC CONTROLLER PRESENT?
E8F1	CA20E9	614	JZ	BS11	; JUMP TO ISD SECTION IF FDCC NOT PRESENT
E8F4	DB78	615	IN	DSTS	; SAMPLE FDCC STATUS AGAIN
E8F6	0F	616	RRC		; DRIVE 0 READY STATUS ROTATED INTO CARRY BIT
E8F7	D28EE9	617	JNC	BSX1	; JUMP TO MONITOR IF FDCC CONTROLLER PRESENT
		618			; AND DRIVE 0 NOT READY
		619			; THE FOLLOWING CODE IS USED TO WRITE THE DISK IOPB TO
		620			; PROCESSOR MEMORY SO THAT IF ICE IS BEING USED TO DEBUG
		621			; THE BOOT/MONITOR, THE DISK CONTROLLER CAN ACCESS THE IOPB
E8FA	210010	622	LXI	H,1000H	; LOAD POINTER TO DESTINATION MEMORY
E8FD	1134EA	623	LXI	D,IOPB	; LOAD POINTER TO SOURCE MEMORY FOR IOPB
E900	0607	624	MVI	B,7	; LOAD IOPB LENGTH COUNT
		625	MLP:		
E902	1A	626	LDAX	D	; LOAD BYTE OF IOPB
E903	77	627	MOV	M,A	; MOVE TO MEMORY
E904	23	628	INX	H	; INCREMENT IOPB POINTER
E905	13	629	INX	D	; INCREMENT MEMORY POINTER
E906	05	630	DCR	B	; DECREMENT IOPB LENGTH COUNT
E907	C202E9	631	JNZ	MLP	; CONTINUE UNTIL ALL OF IOPB MOVED
E90A	210010	632	LXI	H,1000H	; RELOAD POINTER TO IOPB
E90D	7D	633	MOV	A,L	; A CONTAINS LSB OF IOPB ADDRESS
E90E	D379	634	OUT	LOWW	; LOW(IOPB)
E910	7C	635	MOV	A,H	; A CONTAINS MSB OF IOPB ADDRESS
E911	D37A	636	OUT	HI	; HIGH(IOPB), START DISK I/O
		637	BS10:		
E913	DB78	638	IN	DSTS	; WAIT FOR FDCC TO COMPLETE
E915	E604	639	ANI	OPCPL	; TEST FOR DISK COMPLETION
E917	CA13E9	640	JZ	BS10	
E91A	F1	641	POP	PSW	; GET READY TO SET FLAG TO NEW VALUE
E91B	AF	642	XRA	A	; SET A TO ZERO TO INDICATE DRIVE OTHER THAN INTEGRATED
		643			; FLOPPY WAS ACCESSED CORRECTLY
E91C	F5	644	PUSH	PSW	; SAVE ON STACK
E91D	C38EE9	645	JMP	BSX1	; BYPASS INTEGRATED FLOPPY BOOT
		646			
		647			; LOAD ISIS.T0 FROM INTEGRATED DISK IF AVAILABLE
		648			
		649	BS11:		
E920	2A0400	650	LHLD	MEMTOP	; LOAD TOP OF MEMORY PAGE ADDRESS
E923	2ECC	651	MVI	L,BLOC+1 AND LBMK	; LOAD CONFIGURATION ADDRESS
E925	7E	652	MOV	A,M	
E926	0F	653	RRC		; TEST FOR INTEGRATED CONSOLE PRESENT
E927	DA8EE9	654	JC	BSX1	; JUMP IF IOC NOT PRESENT OR FUNCTIONAL
E92A	061C	655	MVI	B,RDSTS	; LOAD FLOPPY DEVICE STATUS COMMAND
E92C	CD21F8	656	CALL	IOCDP1	; READ STATUS FROM I/O CONTROLLER
E92F	E608	657	ANI	DPRT	; TEST FOR DRIVE PRESENT
E931	CA8EE9	658	JZ	BSX1	; JUMP IF NOT PRESENT
E934	061C	659	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	2A0400	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	0F	718	RRC ; TEST FOR INTEGRATED CONSOLE PRESENT
E995	DAA4E9	719	JC BSX2 ; JUMP IF INTEGRATED CONSOLE NOT PRESENT
E998	0613	720	MVI B,KSTS ; LOAD KEYBOARD STATUS COMMAND
E99A	CD21F8	721	CALL IOCDP1 ; READ STATUS FROM IOC
E99D	0F	722	RRC ; TEST FOR KEYBOARD PRESENT
E99E	0F	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	730	MOV D,A ; D CONTAINS 0H, I.E.C=T,R=T,P=T,L=T
E9A6	DBF5	731	IN TTY5 ; GET TTY STATUS
E9A8	E602	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	E602	739	ANI RRDY ; IS IT READY/
E9B8	CAA4E9	740	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	FE20	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	2A0400	750	LHLD MEMTOP ; LOAD TOP OF MEMORY PAGE ADDRESS
E9C7	2ECC	751	MVI L,BLOC+1 AND LBMK ; LOAD CONFIGURATION ADDRESS
E9C9	3E01	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	210300	757	LXI H,IOBYT ; HL POINTS TO I/O STATUS BYTE
E9CF	72	758	MOV M,D ; REPLACE MODIFIED I/O STATUS BYTE
E9D0	2E06	759	MVI L,INITIO ; HL POINTS TO INITIAL I/O STATUS BYTE
E9D2	72	760	MOV M,D ; SET INITIAL I/O STATUS BYTE
		761	;
		762	;
		763	;
		764	G. CALL THE DIAGNOSTIC PROGRAM
E9D3	CD03EB	765	CALL DIAGBT
		766	;
		767	H. IF DISK IS READY, TRANSFER TO ISIS.T0
		768	;
E9D6	F1	768	POP PSW ; UNLOAD FLAG
E9D7	B7	769	ORA A ; SET CONDITION CODES

LOC	OBJ	LINE	SOURCE STATEMENT
E9D8	C2EAE9	770	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    BSX10         ; JUMP IF ERROR CONDITION
E9E1	DB78	774	IN     DSTS            ; SAMPLE FDCC STATUS
E9E3	0F	775	RRC                  ; IS IT READY?
E9E4	D20DEA	776	JNC    BSX9           ; JUMP TO MONITOR IF DISK NOT READY
		777	; OTHERWISE, PRIOR TO TRANSFERRING CONTROL
		778	;    TO T0.BOOT, WRITE AN INSTRUCTION TO
		779	;    TURN OFF BOOTSTRAP PROM
E9E7	C3FEE9	780	JMP    BSX8
		781	BSX6:
E9EA	07	782	RLC                  ; TEST FOR NON DISK ACCESS
E9EB	DA0DEA	783	JC     BSX9           ; JUMP IF NO ACCESS
E9EE	3AFE2F	784	LDA    TRK0-2         ; LOAD TEMPORARY STORAGE FOR RESULT BYTE
E9F1	B7	785	ORA    A               ; SET CONDITION CODES
E9F2	C218EA	786	JNZ    BSX10         ; JUMP IF ERROR CONDITION
E9F5	061C	787	MVI    B,RDSTS       ; LOAD FLOPPY DEVICE STATUS COMMAND
E9F7	CD21F8	788	CALL   IOCDP1        ; READ STATUS FROM I/O CONTROLLER
E9FA	0F	789	RRC                  ; TEST FOR DRIVE READY
E9FB	D20DEA	790	JNC    BSX9           ; JUMP IF NOT READY
		791	BSX8:
E9FE	3ED3	792	MVI    A,(OUT CPUC)   ; LOAD OUTPUT INSTRUCTION
EA00	32FE2F	793	STA    TRK0-2         ; STORE IN RAM BEFORE DISK BOOT
EA03	3EFF	794	MVI    A,CPUC        ; LOAD PORT ADDRESS
EA05	32FF2F	795	STA    TRK0-1
EA08	3E04	796	MVI    A,BTDGOF      ; TURN OFF BOOTSTRAP/DIAGNOSTIC ROM
EA0A	C3FE2F	797	JMP    TRK0-2        ; EFFECT IS SAME AS: MVI    A,BTDGOF
		798	;                            OUT    CPUC
		799	;                            JMP    TRK0
		800	;
		801	;
		802	;
		803	BSX9:
EA0D	213BEA	804	LXI    H,VERS        ; HL POINTS TO ADDRESS OF SIGN-ON MESSAGE
EA10	061B	805	MVI    B,LVER        ; B CONTAINS LENGTH OF MESSAGE
EA12	CD2AEA	806	CALL   PRM            ; PRINT SIGN-ON MESSAGE
		807	;
		808	;
		809	;
EA15	C300F8	810	JMP    BEGIN         ; AT THIS POINT, INTERRUPTS ARE DISABLED
		811	;
		812	;
		813	;
		814	BSX10:
EA18	2156EA	815	LXI    H,ERMSG       ; HL POINTS TO ADDRESS OF DISK ERROR MESSAGE
EA1B	060E	816	MVI    B,LERM        ; B CONTAINS LENGTH OF MESSAGE
EA1D	CD2AEA	817	CALL   PRM            ; PRINT SIGN-ON MESSAGE
EA20	C30DEA	818	JMP    BSX9           ; PRINT MESSAGE
		819	;*****
		820	;
		821	;
		822	;
		823	BDLY:
EA23	0E70	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	;*****



```

LOC  OBJ          LINE      SOURCE STATEMENT
EAF4  C30000      920        JMP      0
                      921 P2LOC:
EAF7  C30000      922        JMP      0
                      923 L1LOC:
EAF8  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          LINE          SOURCE STATEMENT
                                1026 ; MAIN COMMAND LOOP.
                                1027 ;
                                1028 ; THIS LOOP IS THE STARTING POINT OF ALL COMMAND SEQUENCES.
                                1029 ; IT IS ENTERED VIA A JUMP FROM THE BEGINNING OF THE MONITOR PROPER CODE,
                                1030 ; A FALL THROUGH FROM THE ERROR ROUTINE, OR A RETURN FROM A MONITOR COMMAND
                                1031 ; ROUTINE.
                                1032 ; IN THIS CODE INTERRUPTS ARE ENABLED AND A CARRIAGE RETURN
                                1033 ; AND LINE FEED ARE TYPED ALONG WITH THE PROMPT CHARACTER, '.'.
                                1034 ; WHEN A CHARACTER IS ENTERED FROM THE LOCAL CONSOLE KEYBOARD, IT
                                1035 ; IS CHECKED FOR VALIDITY, THEN A BRANCH TO THE PROPER
                                1036 START0:
F851 3E04          1037          MVI    A,BTDGOF      ; DISABLE BOOT, I.E. SWITCH BOOT PROM
F853 D3FF          1038          OUT    CPUC          ;   OUT OF ADDRESSABLE MEMORY SPACE
                                1039 START:
F855 FB           1040          EI              ; ENABLE INTERRUPTS
F856 CDFEFD       1041          CALL   CRLF         ; TYPE <CR>,<LF>
F859 CDDEFC       1042          CALL   COMC        ; OUTPUT A PERIOD
F85C 2E           1043          DB      '.'
F85D CD61FF       1044          CALL   TI           ; GET A CHARACTER, ECHO IT.
F860 FE0D         1045          CPI     CR          ; IS IT A CARRIAGE RETURN?
F862 CA55F8       1046          JZ     START        ; JUMP IF IT IS
F865 D641         1047          SUI    'A'         ; OTHERWISE TEST FOR A-Z (VALID COMMAND RANGE)
F867 FA47F8       1048          JM     ERROR       ; LESS THAN A, NOT A VALID COMMAND
F86A 0E02         1049          MVI    C,2         ; ASSUME THE COMMAND NEEDS 2 PARAMETERS
F86C 1155F8       1050          LXI   D,START      ; SET UP PSEUDO RETURN ADDRESS TO SIMULATE
F86F D5           1051          PUSH   D           ;   EFFECT OF A CALL. COMMANDS WHICH PERFORM
                                1052          ;   A RETURN WILL CAUSE THE STACK TO BE
                                1053          ;   POPPED, THUS RETURNING TO ENTRY POINT
                                1054          ;   START. THE 'G' COMMAND, HOWEVER, WIPES
                                1055          ;   OUT THIS ADDRESS WITH ANOTHER ADDRESS
                                1056          ;   OF ITS OWN CHOOSING (I.E. USER'S PC).
F870 2182F8       1057          LXI   H,CTBL       ; LOAD POINTER TO PROCESSING ROUTINE PTRS
F873 FE1A         1058          CPI     LCT        ; TEST FOR OVERRUN
F875 F247F8       1059          JP     ERROR       ; IF SO, THEN ERROR
F878 5F           1060          MOV    E,A         ; OTHERWISE, MOVE INDEX TO DE
F879 1600         1061          MVI    D,0
F87B 19           1062          DAD    D
F87C 19           1063          DAD    D           ; HL := CTBLBASE + (2 * INDEX); HL NOW POINTS
                                1064          ;   TO PROPER COMMAND IN COMMAND BRANCH TABLE
F87D 7E           1065          MOV    A,M         ; GET LSB OF BRANCH LOCATION
F87E 23           1066          INX   H
F87F 66           1067          MOV    H,M         ; GET MSB OF BRANCH LOCATION
F880 6F           1068          MOV    L,A         ; HL POINTS TO ADDRESS OF COMMAND CODE
F881 E9           1069          PCHL          ; TAKE THE BRANCH
                                1070 ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
                                1071 ;
                                1072 ; COMMAND BRANCH TABLE.
                                1073 ;
                                1074 ; THIS TABLE CONTAINS THE ADDRESSES OF THE ENTRY POINTS OF
                                1075 ; ALL THE COMMAND PROCESSING ROUTINES. IT IS ENTERED FROM THE MAIN
                                1076 ; COMMAND LOOP. NOTE THAT AN ENTRY TO 'ERROR'
                                1077 ; IS AN ERROR CONDITION, I.E., NO COMMAND CORRESPONDING TO THAT
                                1078 ; CHARACTER EXISTS.
                                1079 CTBL:
F882 B6F8         1080          DW     ASSIGN      ; A - ASSIGN I/O UNITS

```



LOC	OBJ	LINE	SOURCE STATEMENT
F884	47F8	1081	DW    ERROR                    ; B -
F886	47F8	1082	DW    ERROR                    ; C -
F888	33F9	1083	DW    DISP                     ; D - DISPLAY RAM MEMORY
F88A	5FF9	1084	DW    EOF                      ; E - ENDFILE A HEXADECIMAL FILE
F88C	7DF9	1085	DW    FILL                     ; F - FILL MEMORY
F88E	8CF9	1086	DW    GOTO                     ; G - GO TO MEMORY ADDRESS
F890	D5F9	1087	DW    HEXN                     ; H - HEXADECIMAL SUM AND DIFFERENCE
F892	47F8	1088	DW    ERROR                    ; I -
F894	47F8	1089	DW    ERROR                    ; J -
F896	47F8	1090	DW    ERROR                    ; K -
F898	47F8	1091	DW    ERROR                    ; L -
F89A	F0F9	1092	DW    MOVE                     ; M - MOVE MEMORY
F89C	01FA	1093	DW    NULL                     ; N - PUNCH NULLS FOR LEADER ON PAPER TAPE
F89E	47F8	1094	DW    ERROR                    ; O -
F8A0	47F8	1095	DW    ERROR                    ; P -
F8A2	14FA	1096	DW    QUERY                    ; Q - QUERY I/O SYSTEM STATUS
F8A4	52FA	1097	DW    READ                     ; R - READ HEXADECIMAL PAPER TAPE FILE
F8A6	BFFA	1098	DW    SUBS                     ; S - SUBSTITUTE MEMORY
F8A8	47F8	1099	DW    ERROR                    ; T -
F8AA	47F8	1100	DW    ERROR                    ; U -
F8AC	47F8	1101	DW    ERROR                    ; V -
F8AE	DDFA	1102	DW    WRITE                    ; W - WRITE FILE TO PAPER TAPE IN HEX FORMAT
F8B0	26FB	1103	DW    X                        ; X - EXAMINE AND MODIFY REGISTERS
F8B2	47F8	1104	DW    ERROR                    ; Y -
F8B4	A6FB	1105	DW    Z                        ; Z - INVOKE THE DIAGNOSTIC PROGRAM
001A		1106	LCT   EQU    (\$-CTBL)/2       ; LCT = NUMBER OF 16-BIT ENTRIES IN TABLE
		1107	; *-*
		1108	;
		1109	; 'A' COMMAND - ASSIGN I/O DEVICE
		1110	;
		1111	; THIS ROUTINE MAPS SYMBOLIC DEVICE IDENTIFIERS TO BITS
		1112	; IN THE I/O STATUS BYTE (IOBYT) TO ALLOW FOR LOCAL CONSOLE
		1113	; MODIFICATION OF SYSTEM I/O CONFIGURATION.
		1114	ASSIGN:
F8B6	CD61FF	1115	CALL   TI                     ; GET LOGICAL DEVICE CHARACTER (C,R,P,L)
F8B9	2103F9	1116	LXI    H,LTBL                ; ADDRESS OF MASTER TABLE
F8BC	0E04	1117	MVI    C,4                   ; MAXIMUM OF 4 ENTRIES
		1118	;
		1119	AS0:                         ; HL POINTS TO IDENTIFYING CHARACTER IN LTBL
F8BE	BE	1120	CMP    M                     ; DOES A-REG CONTAIN C,R,P, OR L?
F8BF	23	1121	INX    H                     ; HL POINTS TO CORRESPONDING DEVICE MASK
F8C0	CACDF8	1122	JZ     AS1                   ; YES IT DOES
F8C3	23	1123	INX    H                     ;
F8C4	23	1124	INX    H                     ;
F8C5	23	1125	INX    H                     ; HL POINTS TO NEXT 4-BYTE ENTRY IN LTBL
F8C6	0D	1126	DCR    C                     ; DECREMENT LOOP COUNT
F8C7	C2BEF8	1127	JNZ    AS0                   ; TRY NEXT ENTRY
F8CA	C347F8	1128	JMP    ERROR                ; NO MATCH, ERROR
		1129	;
		1130	AS1:                         ; USER HAS SPECIFIED A VALID LOGICAL DEVICE
F8CD	46	1131	MOV    B,M                   ; B := LOGICAL DEVICE MASK
F8CE	23	1132	INX    H                     ; HL CONTAINS SUBORDINATE PHYS.DEV.TBL.ADDRESS
F8CF	5E	1133	MOV    E,M                   ; E CONTAINS LSB OF PDT ADDRESS
F8D0	23	1134	INX    H                     ;
F8D1	56	1135	MOV    D,M                   ; D CONTAINS MSB OF PDT ADDRESS

```

LOC OBJ          LINE          SOURCE STATEMENT
F8D2 EB          1136          XCHG                      ; HL POINTS TO I/O SYSTEM PHYSICAL DEVICE
                  1137                      ;   TABLE (I.E. ACT,ART,APT, OR ALT)
                  1138          ; -----
                  1139 ALUP1:          ; SCAN INPUT UNTIL '='
F8D3 CD61FF     1140          CALL    TI
F8D6 FE3D       1141          CPI    '='
F8D8 C2D3F8     1142          JNZ    ALUP1
                  1143          ; -----
                  1144 ALUP2:          ; SCAN INPUT WHILE ' ' (BLANK)
F8DB CD61FF     1145          CALL    TI
F8DE FE20       1146          CPI    ' '
F8E0 CADBF8     1147          JZ     ALUP2
                  1148          ; -----
F8E3 0E04       1149          MVI    C,4                ; SET TABLE LENGTH
                  1150 AS2:          ; INDEX THROUGH PHYSICAL UNIT TABLE
F8E5 BE         1151          CMP    M                  ; COMPARE DEVICE CHAR WITH LEGAL VALUES
F8E6 23         1152          INX   H                  ; HL CONTAINS DEVICE SELECT BIT PATTERN
F8E7 CAF2F8     1153          JZ     AS3                ; USER HAS SPECIFIED A VALID PHYS.DEVICE ASSIGNMNT
F8EA 23         1154          INX   H                  ; HL POINTS TO NEXT ENTRY WITHIN THE TABLE
F8EB 0D         1155          DCR   C
F8EC C2E5F8     1156          JNZ   AS2                ; CONTINUE LOOKUP
F8EF C347F8     1157          JMP   ERROR              ; ERROR RETURN
                  1158          ; -----
                  1159 AS3:          ; SCAN INPUT UNTIL <CR>
                  1160 ALUP3:          ;
F8F2 CD61FF     1161          CALL    TI
F8F5 FE0D       1162          CPI    CR
F8F7 C2F2F8     1163          JNZ   ALUP3
F8FA 3A0300     1164          LDA   IOBYT              ; GET I/O STATUS
F8FD A0         1165          ANA   B                  ; B CONTAINS LOG DEV MASK.  CLEAR OUT THE
                  1166                      ;   APPROPRIATE FIELD IN IOBYT BECAUSE WE ARE
                  1167                      ;   GOING TO CHANGE IT.
F8FE B6         1168          ORA   M                  ; PUT IN THE NEW STATUS FIELD
F8FF 320300     1169          STA   IOBYT              ; RETURN IT TO MEMORY
F902 C9         1170          RET                      ; RETURN CONTROL TO MAIN COMMAND LOOP
                  1171          ;
                  1172 ; MASTER I/O DEVICE TABLE
                  1173 ; 4 BYTES/ENTRY
                  1174          ;
                  1175 ;   BYTE 0 = IDENTIFYING CHARACTER
                  1176 ;   BYTE 1 = LOGICAL DEVICE MASK
                  1177 ;   BYTES 2,3 = ADDRESS OF SUBORDINATE PHYSICAL DEVICE TABLE
                  1178          ;
                  1179 LTBL:
F903 43         1180          DB    'C',CMASK
F904 FC
F905 13F9       1181          DW    ACT
F907 52         1182          DB    'R',RMSK
F908 F3
F909 1BF9       1183          DW    ART
F90B 50         1184          DB    'P',PMSK
F90C CF
F90D 23F9       1185          DW    APT
F90F 4C         1186          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	LINE	SOURCE STATEMENT
		1226	DISP:
F933	CD39FE	1227	CALL    EXPR            ; GET TWO ADDRESSES
F936	D1	1228	POP     D                ; GET HIGH ADDRESS
F937	E1	1229	POP     H               ; GET LOW ADDRESS
		1230	DI0:
F938	CD6AFE	1231	CALL    LCRLF          ; PRINT CR,LF
F93B	CD07FE	1232	CALL    DADR            ; PRINT MEMORY ADDRESS
		1233	DI1:
F93E	0E20	1234	MVI     C, ' '
F940	CD14FD	1235	CALL    LOM             ; PRINT SPACE
F943	7E	1236	MOV     A,M
F944	CD0CFE	1237	CALL    DBYTE          ; PRINT DATA
F947	CD4CFE	1238	CALL    HILO            ; TEST FOR COMPLETION
F94A	DA56F9	1239	JC      DI2             ; RETURN TO MAIN LOOP
F94D	7D	1240	MOV     A,L
F94E	E60F	1241	ANI     0FH             ; PRINT CR,LF,ADDRESS ON MULTIPLE OF 16
F950	C23EF9	1242	JNZ     DI1
F953	C338F9	1243	JMP     DI0
		1244	DI2:
F956	CD6AFE	1245	CALL    LCRLF          ; WRITE CR,LF
F959	0E00	1246	MVI     C,0
F95B	CD14FD	1247	CALL    LOM             ; WRITE A NULL TO TRIGGER CLOSE
F95E	C9	1248	RET
		1249	;*****
		1250	;
		1251	; 'E' COMMAND - PUNCH HEXADECIMAL END-OF-FILE
		1252	;
		1253	; THIS ROUTINE PRODUCES A TERMINATION RECORD WHICH PROPERLY
		1254	; COMPLETES A HEXADECIMAL FILE CREATED BY 'W' COMMANDS.
		1255	; IT EXPECTS ONE HEXADECIMAL PARAMETER, WHICH IT INTERPRETS AS THE
		1256	; START ADDRESS TO BE LOADED INTO THE USER'S PROGRAM COUNTER (LOCATED
		1257	; IN EXIT TEMPLATE) ON A SUBSEQUENT 'R' COMMAND; THIS START ADDRESS
		1258	; WILL REPLACE THE STORED VALUE OF THE USER'S PROGRAM COUNTER ONLY
		1259	; IF THE START ADDRESS IS NONZERO.
		1260	;
		1261	EOF:
F95F	0D	1262	DCR     C                ; C:=1; GET ONE PARAMETER
F960	CD39FE	1263	CALL    EXPR            ; PUT <START ADDRESS> ON TOP OF STACK
F963	CDE5FC	1264	CALL    POC             ; OUTPUT RECORD MARK (':')
F966	3A	1265	DB      ':'
F967	AF	1266	XRA     A                ; ZERO CHECKSUM
F968	57	1267	MOV     D,A             ; D := 0; A := 0
F969	CDAFFE	1268	CALL    PBYTE          ; OUTPUT A RECORD LENGTH OF ZERO
F96C	E1	1269	POP     H                ; RETRIEVE START ADDRESS
F96D	CDAAFE	1270	CALL    PADR            ; OUTPUT IT AS THE LOAD ADDRESS
F970	3E01	1271	MVI     A,1             ; RECORD TYPE = 1
F972	CDAFFE	1272	CALL    PBYTE          ; OUTPUT RECORD TYPE
F975	AF	1273	XRA     A                ; A := 0
F976	92	1274	SUB     D                ; D CONTAINS RUNNING CHECKSUM
F977	CDAFFE	1275	CALL    PBYTE          ; OUTPUT CHECKSUM := -D
F97A	C309FA	1276	JMP     NU0             ; PUNCH TRAILER AND RETURN
		1277	;*****
		1278	;
		1279	; 'F' COMMAND - FILL RAM WITH 8-BIT CONSTANT
		1280	;

```

LOC  OBJ          LINE      SOURCE STATEMENT
                                1281 ; THIS ROUTINE EXPECTS THREE HEXADECIMAL PARAMETERS, THE
                                1282 ; FIRST AND SECOND (16 BITS) ARE INTERPRETED AS THE BOUNDS
                                1283 ; OF A MEMORY AREA TO BE INITIALIZED TO A CONSTANT VALUE,
                                1284 ; THE THIRD PARAMETER (8 BITS) IS THAT VALUE.
                                1285 FILL:
F97D  0C          1286      INR      C              ; C:=3; GET 3 PARAMETERS
F97E  CD39FE     1287      CALL    EXPR
F981  C1          1288      POP      B              ; C := 8-BIT CONSTANT
F982  D1          1289      POP      D              ; DE := HIGH ADDRESS
F983  E1          1290      POP      H              ; HL := LOW ADDRESS
                                1291 FI0:
F984  71          1292      MOV      M,C          ; STORE CONSTANT IN MEMORY
F985  CD4CFE     1293      CALL    HILO         ; TEST FOR COMPLETION
F988  D284F9     1294      JNC     FI0          ; CONTINUE LOOPING
F98B  C9          1295      RET
                                1296 ; *-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
                                1297 ;
                                1298 ; 'G' COMMAND - GO TO <ADDRESS>, OPTIONALLY SET BREAKPOINT(S)
                                1299 ;
                                1300 ; THE G COMMAND IS USED FOR TRANSFERRING CONTROL FROM THE
                                1301 ; MONITOR TO A USER PROGRAM. IT HAS SEVERAL MODES OF
                                1302 ; OPERATION.
                                1303 ; IF ONE HEXADECIMAL PARAMETER IS ENTERED, IT IS INTERPRETED
                                1304 ; AS THE ENTRY POINT OF THE USER PROGRAM AND A TRANSFER TO
                                1305 ; THAT LOCATION IS EXECUTED.
                                1306 ; IF ADDITIONAL (UP TO 2) PARAMETERS ARE ENTERED, THESE ARE
                                1307 ; CONSIDERED 'BREAKPOINTS', I.E., LOCATIONS WHERE
                                1308 ; CONTROL IS TO BE RETURNED TO THE MONITOR WHEN THEY ARE
                                1309 ; ENCOUNTERED IN COURSE OF EXECUTING THE USER PROGRAM.
                                1310 ; IF THE FIRST PARAMETER IS NOT ENTERED, THE STORED VALUE
                                1311 ; OF THE USER'S PROGRAM COUNTER (REGISTER P) IS USED AS
                                1312 ; THE USER PROGRAM ENTRY POINT.
                                1313 ;
                                1314 ; THIS COMMAND WORKS IN THE FOLLOWING MANNER:
                                1315 ; 1. IT FINDS THE EXIT CODE IN TOP OF RAM AND PLACES THIS ADDRESS IN THE
                                1316 ; MONITOR'S STACK, REPLACING THE RETURN ADDRESS TO ENTRY POINT START
                                1317 ; THAT WAS PLACED THERE BY THE MAIN COMMAND LOOP.
                                1318 ; 2. IF THERE IS NO FURTHER INPUT (I.E. ONLY <CR>) THEN BY EXECUTING A
                                1319 ; RET, WE CAUSE EXECUTION OF THE EXIT CODE, WHICH CONTAINS A JUMP TO
                                1320 ; A) A DUMMY ADDRESS (IF IMPROPER USE OF COMMAND), B) THE PROGRAM
                                1321 ; COUNTER FROM WHEN THE USER PROGRAM WAS INTERRUPTED OR BREAKPOINT
                                1322 ; WAS ENCOUNTERED.
                                1323 ; 3. IF THERE IS A START ADDRESS SPECIFIED, THIS VALUE IS STORED OVER
                                1324 ; THAT PART OF THE EXIT CODE WHICH CONTAINS THE JMP INSTRUCTION.
                                1325 ; IF THERE IS NO FURTHER INPUT, A RET IS EXECUTED AND THE EXIT
                                1326 ; CODE IS EXECUTED.
                                1327 ; 4. IF TRAPS (BREAKPOINTS) ARE TO BE SET, THEN THEY ARE READ IN AND PLACED
                                1328 ; ON THE MONITOR STACK. THEY ARE THEN STORED IN THE PROPER SECTION OF
                                1329 ; THE EXIT TEMPLATE. ALSO, IN THE USER'S PROGRAM THE INSTRUCTION SPECIFIED
                                1330 ; BY THE BREAKPOINT ADDRESS IS SAVED IN THE EXIT TEMPLATE AND REPLACED
                                1331 ; WITH A RST 0 INSTRUCTION.
                                1332 ; 5. THE EXIT CODE IS EXECUTED AND CONTROL IS PASSED TO THE USER PROGRAM.
                                1333 GOTO:
F98C  2A0400     1334      LHLD    MEMTOP
F98F  2ED2       1335      MVI     L,EXIT AND 0FFH ; HL NOW POINTS TO EXIT CODE IN TOP OF RAM

```

LOC	OBJ	LINE	SOURCE STATEMENT
F991	E3	1336	XTHL ; 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
		1340	; 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 GO0 ; IF ' ', ',', OR <CR>: JUMP, DON'T CHANGE PC
F998	CD7AFE	1344	CALL PA0 ; GET NEW PC VALUE
F99B	EB	1345	XCHG ; DE = NEW PC
F99C	2A0400	1346	LHLD MEMTOP
F99F	2EE1	1347	MVI L,PLOC AND 0FFH ; 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	1350	MOV M,E ; STORE LSB OF MODIFIED PC IN EXIT CODE IN RAM
		1351	GO0:
F9A4	DAD1F9	1352	JC GO4 ; JUMP IF <CR> (NO TRAPS TO BE SET)
F9A7	110200	1353	LXI D,2 ; SET COUNTER(S), D=0, 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	1360	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:
F9BA	D247F8	1364	JNC ERROR ; D CONTAINS HOW MANY TRAPS (1 OR 2)
F9BD	2A0400	1365	LHLD MEMTOP ; LAST TRAP NOT FOLLOWED BY CR
F9C0	2EE2	1366	MVI L,TLOC AND 0FFH ; 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	1370	MOV M,C ; STORE LSB OF TRAP ADDRESS INTO TRAP AREA
F9C4	23	1371	INX H
F9C5	70	1372	MOV M,B ; STORE MSB OF TRAP ADDRESS INTO TRAP AREA
F9C6	23	1373	INX H
F9C7	0A	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 0) ; REPLACE THE USER'S OPCODE IN USER PROGRAM
F9CC	02	1378	STAX B ; WITH A RST 0
F9CD	15	1379	DCR D
F9CE	C2C2F9	1380	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
		1390	; AND DISPLAYS THEM ON THE LOCAL CONSOLE DEVICE AS FOLLOWS:

LOC	OBJ	LINE	SOURCE STATEMENT
		1391	; <P1+P2> <P1-P2>
		1392	HEXN:
F9D5	CD39FE	1393	CALL    EXPR                   ; GET TWO NUMBERS
F9D8	CDFEFD	1394	CALL    CRLF
F9DB	D1	1395	POP     D                    ; DE CONTAINS P2
F9DC	E1	1396	POP     H
F9DD	E5	1397	PUSH    H                    ; HL CONTAINS P1
F9DE	19	1398	DAD     D                    ; HL := HL + DE := P1 + P2
F9DF	CD56FE	1399	CALL    LADR                 ; DISPLAY SUM
F9E2	CD93FC	1400	CALL    BLK                 ; TYPE A SPACE
F9E5	E1	1401	POP     H                    ; HL CONTAINS P1 AGAIN
F9E6	7D	1402	MOV     A,L                 ; COMPUTE HL-DE
F9E7	93	1403	SUB     E                    ; A := LSB OF P1 - LSB OF P2
F9E8	6F	1404	MOV     L,A                 ; A := LSB OF (P1 - P2)
F9E9	7C	1405	MOV     A,H
F9EA	9A	1406	SBB     D                    ; A := MSB OF P1 - MSB OF P2 WITH CARRY
F9EB	67	1407	MOV     H,A                 ; H := MSB OF (P1 -P2)
F9EC	CD56FE	1408	CALL    LADR                 ; DISPLAY DIFFERENCE
F9EF	C9	1409	RET
		1410	;*-*
		1411	;
		1412	; 'M' COMMAND - MOVE A BLOCK OF MEMORY
		1413	;
		1414	; THIS ROUTINE EXPECTS THREE HEXADECIMAL PARAMETERS FROM THE
		1415	; LOCAL CONSOLE. THE FIRST AND SECOND PARAMETERS ARE THE BOUNDS OF
		1416	; THE MEMORY AREA TO BE MOVED, THE THIRD PARAMETER IS THE
		1417	; STARTING ADDRESS OF THE DESTINATION AREA.
		1418	MOVE:
F9F0	0C	1419	INR     C                    ; GET THREE ADDRESSES
F9F1	CD39FE	1420	CALL    EXPR
F9F4	C1	1421	POP     B                    ; DESTINATION ADDRESS
F9F5	D1	1422	POP     D                    ; SOURCE END ADDRESS
F9F6	E1	1423	POP     H                    ; SOURCE START ADDRESS
		1424	MV0:
F9F7	7E	1425	MOV     A,M                 ; GET A DATA BYTE
F9F8	02	1426	STAX    B                    ; STORE AT DESTINATION
F9F9	03	1427	INX     B                    ; MOVE DESTINATION POINTER
F9FA	CD4CFE	1428	CALL    HILO                ; TEST FOR COMPLETION
F9FD	D2F7F9	1429	JNC     MV0
FA00	C9	1430	RET
		1431	;*-*
		1432	;
		1433	; 'N' COMMAND - PUNCH NULL CHARACTERS FOR TAPE LEADER/TRAILER
		1434	;
		1435	; THIS ROUTINE PUNCHES 60 NULL CHARACTERS ON THE DEVICE ASSIGNED
		1436	; AS THE PUNCH. IT IS ENTERED VIA A JUMP TO ENTRY POINT NU0
		1437	; FROM THE 'E' COMMAND AS WELL AS BEING INVOKED BY
		1438	; THE 'N' COMMAND.
		1439	NULL:
FA01	CD61FF	1440	CALL    TI                   ; REQUIRE CR
FA04	FE0D	1441	CPI     CR
FA06	C247F8	1442	JNZ     ERROR
		1443	NU0:
FA09	063C	1444	MVI     B,60                 ; SET TO PUNCH 60 NULLS
		1445	NLEADX:





LOC	OBJ	LINE	SOURCE STATEMENT	
		1501	;*****	
		1502	;	
		1503	; 'R' COMMAND - READ HEXADECIMAL FILE	
		1504	;	
		1505	; THIS ROUTINE READS A HEXADECIMAL FILE FROM THE ASSIGNED	
		1506	; READER DEVICE AND LOADS IT INTO MEMORY. ONE HEXADECIMAL	
		1507	; PARAMETER IS EXPECTED. THIS PARAMETER IS A BASE ADDRESS	
		1508	; TO BE ADDED TO THE MEMORY ADDRESS OF EACH DATA BYTE ENCOUNTERED.	
		1509	; IN THIS WAY, HEXADECIMAL FILES MAY BE LOADED INTO MEMORY	
		1510	; IN AREAS OTHER THAN THAT FOR WHICH THEY WERE ASSEMBLED OR COMPILED.	
		1511	; ALL RECORDS READ ARE CHECKSUMMED AND COMPARED AGAINST THE	
		1512	; CHECKSUM IN THE RECORD. IF A CHECKSUM ERROR (OR TAPE READ ERROR)	
		1513	; OCCURS, THE ROUTINE TAKES AN ERROR EXIT. NORMAL LOADING IS	
		1514	; TERMINATED WHEN AN EOF RECORD IS ENCOUNTERED. THE ADDRESS	
		1515	; GIVEN WHEN THE EOF RECORD WAS CREATED (VIA THE 'E' COMMAND) REPLACES	
		1516	; THE USER'S STORED PC VALUE ONLY IF THE ADDRESS WAS NONZERO.	
		1517	; A TRANSFER TO THE PROGRAM MAY THEN BE ACCOMPLISHED BY A 'G<CR>'.	
		1518	READ:	
FA52	0D	1519	DCR	C ; GET ONE ADDRESS; C := 1
FA53	CD39FE	1520	CALL	EXPR ; GET THE HEX BASE ADDRESS
FA56	CDFEPD	1521	CALL	CRLF ; OUTPUT A <CR>,<LF>
		1522	RED0:	
FA59	CD58FF	1523	CALL	RIX ; GET AN ASCII CHARACTER FROM THE READER
FA5C	FE3A	1524	CPI	':' ; IS IT A START OF RECORD MARK (':')?
FA5E	C259FA	1525	JNZ	RED0 ; LOOP UNTIL WE FIND SUCH A RECORD MARK
FA61	AF	1526	XRA	A
FA62	57	1527	MOV	D,A ; D WILL CONTAIN THE CHECKSUM; INITIALIZE TO 0
FA63	CDDBFD	1528	CALL	BYTE ; READ 2 ASCII CHAR REPRESENTING THE RECORD
		1529		; LENGTH AND DECODE THEM INTO 8 BITS BINARY
		1530		; STORING THE RESULT IN A-REG
FA66	CA9EFA	1531	JZ	RED3 ; JUMP IF ZERO RECORD LENGTH BECAUSE THIS
		1532		; MEANS IT'S AN EOF RECORD SO WE'RE DONE
FA69	5F	1533	MOV	E,A ; E := RECORD LENGTH
FA6A	CDDBFD	1534	CALL	BYTE ; GET MSB OF LOAD ADDRESS
FA6D	67	1535	MOV	H,A ; H := MSB OF LOAD ADDRESS
FA6E	CDDBFD	1536	CALL	BYTE ; GET LSB OF LOAD ADDRESS
FA71	6F	1537	MOV	L,A ; L := LSB OF LOAD ADDRESS
FA72	CDDBFD	1538	CALL	BYTE ; GET RECORD TYPE AND IGNORE IT
FA75	4B	1539	MOV	C,E ; C := RECORD LENGTH
FA76	E5	1540	PUSH	H ; STORE LOAD ADDRESS ON THE STACK
FA77	2100FF	1541	LXI	H,-256 ; COMPUTE BUFFER POINTER
FA7A	39	1542	DAD	SP ; HL NOW POINTS TO THAT PART OF THE MONITOR
		1543		; STACK ONE PAGE (256 BYTES) BELOW WHERE
		1544		; THE SP IS CURRENTLY POINTING
		1545		; WE WILL NOW READ DATA FROM THE FILE RECORD
		1546		; AND STORE THEM TEMPORARILY IN THE MONITOR'S
		1547		; STACK STARTING FROM A LOW MEMORY ADDRESS AND
		1548		; MOVING TOWARD A HIGHER MEMORY ADDRESS (REVERSE
		1549		; OF USUAL PROCEDURE WHERE STACK GROWS DOWN)
		1550	RED1:	
FA7B	CDDBFD	1551	CALL	BYTE ; READ DATA; NOTE: 8 BITS OF MEMORY (DATA)
		1552		; IS REPRESENTED AS 2 HEX CHAR AND EACH HEX
		1553		; HEX CHAR IS REPRESENTED AS ONE 8 BIT ASCII CHAR
FA7E	77	1554	MOV	M,A ; PUT DATA IN MONITOR BUFFER
FA7F	23	1555	INX	H ; MOVE "UP" THE STACK

```

LOC  OBJ          LINE          SOURCE STATEMENT
FA80  1D           1556          DCR      E           ; DECREMENT RECORD LENGTH COUNT
FA81  C27BFA      1557          JNZ      RED1        ; LOOP UNTIL RECORD LENGTH COUNTER IS 0
FA84  CDDDBFD     1558          CALL     BYTE        ; READ THE CHECKSUM RECORD FRAME --- PRIOR TO
                               1559          ; CALL TO BYTE, D-REG CONTAINED SUM OF DATA
                               1560          ; RECORDS. THE CHECKSUM FRAME SHOULD CONTAIN
                               1561          ; THE NEGATIVE OF THIS SUM. BYTE ADDS D AND A
                               1562          ; TOGETHER AND SETS THE ZERO BIT IF D = (-A)
FA87  C247F8     1563          JNZ      ERROR      ; CHECKSUM ERROR
FA8A  D1          1564          POP      D           ; DE = LOAD ADDRESS; STACK ENTRY POINTED TO BY SP
                               1565          ; NOW CONTAINS BASE (BIAS) ADDRESS
FA8B  E3          1566          XTHL                    ; HL = BIAS ADDRESS; CONTENTS OF STACK ENTRY
                               1567          ; POINTED TO BY SP NOW IS ADDRESS ONE ABOVE
                               1568          ; WHERE LAST DATA IS STORED IN MONITOR STACK
FA8C  EB          1569          XCHG                    ; DE = BIAS ADDRESS, HL = LOAD ADDRESS
FA8D  19          1570          DAD      D           ; HL = BIAS + LA
FA8E  0600        1571          MVI      B,0         ; BC = RECORD LENGTH (RL)
FA90  09          1572          DAD      B           ; HL = BIAS + LA + RL
FA91  EB          1573          XCHG                    ; DE = BIAS + LA + RL, HL = BIAS
FA92  E3          1574          XTHL                    ; HL POINTS TO ADDRESS 1 GREATER THAN WHERE LAST
                               1575          ; DATA IS STORED IN MONITOR STACK
                               1576          ;-----
                               1577  RED2:          ; LOAD INTO PROPER AREA IN RAM BUT IN
                               1578          ; REVERSE ORDER
FA93  2B          1579          DCX      H           ; DECREMENT STACK BUFFER POINTER
FA94  7E          1580          MOV      A,M         ; A := DATA
FA95  1B          1581          DCX      D           ; DECREMENT MEMORY POINTER
FA96  12          1582          STAX    D           ; PUT DATA IN DESIGNATED ADDRESS
FA97  0D          1583          DCR      C           ; KEEP DOING THIS UNTIL RECORD LENGTH
FA98  C293FA      1584          JNZ      RED2        ; COUNT IS EXHAUSTED
FA9B  C359FA      1585          JMP      RED0        ; DONE WITH ONE RECORD, GO GET ANOTHER
                               1586          ;-----
                               1587  RED3:          ; EOF RECORD - ENTIRE FILE HAS BEEN READ IN
FA9E  C5          1588          PUSH    B           ; SAVE B,C
FA9F  CDDDBFD     1589          CALL    BYTE        ; GET MSB OF LOAD ADDRESS OF EOF RECORD ---
                               1590          ; THIS IS THE <START ADDRESS> SPECIFIED IN
                               1591          ; THE 'E' COMMAND. IF IT IS ZERO, DO NOT
                               1592          ; MODIFY THE USER'S STORED PC IN EXIT TEMPLATE
FAA2  47          1593          MOV      B,A         ; B := MSB OF START ADDRESS
FAA3  CDDDBFD     1594          CALL    BYTE        ; GET LSB OF START ADDRESS
FAA6  4F          1595          MOV      C,A         ; C := LSB OF START ADDRESS
FAA7  B0          1596          ORA     B           ; SEE IF START ADDRESS IS 0000
FAA8  CAB3FA      1597          JZ      RED4        ; JUMP IF IT IS (DON'T SET NEW PC)
FAAB  2A0400      1598          LHLD   MEMTOP
FAAE  2EE1        1599          MVI    L,PLOC AND 0FFH ; HL POINTS TO PLOC IN EXIT CODE IN TOP OF RAM
FAB0  70          1600          MOV     M,B         ; STORE MSB OF START ADDRESS
FAB1  2B          1601          DCX    H           ; HL POINTS TO PLOC - 1 OF EXIT CODE
FAB2  71          1602          MOV     M,C         ; STORE LSB OF START ADDRESS
                               1603  RED4:          ; FINISH PROCESSING EOF RECORD
FAB3  C1          1604          POP     B           ; RESTORE B,C
FAB4  CDDDBFD     1605          CALL    BYTE        ; GET RECORD TYPE AND IGNORE IT
FAB7  CDDDBFD     1606          CALL    BYTE        ; GET CHECKSUM
FABA  C247F8     1607          JNZ    ERROR      ; JUMP IF CHECKSUM ERROR
FABD  E1          1608          POP     H           ; CUT BACK STACK POINTER
FABE  C9          1609          RET
                               1610  ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*

```

```

LOC  OBJ          LINE      SOURCE STATEMENT
                                     1611 ;
                                     1612 ; 'S' COMMAND - SUBSTITUTE MEMORY
                                     1613 ;
                                     1614 ; THIS ROUTINE EXPECTS ONE PARAMETER FROM THE LOCAL CONSOLE, FOLLOWED
                                     1615 ; BY A SPACE. THE PARAMETER IS INTERPRETED AS A MEMORY LOCATION
                                     1616 ; AND THE ROUTINE WILL DISPLAY THE CONTENTS OF THAT LOCATION,
                                     1617 ; FOLLOWED BY A DASH (-). TO MODIFY MEMORY, TYPE IN THE NEW DATA
                                     1618 ; FOLLOWED BY A SPACE OR A CARRIAGE RETURN. IF NO MODIFICATION
                                     1619 ; OF THE LOCATION IS REQUIRED, TYPE ONLY A SPACE OR CARRIAGE RETURN.
                                     1620 ; IF A SPACE WAS LAST TYPED, THE NEXT MEMORY LOCATION WILL BE DISPLAYED
                                     1621 ; AND MODIFICATION OF IT IS ALLOWED. IF A CARRIAGE RETURN WAS ENTERED,
                                     1622 ; THE COMMAND IS TERMINATED.
                                     1623 ;
                                     1624 SUBS:
FABF  CD74FE      1625      CALL    PARAM          ; GET MEMORY ADDRESS
FAC2  D8          1626      RC              ; ONLY CR ENTERED SO RETURN TO MAIN COMMAND LOOP
                                     1627 SU0:
FAC3  7E          1628      MOV     A,M           ; HL HAS REQUESTED MEMORY ADDRESS
FAC4  CD5BFE      1629      CALL    LBYTE        ; DISPLAY CONTENTS OF THAT ADDRESS
FAC7  CDDEFC      1630      CALL    COMC         ; OUTPUT PROMPT CHARACTER
FACA  2D          1631      DB      '-'
FACB  CDC5FE      1632      CALL    PCHK
FACE  D8          1633      RC              ; CR ENTERED, RETURN TO COMMAND MODE
FACF  CAD9FA      1634      JZ     SU1           ; SPACE ENTERED, SPACE BY
FAD2  EB          1635      XCHG                    ; SAVE MEMORY ADDRESS
FAD3  CD7AFE      1636      CALL    PA0          ; GET NEW VALUE
FAD6  EB          1637      XCHG                    ; E = VALUE
FAD7  73          1638      MOV     M,E          ; STORE NEW VALUE
FAD8  D8          1639      RC              ; CR ENTERED AFTER VALUE, RETURN
                                     1640 SU1:
FAD9  23          1641      INX     H           ; HL POINTS TO NEXT MEMORY LOCATION
FADA  C3C3FA      1642      JMP     SU0
1643 ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
1644 ;
1645 ; 'W' COMMAND - WRITE HEXADECIMAL FILE
1646 ;
1647 ; THIS ROUTINE EXPECTS TWO HEXADECIMAL PARAMETERS WHICH ARE
1648 ; INTERPRETED AS THE BOUNDS OF A MEMORY AREA TO BE ENCODED
1649 ; INTO HEXADECIMAL FORMAT AND PUNCHED ON THE ASSIGNED PUNCH
1650 ; DEVICE.
1651 WRITE:
FADD  CD39FE      1652      CALL    EXPR          ; GET ADDRESS RANGE
FAE0  CDFEFD      1653      CALL    CRLF         ; NEW LINE
FAE3  D1          1654      POP     D           ; DE := HIGH ADDRESS
FAE4  E1          1655      POP     H           ; HL := LOW ADDRESS
                                     1656 WR0:
FAE5  CDE5FC      1657      CALL    POC          ; EMIT RECORD MARK
FAE8  3A          1658      DB      ':'
FAE9  011000      1659      LXI    B,16        ; INITIALIZE B := 0, C := AH (DECIMAL 16)
1660 ;-----
FAEC  E5          1661      PUSH   H           ; SAVE HL
                                     1662 WR1:
FAED  04          1663      INR     B           ; INCREMENT RECORD LENGTH
FAEE  0D          1664      DCR     C
FAEF  CAF8FA      1665      JZ     WR2          ; TERMINATE ON COUNT OF 16 BYTES

```

LOC	OBJ	LINE	SOURCE STATEMENT
FAF2	CD4CFE	1666	CALL HILO ; OR END OF RANGE
FAF5	D2EDFA	1667	JNC WR1 ; WHICHEVER OCCURS FIRST.
		1668	-----
		1669	WR2: ; OUTPUT A DATA RECORD
FAF8	E1	1670	POP H ; RESTORE HL := LOW ADDRESS
FAF9	D5	1671	PUSH D ; SAVE HIGH ADDRESS
FAFA	1600	1672	MVI D,0 ; INITIALIZE CHECKSUM D := 0
FAFC	78	1673	MOV A,B ; A := RECORD LENGTH
FAFD	CDAFFE	1674	CALL PBYTE ; EMIT RECORD LENGTH
FB00	CDAAFE	1675	CALL PADR ; EMIT HL := LOW ADDRESS
FB03	AF	1676	XRA A
FB04	CDAFFE	1677	CALL PBYTE ; EMIT RECORD TYPE = 1
		1678	-----
		1679	WR3:
FB07	7E	1680	MOV A,M ; FETCH DATA
FB08	CDAFFE	1681	CALL PBYTE ; EMIT IT
FB0B	23	1682	INX H ; INCREMENT MEMORY ADDRESS
FB0C	05	1683	DCR B ; DECREMENT COUNT
FB0D	C207FB	1684	JNZ WR3 ; LOOP UNTIL ENTIRE RECORD HAS BEEN OUTPUT
FB10	AF	1685	XRA A
FB11	92	1686	SUB D ; D CONTAINS RUNNING CHECKSUM
FB12	CDAFFE	1687	CALL PBYTE ; EMIT CHECKSUM := -D
FB15	D1	1688	POP D ; RESTORE DE := HIGH ADDRESS
FB16	2B	1689	DCX H ; BACKUP MEMORY POINTER
		1690	; NOW PUNCH CR,LF --- IGNORED BY THE 'R'
		1691	; COMMAND BUT HANDY IF LISTING PUNCHED
		1692	; TAPE ON THE TTY
FB17	CDE5FC	1693	CALL POC ; PUNCH CARRIAGE RETURN
FB1A	0D	1694	DB CR
FB1B	CDE5FC	1695	CALL POC ; PUNCH LINE FEED CHARACTER
FB1E	0A	1696	DB LF
FB1F	CD4CFE	1697	CALL HILO ; TEST FOR TERMINATION
FB22	D2E5FA	1698	JNC WR0 ; IF NOT DONE, FORM NEXT RECORD AND OUTPUT IT
FB25	C9	1699	RET
		1700	*-----*
		1701	;
		1702	; 'X' COMMAND - EXAMINE AND MODIFY CPU REGISTERS
		1703	;
		1704	; THIS ROUTINE ALLOWS THE OPERATOR TO EXAMINE AND/OR MODIFY
		1705	; THE CONTENTS OF THE USER PROGRAM'S REGISTERS. THE REGISTER
		1706	; VALUES WERE STORED AS A RESULT OF A PREVIOUS BREAKPOINT AND
		1707	; WILL BE RESTORED TO THE USER PROGRAM DURING A SUBSEQUENT 'G'
		1708	; COMMAND.
		1709	X:
FB26	2181FB	1710	LXI H,ACTBL ; POINT TO ACCESS TABLE
FB29	CD5FE	1711	CALL PCHK ; GET REGISTER IDENTIFIER
FB2C	DA6AFB	1712	JC X5 ; IF CARRY = 1, CR ENTERED
FB2F	0E0C	1713	MVI C,NREGS
		1714	X0:
FB31	BE	1715	CMP M
FB32	CA3FFB	1716	JZ X1 ; MATCHED REGISTER IDENTIFIER
FB35	23	1717	INX H ; POINT TO NEXT TABLE ENTRY
FB36	23	1718	INX H
FB37	23	1719	INX H
FB38	0D	1720	DCR C ; DECREMENT REGISTER COUNTER

LOC	OBJ	LINE	SOURCE STATEMENT
FB39	C231FB	1721	JNZ X0 ; 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	1730	RC ; CR ENTERED, RETURN TO COMMAND MODE
FB4D	CA60FB	1731	JZ X4
FB50	E5	1732	PUSH H ; SAVE POINTER TO ACTBL
FB51	C5	1733	PUSH B ; SAVE PRECISION
FB52	CD7AFE	1734	CALL PA0 ; 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	1740	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:
FB60	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	FE0D	1750	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
FB70	AF	1758	XRA A ; CLEAR A
FB71	B6	1759	ORA M ; SET CONDITION CODES
FB72	F8	1760	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:
		1770	; (1) REGISTER IDENTIFIER
		1771	; (2) LOCATION ON STORAGE PAGE
		1772	; (3) PRECISION
		1773	;
		1774	ACTBL:
FB81	41	1775	DB 'A', ALOC AND HMSK, 0

LOC	OBJ	LINE	SOURCE STATEMENT
FB82	CF		
FB83	00		
FB84	42	1776	DB     'B',     BLOC AND HMSK, 0
FB85	CB		
FB86	00		
FB87	43	1777	DB     'C',     CLOC AND HMSK, 0
FB88	CA		
FB89	00		
FB8A	44	1778	DB     'D',     DLOC AND HMSK, 0
FB8B	C9		
FB8C	00		
FB8D	45	1779	DB     'E',     ELOC AND HMSK, 0
FB8E	C8		
FB8F	00		
FB90	46	1780	DB     'F',     FLOC AND HMSK, 0
FB91	CE		
FB92	00		
FB93	48	1781	DB     'H',     HLOC AND HMSK, 0
FB94	DD		
FB95	00		
FB96	49	1782	DB     'I',     ILOC AND HMSK, 0
FB97	CD		
FB98	00		
FB99	4C	1783	DB     'L',     LLOC AND HMSK, 0
FB9A	DC		
FB9B	00		
FB9C	4D	1784	DB     'M',     HLOC AND HMSK, 1
FB9D	DD		
FB9E	01		
FB9F	50	1785	DB     'P',     PLOC AND HMSK, 1
FBA0	E1		
FBA1	01		
FBA2	53	1786	DB     'S',     SLOC AND HMSK, 1
FBA3	D1		
FBA4	01		
FBA5	FF	1787	DB     -1
000C		1788	NREGS   EQU     (\$-ACTBL)/3     ; LENGTH OF ACCESS TABLE
		1789	;*****
		1790	;
		1791	; 'Z' COMMAND - TRANSFER CONTROL TO DIAGNOSTIC PROGRAM IN PROM
		1792	; THIS ROUTINE EXPECTS A '\$' AT WHICH POINT IT WILL CALL THE DIAGNOSTIC PROGRAM.
		1793	Z:
FBA6	CD61FF	1794	CALL    TI                     ; GET A CHARACTER FROM THE CONSOLE
FBA9	FE24	1795	CPI     '\$'                    ; IS IT A '\$'?
FBAB	C247F8	1796	JNZ     ERROR                 ; ERROR IF IT ISN'T
FBAE	CD61FF	1797	CALL    TI                     ; GET A CHARACTER FROM THE CONSOLE
FBB1	FE0D	1798	CPI     CR                     ; EXPECT A CARRIAGE RETURN
FBB3	C247F8	1799	JNZ     ERROR                 ; ERROR IF IT ISN'T
FBB6	3E0C	1800	MVI     A,BTDGON               ; TURN ON THE BOOT/DIAGNOSTIC PROM
FBB8	D3FF	1801	OUT     CPUC
FBBA	CD00EB	1802	CALL    DIAGMN                ; CALL THE DIAGNOSTIC PROGRAM
FBBD	C9	1803	RET                            ; RETURN TO MAIN COMMAND LOOP
		1804	*****
		1805	;
		1806	;*    END OF MONITOR COMMANDS, BEGINNING OF I/O ROUTINES            *

```

LOC  OBJ          LINE          SOURCE STATEMENT
1807 ;*
1808 ;*****
1809 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
1810 ; 'CI' - EXTERNALLY REFERENCED ROUTINE
1811 ;            ENTERED VIA CALL FROM 'TI' ROUTINE
1812 ; PROCESS: LOCAL CONSOLE INPUT CODE
1813 ; INPUT:
1814 ; OUTPUT: CHARACTER RETURNED IN A-REG
1815 ; MODIFIED: A, FLAGS
1816 ; STACK USAGE: 2 BYTES
1817 ; EXPLANATION: BASED ON I/O STATUS BYTE (IOBYT), DECIDE IF CONSOLE INPUT
1818 ;    DEVICE IS TTY, CRT, BATCH, OR USER-DEFINED DEVICE.  IF IT IS TTY OR CRT
1819 ;    LOOP UNTIL READ, INPUT THE CHARACTER, THEN RETURN.  IF IT IS BATCH,
1820 ;    JUMP TO 'RI' ROUTINE.  IF IT IS USER-DEFINED DEVICE, JUMP TO @USER.
1821 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
1822 CI:                           ; LOCAL CONSOLE INPUT
FBBE 3A0300       1823           LDA    IOBYT           ; GET STATUS BYTE
FBC1 E603         1824           ANI    NOT CMSK         ; LOOK AT ONLY CONSOLE FIELD
FBC3 C2D0FB       1825           JNZ    CI0             ; JUMP IF CONSOLE IS NOT TTY
1826 ;-----
1827 ; CONSOLE = TTY
1828 TTYIN:
FBC6 DBF5         1829           IN     TTYS             ; TTY STATUS PORT
FBC8 E602         1830           ANI    RRDY             ; CHECK FOR RECEIVE BUFFER READY
FBCA CAC6FB       1831           JZ     TTYIN            ; LOOP UNTIL IT IS READY
FBCD DBF4         1832           IN     TTYI             ; INPUT CHARACTER FROM TTY
FBCF C9           1833           RET                   ; RETURN; CHARACTER IN A-REG
1834 ;-----
1835 ; CONSOLE = CRT, BATCH, OR USER-DEFINED
1836 CI0:
FBD0 FE01         1837           CPI    CCRT             ; LOCAL CONSOLE = CRT?
FBD2 C2FDFB       1838           JNZ    CI4             ; JUMP IF CONSOLE IS NOT CRT
FBD5 E5           1839           PUSH   H               ; SAVE HL
FBD6 2A0400       1840           LHLD  MEMTOP
FBD9 2ECC         1841           MVI   L,ILOC-1 AND 0FFH; HL NOW POINTS TO CONFIGURATION BYTE STORED
1842                               ;    IN EXIT TEMPLATE IN TOP PAGE OF RAM
FBD8 7E           1843           MOV    A,M             ; A := CONFIGURATION BYTE
FDBC E1           1844           POP    H               ; RESTORE HL
FBDD 0F           1845           RRC                   ; ROTATE BIT 0 INTO CARRY BIT, THUS CARRY = 1
1846                               ;    MEANS RUNNING ON SYSTEM WITHOUT INTEGRATED
1847                               ;    CRT
FBDE D2EBFB       1848           JNC    CI2             ; JUMP IF INTEGRATED CRT IS PRESENT
1849 ;-----
1850 ; CONSOLE = SERIAL CRT
1851 CI1:
FBE1 DBF7         1852           IN     USCS             ; INPUT CRT STATUS
FBE3 E602         1853           ANI    RRDY             ; CHECK FOR RECEIVER BUFFER READY
FBE5 CAE1FB       1854           JZ     CI1             ; LOOP UNTIL IT IS READY
FBE8 DBF6         1855           IN     USCI            ; GET CHARACTER FROM THE CRT
FBEA C9           1856           RET                   ; RETURN; CHARACTER IS IN A-REG
1857 ;-----
1858 ; CONSOLE = INTEGRATED CRT
1859 CI2:
FBEB C5           1860           PUSH   B               ; SAVE B,C
1861 CI3:

```

LOC	OBJ	LINE	SOURCE STATEMENT
FBEC	0613	1862	MVI    B,KSTS                   ; LOAD KEYBOARD STATUS COMMAND
FBEE	CD7FFF	1863	CALL   IOCDRI                   ; INPUT KEYBOARD STATUS FROM IOC
FBF1	E601	1864	ANI    KRDY                    ; IS THE KEYBOARD READY?
FBF3	CAECFB	1865	JZ     CI3                     ; LOOP UNTIL IT IS
FBF6	0612	1866	MVI    B,KEYC                  ; LOAD INPUT DATA COMMAND
FBF8	CD7FFF	1867	CALL   IOCDRI                   ; INPUT DATA FROM THE KEYBOARD
FBFB	C1	1868	POP    B                       ; RESTORE B,C
FBFC	C9	1869	RET                             ; RETURN; CHARACTER IS IN A-REG
		1870	-----
		1871	; CONSOLE IS BATCH OR USER-DEFINED DEVICE
		1872	CI4:
FBFD	FE02	1873	CPI    BATCH                   ; BATCH MODE, INPUT = READER
FBFF	CA0FFC	1874	JZ     RI                      ; BATCH MODE, INPUT = READER
FC02	3EE8	1875	MVI    A,CILOC AND HMSK; USER DEFINE LOCAL CONSOLE INPUT
FC04	C38CFC	1876	JMP    @USER
		1877	;;;
		1878	; 'BREAK' - ENTERED VIA CALLS FROM 'BLK','COM','LOM' ROUTINES
		1879	; PROCESS: TEST FOR OPERATOR INTERRUPTION OF COMMAND (I.E. DID OPERATOR
		1880	;                   DEPRESS THE "BREAK" KEY)
		1881	; INPUT:
		1882	; OUTPUT:
		1883	; MODIFIED: A,FLAGS
		1884	; STACK USAGE: 4 BYTES
		1885	BREAK:
FC07	CD44FD	1886	CALL   CSTS                    ; SEE IF A KEY WAS DEPRESSED
FC0A	B7	1887	ORA    A
FC0B	C8	1888	RZ                             ; NO CHARACTER READY
FC0C	C361FF	1889	JMP    TI                      ; GET THE CHARACTER
		1890	;;;
		1891	; 'RI' - EXTERNALLY REFERENCED ROUTINE
		1892	;                   ENTERED VIA CALLS FROM 'CI','RIX' ROUTINES
		1893	; PROCESS: READER INPUT CODE
		1894	; INPUT:
		1895	; OUTPUT: CARRY = 0 AND VALID CHARACTER IN A-REG, OTHERWISE
		1896	;                   CARRY = 1 AND INVALID DATA (ZEROES) IN A-REG
		1897	; MODIFIED: A, FLAGS
		1898	; STACK USAGE: 8 BYTES
		1899	;;;
		1900	RI:
FC0F	E5	1901	PUSH   H                       ; SAVE HL
FC10	3A0300	1902	LDA    IOBYT                   ; GET STATUS BYTE
FC13	E60C	1903	ANI    NOT RMSK                ; GET READER BITS
FC15	C258FC	1904	JNZ    RI5                     ; JUMP IF READER IS NOT THE TTY
		1905	-----
		1906	; READER = TTY
FC18	C5	1907	PUSH   B                       ; SAVE BC
FC19	3E0D	1908	MVI    A,DISABL               ; HOLD UP INTERRUPTS WHILE TAPE IS ADVANCING
FC1B	D3FF	1909	OUT    CPUC
FC1D	DBF4	1910	IN     TTYI                    ; CLEAR RECEIVE BUFFER BY READING IN ANY
		1911	;                   DATA THAT MAY BE THERE
		1912	RI0:
FC1F	DBF5	1913	IN     TTYS                    ; READ IN USART STATUS
FC21	E604	1914	ANI    TXBE                   ; CHECK FOR TRANSMITTER BUFFER EMPTY
FC23	CA1FFC	1915	JZ     RI0                     ; TRY AGAIN IF NOT EMPTY
FC26	3E27	1916	MVI    A,TADV                  ; ADVANCE THE TAPE



LOC	OBJ	LINE	SOURCE STATEMENT
FC28	D3F5	1917	OUT    TTYC                   ; OUTPUT THE ADVANCE COMMAND
FC2A	0628	1918	MVI    B,RADCT               ; INITIALIZE TIMER FOR 45 MS.
		1919	RI1:
FC2C	CD1EFE	1920	CALL   DELAY                 ; DELAY FOR 1 MILLISECONDS
FC2F	05	1921	DCR    B                     ; DECREMENT TIMER
FC30	C22CFC	1922	JNZ    RI1                   ; JUMP IF TIMER NOT EXPIRED
FC33	3E25	1923	MVI    A,COMD               ; STOP THE READER ADVANCE
FC35	D3F5	1924	OUT    TTYC                   ; OUTPUT STOP COMMAND
FC37	06FA	1925	MVI    B,RTOCT               ; INITIALIZE TIMER FOR 250 MS.
		1926	RI2:
FC39	DBF5	1927	IN     TTYS                  ; INPUT READER STATUS
FC3B	E602	1928	ANI    RRDY                 ; CHECK FOR RECEIVER BUFFER READY
FC3D	C24CFC	1929	JNZ    RI4                   ; YES - DATA IS READY
FC40	CD1EFE	1930	CALL   DELAY                 ; DELAY 1 MS
FC43	05	1931	DCR    B                     ; DECREMENT TIMER
FC44	C239FC	1932	JNZ    RI2                   ; JUMP IF TIMER NOT EXPIRED
		1933	RI3:
FC47	AF	1934	XRA    A                     ; ZERO A, RESET CARRY
FC48	37	1935	STC                         ; SET CARRY INDICATING EOF
FC49	C34FFC	1936	JMP    RI4B
		1937	RI4:
FC4C	DBF4	1938	IN     TTYI                  ; CLEAR CARRY
FC4E	B7	1939	ORA    A
		1940	RI4B:
FC4F	F5	1941	PUSH   PSW                  ; SAVE DATA
FC50	3E05	1942	MVI    A,ENABL               ; PERMIT INTERRUPTS TO GO THROUGH
FC52	D3FF	1943	OUT    CPUC
FC54	F1	1944	POP    PSW
FC55	C1	1945	POP    B                     ; RESTORE BC
FC56	E1	1946	POP    H
FC57	C9	1947	RET                         ; RETURN
		1948	;-----
		1949	; READER IS PTR, USER-DEV-1, OR USER-DEV-2
		1950	RI5:
FC58	FE04	1951	CPI    RPTR                  ; IS READER THE PAPER TAPE READER?
FC5A	C282FC	1952	JNZ    RI8                   ; JUMP IF IT ISN'T
		1953	;-----
		1954	; READER = PAPER TAPE READER
FC5D	C5	1955	PUSH   B                     ; SAVE BC
FC5E	0650	1956	MVI    B,RDRC OR PTRADV; LOAD READER ADVANCE 1 FRAME COMMAND
FC60	CDE4FF	1957	CALL   PIOCOM               ; OUTPUT THE COMMAND
FC63	26FA	1958	MVI    H,TOUT               ; 250 MS. TIMEOUT COUNTER
		1959	RI6:
FC65	0611	1960	MVI    B,RSTC               ; LOAD READER STATUS COMMAND
FC67	CDB5FF	1961	CALL   PIODR1               ; READ STATUS
FC6A	E601	1962	ANI    PTRDY                 ; IS THE READER READY?
FC6C	C279FC	1963	JNZ    RI7                   ; JUMP IF IT IS
FC6F	CD1EFE	1964	CALL   DELAY                 ; STALL FOR 1 MS.
FC72	25	1965	DCR    H                     ; 250 MS. TIMEOUT LOOP
FC73	C265FC	1966	JNZ    RI6
FC76	C347FC	1967	JMP    RI3                   ; 250 MS. ARE UP; RETURN WITH CARRY = 1 (EOF COND)
		1968	RI7:
FC79	0610	1969	MVI    B,RDRC               ; LOAD READER COMMAND
FC7B	CDB5FF	1970	CALL   PIODR1               ; READ A CHARACTER FROM THE PAPER TAPE READER
FC7E	B7	1971	ORA    A                     ; RESET CARRY BIT

```

LOC  OBJ          LINE          SOURCE STATEMENT
FC7F  C1          1972          POP      B           ; RESTORE BC
FC80  E1          1973          POP      H
FC81  C9          1974          RET              ; RETURN SUCCESSFULLY WITH CARRY = 0
1975  ;-----
1976  ; READER IS USER-DEFINED DEVICE 1 OR DEVICE 2
1977  RI8:
FC82  E1          1978          POP      H
FC83  FE08        1979          CPI      RUSE1
FC85  3EEE        1980          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)
1990  ; OUTPUT:
1991  ; MODIFIED:
1992  ; STACK USAGE:
1993  @USER:
FC8C  E5          1994          PUSH   H           ; SAVE HL, CREATE A STACK ENTRY
FC8D  2A0400      1995          LHLD   MEMTOP
FC90  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
2000  ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
2001  ; 'CO' - EXTERNALLY REFERENCED ROUTINE
2002  ;      ENTERED VIA CALL FROM 'TI' ROUTINE
2003  ; 'BLK' - ENTERED VIA CALLS FROM 'H', 'X' COMMANDS
2004  ; 'COM' - ENTERED VIA CALLS FROM 'Q', 'X' COMMANDS
2005  ;      ENTERED VIA JUMPS FROM 'COMC', 'HXD' ROUTINES
2006  ; 'TTYOUT' - ENTERED VIA JUMPS FROM 'LOM','LO','POC','PO' ROUTINES
2007  ; 'CRTOUT' - ENTERED VIA JUMPS FROM 'LOM','LO' ROUTINES
2008  ;      ENTERED VIA CALL FROM BOOTSTRAP PROGRAM
2009  ; PROCESS: LOCAL CONSOLE OUTPUT CODE
2010  ; INPUT: VALUE IN C-REG
2011  ; OUTPUT: DATA OUTPUT TO APPROPRIATE DEVICE
2012  ; MODIFIED: A, FLAGS, C
2013  ; STACK USAGE: 2 BYTES
2014  ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
2015  BLK:
FC93  0E20        2016          MVI     C,' '      ; PRINT A BLANK
2017  COM:
FC95  3A0300      2018          LDA     IOBYT       ; LOCAL CONSOLE OUTPUT
FC98  E603        2019          ANI    NOT CMSK    ; GET STATUS BYTE
FC9A  FE02        2020          CPI    BATCH       ; LOOK ONLY AT CONSOLE FIELD
FC9C  C407FC      2021          CNZ    BREAK      ; IS CONSOLE = BATCH?
2022  ; IF SO, DO NOT HONOR BREAK KEY IN BATCH.MODE
2023  ; IF IT ISN'T, THEN TEST FOR BREAK KEY
2024  CO:
FC9F  3A0300      2024          LDA     IOBYT       ; EXTERNAL ENTRY POINT
FCA2  E603        2025          ANI    NOT CMSK    ; GET STATUS BYTE
FCA4  C2B2FC      2026          JNZ    CO0         ; LOOK ONLY AT CONSOLE FIELD
                ; JUMP IF CONSOLE IS NOT TTY

```

```

LOC  OBJ          LINE      SOURCE STATEMENT
                                2027 ;-----
                                2028 ; CONSOLE = TTY
                                2029 TTYOUT:
FCA7 DBF5          2030      IN      TTYS          ; LOCAL CONSOLE = TTY; GET TTY STATUS
FCA9 E601          2031      ANI      TRDY           ; IS IT READY?
FCAB CAA7FC        2032      JZ       TTYOUT        ; LOOP UNTIL IT IS
FCAE 79            2033      MOV      A,C           ; LOAD CHARACTER TO BE OUTPUT
FCAF D3F4          2034      OUT     TTYO           ; OUTPUT CHARACTER
FCB1 C9            2035      RET      ; RETURN
                                2036 ;-----
                                2037 ; CONSOLE IS CRT, BATCH, OR USER-DEFINED
                                2038 CO0:
FCB2 FE02          2039      CPI      BATCH          ; CONSOLE = BATCH?
FCB4 CALEFD        2040      JZ       LO              ; JUMP TO LIST OUTPUT IF IT IS
FCB7 FE01          2041      CPI      CCRT           ; LOCAL CONSOLE = CRT?
FCB9 3EEB          2042      MVI     A,COLC AND 0FFH ;
FCBB C28CFC        2043      JNZ     @USER           ; JUMP IF IT ISN'T, I.E. CONSOLE IS
                                2044 ; USER DEFINED LOCAL CONSOLE OUTPUT
                                2045 ;-----
                                2046 ; CONSOLE = CRT
                                2047 CRTOUT:
FCBE E5            2048      PUSH    H              ; SAVE H,L
FCBF 2A0400        2049      LHLD   MEMTOP
FCC2 2ECC          2050      MVI     L,ILOC-1 AND 0FFH; HL NOW POINTS TO CONFIGURATION BYTE IN EXIT TEMPLATE
FCC4 7E            2051      MOV     A,M             ; A NOW CONTAINS THIS CONFIGURATION BYTE
FCC5 E1            2052      POP     H              ; RESTORE H,L
FCC6 0F            2053      RRC           ; ROTATE BIT 0 INTO CARRY BIT; THUS CARRY
                                2054 ; = 1 IF INTEGRATED CRT NOT PRESENT
FCF7 D2D5FC        2055      JNC     CRTOT2          ; JUMP IF INTEGRATED CRT
                                2056 ;-----
                                2057 ; CONSOLE = SERIAL CRT
                                2058 CRTOT1:
FCCA DBF7          2059      IN      USCS           ; INTELLEC WITH SERIALLY CONNECTED CRT
FCCC E601          2060      ANI     TRDY           ; INPUT CRT STATUS
FCCE CACAFC        2061      JZ      CRTOT1         ; IS IT READY?
FCD1 79            2062      MOV     A,C           ; LOOP UNTIL IT IS
FCD2 D3F6          2063      OUT    USCO           ; MOVE CHARACTER TO BE OUTPUT TO C-REG
FCD4 C9            2064      RET      ; OUTPUT IT TO THE CRT
                                2065 ;-----
                                2066 ; CONSOLE = INTEGRATED CRT
                                2067 CRTOT2:
FCD5 79            2068      MOV     A,C           ; INTELLEC WITH INTEGRATED CRT
FCD6 C5            2069      PUSH   B              ; MOVE CHARACTER TO BE OUTPUT TO A-REG
                                2070 ; SAVE B,C
                                2071 ; CRT IS ALWAYS READY AND PRESENT - NO NEED
                                2072 ; TO CHECK ITS STATUS
FCD7 0610          2072      MVI     B,CRTC         ; LOAD OUTPUT TO CRT COMMAND
FCD9 CD94FF        2073      CALL   IOCDR2         ; OUTPUT DATA TO CRT
FCDC C1            2074      POP     B              ; RESTORE B,C
FCDD C9            2075      RET
                                2076 ;-----
                                2077 ; 'COMC' - ENTERED VIA CALLS FROM 'G','Q','S','X' COMMANDS AND 'ERROR',
                                2078 ; 'START','CRLF','RESTART' ROUTINES
                                2079 ; PROCESS: LOCAL CONSOLE OUTPUT OF CONSTANT DATA
                                2080 ; INPUT: SP
                                2081 ; OUTPUT: CONTENTS OF ADDRESS POINTED TO BY SP IS A RETURN ADDRESS TWO GREATER

```

```

LOC  OBJ          LINE      SOURCE STATEMENT
                                2082 ;           THAN THAT OF THE CALL COMC INSTRUCTION
                                2083 ; MODIFIED: C,H,L
                                2084 ; STACK USAGE: 2 BYTES
                                2085 COMC:
FCDE  E3          2086      XTHL                ; SINCE COMC WAS CALLED, SP NOW POINTS TO A STACK
                                2087 ;           ENTRY CONTAINING THE ADDRESS OF THE NEXT
                                2088 ;           INSTRUCTION, WHICH IN THIS CASE IS A DB.
                                2089 ;           HL NOW POINTS TO THIS DB.
FCDF  4E          2090      MOV      C,M          ; C NOW CONTAINS THE CHARACTER TO BE OUTPUT
FCE0  23          2091      INX      H            ; BUMP RETURN ADDRESS, I.E. POINT IT BEYOND THE DB.
FCE1  E3          2092      XTHL                ; SP MODIFIED, HL IS AS IT WAS ORIGINALLY
FCE2  C395FC     2093      JMP      COM          ; OUTPUT IT
                                2094 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                                2095 ; 'PO' - EXTERNALLY REFERENCED ROUTINE
                                2096 ;           ENTERED VIA CALL FROM 'PBYTE' ROUTINE
                                2097 ; 'POC' - ENTERED VIA CALLS FROM 'E','N','W' COMMANDS AND 'LEAD','PEOL'
                                2098 ;           ROUTINES
                                2099 ; PROCESS: PUNCH OUTPUT CODE
                                2100 ; INPUT: VALUE IN C-REG
                                2101 ; OUTPUT:
                                2102 ; MODIFIED: A, FLAGS, C
                                2103 ; STACK USAGE: 2 BYTES
                                2104 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                                2105 POC:
                                2106 ;           PUNCH A CONSTANT
                                2107 ;           SINCE POC ENTERED VIA CALL, SP POINTS TO STACK
                                2108 ;           ENTRY CONTAINING ADDRESS OF NEXT INSTRUCTION
                                2109 ;           WHICH IS A DB. HL NOW POINTS TO THIS DB.
FCE5  E3          2106      XTHL                ; SINCE POC ENTERED VIA CALL, SP POINTS TO STACK
                                2107 ;           ENTRY CONTAINING ADDRESS OF NEXT INSTRUCTION
                                2108 ;           WHICH IS A DB. HL NOW POINTS TO THIS DB.
FCE6  4E          2109      MOV      C,M          ; C NOW CONTAINS CHARACTER TO BE PUNCHED
FCE7  23          2110      INX      H            ; BUMP RETURN ADDRESS, I.E. POINT IT BEYOND DB
FCE8  E3          2111      XTHL                ; SP MODIFIED, HL IS AS IT WAS ORIGINALLY
                                2112 PO:
                                2113 ;           PUNCH OUTPUT
FCE9  3A0300     2113      LDA      IOBYT        ; GET STATUS BYTE
FCEC  E630       2114      ANI      NOT PMSK      ; GET PUNCH BITS
FCEE  CAA7FC     2115      JZ       TTYOUT        ; JUMP IF PUNCH ISN'T TTY
FCF1  FE10       2116      CPI      PPTP         ; IS PUNCH = PAPER TAPE PUNCH?
FCF3  C208FD     2117      JNZ      PO1          ; JUMP IF IT ISN'T
                                2118 ;-----
                                2119 ; PUNCH = PAPER TAPE PUNCH
FCF6  C5         2120      PUSH     B            ; SAVE BC
                                2121 PO0:
                                2122 ;           PUNCH = PTP
                                2123 ;           LOAD PUNCH STATUS COMMAND
FCF7  0613       2122      MVI      B,PSTC      ; LOAD PUNCH STATUS COMMAND
FCF9  CDB5FF     2123      CALL    PIODR1       ; READ STATUS
FCFC  E601       2124      ANI      PTPRY       ; IS THE PUNCH READY?
FCFE  CAF7FC     2125      JZ       PO0         ; LOOP UNTIL READY
FD01  0612       2126      MVI      B,PUNC      ; LOAD PUNCH OUTPUT COMMAND
FD03  CDCEFF     2127      CALL    PIODR3       ; OUTPUT CHARACTER THAT WAS IN C-REG
FD06  C1         2128      POP     B            ; RESTORE BC
FD07  C9         2129      RET
                                2130 ;-----
                                2131 ; PUNCH IS USER-DEFINED DEVICE 1 OR DEVICE 2
                                2132 PO1:
FD08  FE20       2133      CPI      PUSE1
FD0A  3EF4       2134      MVI      A,P1LOC AND 0FFH
FD0C  CA8CFC     2135      JZ       @USER       ; PUNCH = USER DEFINED PUNCH 1
FD0F  3EF7       2136      MVI      A,P2LOC AND 0FFH

```

```

LOC OBJ          LINE          SOURCE STATEMENT
FD11 C38CFC      2137          JMP      @USER          ; PUNCH = USER DEFINED PUNCH 2
                2138 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                2139 ; 'LO' - EXTERNALLY REFERENCED ROUTINE ;
                2140 ; ENTERED VIA JUMPS FROM 'COM','CO','BLK' ROUTINES ;
                2141 ; 'LOM' - ENTERED VIA CALLS FROM 'D' COMMAND AND 'DBYTE','LCRLF' ROUTINES ;
                2142 ; ENTERED VIA JUMPS FROM 'DBYTE','LCRLF' ROUTINES ;
                2143 ; PROCESS: LIST OUTPUT ;
                2144 ; INPUT: VALUE IN C-REG ;
                2145 ; OUTPUT: ;
                2146 ; MODIFIED: A, FLAGS, C ;
                2147 ; STACK USAGE: 2 BYTES ;
                2148 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                2149 LOM: ; LIST OUTPUT ON CONSOLE
FD14 3A0300      2150          LDA      IOBYT
FD17 E603        2151          ANI      NOT CMSK      ; LOOK ONLY AT CONSOLE FIELD OF IOBYT
FD19 FE02        2152          CPI      BATCH        ; IS CONSOLE ASSIGNED TO BATCH MODE?
FD1B C407FC      2153          CNZ      BREAK      ; IF IT ISN'T, WE SHOULD TEST FOR BREAK KEY
                2154 ; I.E. IN BATCH MODE THE BREAK KEY IS NOT
                2155 ; HONORED
                2156 LO: ; LIST OUTPUT
                2157          LDA      IOBYT      ; GET STATUS BYTE
FD21 E6C0        2158          ANI      NOT LMSK      ; LOOK AT LIST FIELD
FD23 CAA7FC      2159          JZ      TTYOUT        ; JUMP IF LIST = TTY
FD26 FE40        2160          CPI      LCRT
FD28 CABEFC      2161          JZ      CRTOUT        ; JUMP IF LIST = CRT
FD2B FEC0        2162          CPI      LUSE        ; TEST FOR USER DEFINED LIST DEVICE
FD2D 3EFA        2163          MVI      A,L1LOC AND 0FFH; A := LSB OF L1LOC ADDRESS
FD2F CA8CFC      2164          JZ      @USER        ; JUMP IF LIST = USER-DEFINED DEVICE
                2165 ;-----
                2166 ; LIST = LPT
FD32 C5          2167          PUSH     B            ; SAVE BC
                2168 LP0:
FD33 0615        2169          MVI      B,LSTC        ; LOAD LINE PRINTER STATUS COMMAND
FD35 CDB5FF      2170          CALL     PIODR1        ; READ STATUS
FD38 E601        2171          ANI      LPTRY        ; IS IT READY?
FD3A CA33FD      2172          JZ      LP0           ; LOOP UNTIL IT IS
FD3D 0614        2173          MVI      B,LPTC        ; LOAD LINE PRINTER PRINT COMMAND
FD3F CDCEFF      2174          CALL     PIODR3        ; OUTPUT CHARACTER CONTAINED IN C-REG
FD42 C1          2175          POP      B            ; RESTORE BC
FD43 C9          2176          RET
                2177 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                2178 ; 'CSTS' - EXTERNALLY REFERENCED ROUTINE ;
                2179 ; ENTERED VIA CALL FROM 'BREAK' ROUTINE ;
                2180 ; PROCESS: LOCAL CONSOLE INPUT STATUS ;
                2181 ; INPUT: ;
                2182 ; OUTPUT: A-REG CONTAINS 00 IF NO KEY HAS BEEN DEPRESSED, ;
                2183 ; A-REG CONTAINS FFH IF A KEY HAS BEEN DEPRESSED ;
                2184 ; MODIFIED: A, FLAGS ;
                2185 ; STACK USAGE: 2 BYTES ;
                2186 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                2187 CSTS: ; LOCAL CONSOLE INPUT STATUS
                2188          LDA      IOBYT      ; GET STATUS BYTE
FD44 3A0300      2189          ANI      NOT CMSK      ; LOOK ONLY AT CONSOLE FIELD OF IOBYT
FD47 E603        2190          JNZ     CS0           ; JUMP IF CONSOLE IS NOT TTY
                2191 ;-----

```

```

LOC  OBJ          LINE      SOURCE STATEMENT
-----
                2192 ; CONSOLE = TTY
FD4C  DBF5         2193      IN      TTYS          ; GET TTY STATUS
FD4E  E602         2194      ANI      RRDY          ; IS RECEIVE BUFFER READY? (IF TTY KEY WAS
                2195                        ; DEPRESSED, ZERO BIT WILL BE RESET)
FD50  C374FD      2196      JMP      CS2
                2197 ;-----
                2198 ; CONSOLE = CRT, BATCH, OR USER-DEFINED
                2199 CS0:
FD53  FE01         2200      CPI      CCRT          ; CONSOLE = CRT?
FD55  C279FD      2201      JNZ      CS3          ; JUMP IF CONSOLE IS NOT CRT
FD58  E5          2202      PUSH     H            ; SAVE H,L
FD59  2A0400      2203      LHL     MEMTOP
FD5C  2ECC        2204      MVI     L,ILOC-1 AND 0FFH; HL POINTS TO CONFIGURATION BYTE IN EXIT TEMPLATE
FD5E  7E          2205      MOV     A,M          ; A CONTAINS THIS CONFIGURATION BYTE
FD5F  E1          2206      POP     H            ; RESTORE H,L
FD60  0F          2207      RRC          ; ROTATE BIT 0 INTO CARRY; THUS CARRY = 1
                2208                        ; MEANS INTEGRATED CRT NOT PRESENT
FD61  D26BFD      2209      JNC     CS1          ; JUMP IF INTEGRATED CRT PRESENT
                2210 ;-----
                2211 ; CONSOLE = SERIAL CRT
FD64  DBF7         2212      IN      USCS          ; GET CRT STATUS
FD66  E602         2213      ANI      RRDY          ; IS RECEIVE BUFFER READY? (IF KEY HAS BEEN
                2214                        ; DEPRESSED, ZERO BIT WILL BE RESET)
FD68  C374FD      2215      JMP      CS2
                2216 ;-----
                2217 ; CONSOLE = INTEGRATED CRT
                2218 CS1:
FD6B  C5          2219      PUSH     B            ; INTELLEC WITH INTEGRATED CRT
FD6C  0613        2220      MVI     B,KSTS       ; SAVE B,C
FD6E  CD7FFF      2221      CALL    IOCDR1       ; LOAD CRT STATUS COMMAND
FD71  E601        2222      ANI      KRDY          ; GET CRT STATUS
                2223                        ; IS RECEIVE BUFFER READY? (IF KEY HAS BEEN
                2224                        ; DEPRESSED, ZERO BIT WILL BE RESET)
FD73  C1          2224      POP     B            ; RESTORE B,C
                2225 CS2:
                2225                        ; COMMON RETURN POINT FOR CRT,TTY
FD74  3E00        2226      MVI     A,FALSE      ; INITIALIZE A-REG TO 00
FD76  C8          2227      RZ              ; RETURN WITH A := 00 IF NO DATA AVAILABLE
FD77  2F          2228      CMA
FD78  C9          2229      RET              ; RETURN WITH A := FF IF DATA AVAILABLE
                2230 ;-----
                2231 ; CONSOLE = BATCH OR USER-DEFINED DEVICE
                2232 CS3:
FD79  FE02        2233      CPI      BATCH        ; IS IT BATCH?
FD7B  3EFF        2234      MVI     A,TRUE
FD7D  C8          2235      RZ              ; RETURN IF CONSOLE IS BATCH; A := FF
FD7E  3EFD        2236      MVI     A,CSLOC AND 0FFH; CONSOLE = USER DEFINED LOCAL CONSOLE, BRANCH
                2237                        ; TO USER'S OWN STATUS ROUTINE
FD80  C38CFC      2238      JMP     @USER
                2239 ;-----
                2240 ; 'IOCHK' - EXTERNALLY REFERENCED ROUTINE
                2241 ; PROCESS: GET I/O SYSTEM STATUS
                2242 ; INPUT:
                2243 ; OUTPUT: STATUS BYTE RETURNED IN A-REG
                2244 ; MODIFIED: A
                2245 ; STACK USAGE: 2 BYTES
                2246 ;-----

```

LOC	OBJ	LINE	SOURCE STATEMENT
		2247	IOCHK:
FD83	3A0300	2248	LDA     IOBYT                   ; GET STATUS BYTE
FD86	C9	2249	RET                             ; RETURN
		2250	;;;
		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	320300	2260	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	;;;
		2270	MEMCHK:
FD8C	3A0500	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
FD90	47	2275	MOV     B,A                      ; SO MSB GOES IN B-REG
FD91	3EC0	2276	MVI     A,USER AND 0FFH         ; LSB IN A-REG
FD93	C9	2277	RET                             ; AB POINTS TO BASE OF USER STACK IN SECOND
		2278	;                                 ; FROM TOP PAGE OF RAM
		2279	;;;
		2280	; '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	2290	PUSH    H                        ; SAVE H & L
FD95	C5	2291	PUSH    B                        ; SAVE B & C
FD96	2A0400	2292	LHLD   MEMTOP                   ; GET XTBL+1
FD99	2EE9	2293	MVI     L,XTBL+1 AND 0FFH; HL NOW POINTS TO XTBL+1 IN TOP PAGE OF RAM
FD9B	79	2294	MOV     A,C                      ; A := LOGICAL DEVICE CATEGORY
FD9C	FE08	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	0600	2300	MVI     B,0
FDA6	09	2301	DAD     B                        ; COMPUTE PROPER INDEX INTO XTBL

LOC	OBJ	LINE	SOURCE STATEMENT
FDA7	73	2302	MOV M,E ; STORE BRANCH OPERAND IN INSTRUCTION
FDA8	23	2303	INX H
FDA9	72	2304	MOV M,D ; STORE THE USER-DEFINED I/O ENTRY ROUTINE
		2305	; ADDRESS IN THE PROPER PLACE IN XTBL,
		2306	; SO IT LOOKS LIKE:
		2307	; JMP <USER-DEFINED ADDRESS>
FDA A	C1	2308	POP B ; RESTORE B & C
FDA B	E1	2309	POP H ; RESTORE H & L
FDA C	C9	2310	RET
		2311	;;;
		2312	; 'UI' - EXTERNALLY REFERENCED ROUTINE ;
		2313	; PROCESS: INPUT A CHARACTER FROM THE UPP ;
		2314	; INPUT: B CONTAINS MSB OF PROM ADDRESS ;
		2315	; C CONTAINS LSB OF PROM ADDRESS ;
		2316	; OUTPUT: DATA IN A-REG ;
		2317	; MODIFIED:A,FLAGS ;
		2318	; STACK USAGE: 6 BYTES ;
		2319	;;;
		2320	UI:
		2321	; IT IS ASSUMED THE 'UPPS' ROUTINE HAS BEEN
		2322	; CALLED AND THAT THE UPP UNIT IS READY
FDAD	C5	2323	PUSH B ; SAVE B,C
FDAE	0617	2324	MVI B,RPPC ; LOAD THE READ PROM COMMAND
		2325	; C CONTAINS PROM LOW ADDRESS
FDB0	CDCEFF	2326	CALL PIODR3 ; OUTPUT READ PROM COMMAND
		2327	; OUTPUT PROM LOW ADDRESS
FDB3	C1	2328	POP B ; RESTORE B,C; B CONTAINS PROM HIGH ADDRESS
FDB4	C5	2329	PUSH B ; SAVE B,C
FDB5	48	2330	MOV C,B ; C CONTAINS PROM HIGH ADDRESS
FDB6	CDD1FF	2331	CALL PIODR4 ; OUTPUT PROM HIGH ADDRESS
FDB9	C1	2332	POP B ; RESTORE B,C
FDBA	CDB8FF	2333	CALL PIODR2 ; INPUT PROM DATA
FDBD	C9	2334	RET
		2335	;;;
		2336	; 'UO' - EXTERNALLY REFERENCED ROUTINE ;
		2337	; PROCESS: OUTPUT A CHARACTER TO THE UPP ;
		2338	; INPUT: C CONTAINS THE CHARACTER TO BE WRITTEN INTO THE PROM ;
		2339	; D CONTAINS THE MSB OF THE PROM ADDRESS ;
		2340	; E CONTAINS THE LSB OF THE PROM ADDRESS ;
		2341	; OUTPUT: ;
		2342	; MODIFIED:A,FLAGS ;
		2343	; STACK USAGE: 8 BYTES ;
		2344	;;;
		2345	UO:
		2346	; IT IS ASSUMED THE 'UPPS' ROUTINE HAS BEEN
		2347	; CALLED AND THAT THE UPP UNIT IS READY
FDBE	C5	2348	PUSH B ; SAVE B,C
FDBF	0616	2349	MVI B,WPPC ; LOAD WRITE PROM COMMAND
FDC1	4B	2350	MOV C,E ; LOAD PROM LOW ADDRESS
FDC2	CDCEFF	2351	CALL PIODR3 ; OUTPUT WRITE PROM COMMAND
		2352	; OUTPUT PROM LOW ADDRESS
FDC5	4A	2353	MOV C,D ; LOAD PROM HIGH ADDRESS
FDC6	CDD1FF	2354	CALL PIODR4 ; OUTPUT PROM HIGH ADDRESS
FDC9	C1	2355	POP B ; RESTORE B,C; C CONTAINS THE DATA TO BE
		2356	; WRITTEN TO THE PROM



```

LOC  OBJ          LINE      SOURCE STATEMENT

FDCA  CDD1FF      2357          CALL   Piodr4           ; OUTPUT DATA TO PROM
FDCC  C9          2358          RET
2359 ;;;;;;;;;;;;;;
2360 ; 'UPPS' - EXTERNALLY REFERENCED ROUTINE ;
2361 ; PROCESS: INPUT THE UPP STATUS BYTE ;
2362 ; INPUT: ;
2363 ; OUTPUT: A-REG CONTAINS THE UPP STATUS BYTE ;
2364 ; MODIFIED: ;
2365 ; STACK USAGE: 8 BYTES ;
2366 ;;;;;;;;;;;;;;
2367 UPPS:
FDCE  C5          2368          PUSH  B                ; SAVE BC
FDCE  0618        2369          MVI   B,RPSTC         ; B CONTAINS STATUS COMMAND
FDD1  CDB5FF      2370          CALL  Piodr1         ; GET UPP STATUS BYTE
FDD4  F5          2371          PUSH  PSW            ; SAVE IT ON THE STACK
FDD5  CDB8FF      2372          CALL  Piodr2         ; GET PIO DEVICE STATUS BYTE AND IGNORE IT
FDD8  F1          2373          POP   PSW            ; A NOW CONTAINS UPP STATUS BYTE
FDD9  C1          2374          POP   B              ; RESTORE BC
FDDA  C9          2375          RET
2376 ;*****
2377 ;*
2378 ;* END OF I/O SUBROUTINES, BEGINNING OF MONITOR SUBROUTINES *
2379 ;*
2380 ;*****
2381 ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
2382 ; 'BYTE' - ENTERED VIA CALL FROM 'R' COMMAND
2383 ; PROCESS: READ TWO 8-BIT ASCII CHARACTERS, DECODE INTO ONE 8-BIT BINARY WORD
2384 ; INPUT: D CONTAINS RUNNING CHECKSUM
2385 ; OUTPUT: DECODED BYTE IN A-REG, RUNNING CHECKSUM IN D-REG, ZERO BIT SET OR RESET
2386 ; MODIFIED: A,F,C,D
2387 ; STACK USAGE:
2388 BYTE:
FDDB  C5          2389          PUSH  B                ; SAVE B,C
FDDC  CD58FF      2390          CALL  RIX             ; READ ONE ASCII CHAR FROM TAPE, PUT IN A-REG
FDDF  CD98FE      2391          CALL  NIBBLE         ; CONVERT 8-BIT ASCII TO 4-BIT HEXADECIMAL VALUE
FDE2  07          2392          RLC                    ; SHIFT FOUR PLACES TO THE LEFT
FDE3  07          2393          RLC
FDE4  07          2394          RLC
FDE5  07          2395          RLC                    ; MOVE HEX CHAR TO 4 MSB OF A-REG
FDE6  4F          2396          MOV   C,A            ; STORE TEMPORARILY IN C
FDE7  CD58FF      2397          CALL  RIX             ; GET ANOTHER ASCII CHAR FROM READER
FDEA  CD98FE      2398          CALL  NIBBLE         ; CONVERT TO 4 BIT HEX; NOW LSB OF A-REG
FDED  B1          2399          ORA   C              ; ASSEMBLE IT ALL TOGETHER
FDEE  4F          2400          MOV   C,A            ; STORE IT TEMPORARILY IN C
FDEF  82          2401          ADD   D              ; UPDATE CHECKSUM (ZERO BIT IS SET/RESET)
FDF0  57          2402          MOV   D,A            ; D CONTAINS UPDATED CHECKSUM
FDF1  79          2403          MOV   A,C            ; LOAD THE CONVERTED WORD
FDF2  C1          2404          POP   B
FDF3  C9          2405          RET                    ; RETURN
2406 ;;;;;;;;;;;;;;
2407 ; 'CONV' - ENTERED VIA CALLS FROM 'DBYTE','HXD','PBYTE' ROUTINES
2408 ; PROCESS: CONVERT 4 BIT HEX VALUE TO ASCII CHARACTER
2409 ; INPUT : 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E, OR F IN HEX IN A-REG
2410 ; OUTPUT: 30H,...,39H,41H,...,46H IN C-REG
2411 ; MODIFIED: A, FLAGS, C

```

LOC	OBJ	LINE	SOURCE STATEMENT
		2412	; STACK USAGE:
		2413	;
		2414	CONV:
FDF4	E60F	2415	ANI     0FH                    ; ONLY 4 LSB ARE SIGNIFICANT, SO MASK 4 MSB
FDF6	C690	2416	ADI     90H                    ; SET UP A-REG SO THAT A-F CAUSE CARRY
FDF8	27	2417	DAA
FDF9	CE40	2418	ACI     40H                    ; ADD IN CARRY AND ADJUST UPPER NIBBLE
FDFB	27	2419	DAA
FDFC	4F	2420	MOV     C,A                    ; STORE CONVERTED RESULT IN C-REG
FDFD	C9	2421	RET                            ; RETURN
		2422	;/;;/
		2423	; 'CRLF' - ENTERED VIA CALLS FROM 'G','H','Q','R','W','X' COMMANDS AND
		2424	;                    'START' ROUTINE
		2425	; PROCESS: TYPE CARRIAGE RETURN AND LINE FEED ON LOCAL CONSOLE
		2426	; INPUT:
		2427	; OUTPUT:
		2428	; MODIFIED:
		2429	; STACK USAGE:
		2430	CRLF:
FDFE	CDDEFC	2431	CALL    COMC                    ; OUTPUT <CR> ON CONSOLE
FE01	0D	2432	DB     CR
FE02	CDDEFC	2433	CALL    COMC                    ; OUTPUT <LF> ON CONSOLE
FE05	0A	2434	DB     LF
FE06	C9	2435	RET
		2436	;/;;/
		2437	; 'DADR' - ENTERED VIA CALL FROM 'D' COMMAND
		2438	; PROCESS: PRINT CONTENTS OF HL IN HEX FORMAT ON LIST DEVICE
		2439	; INPUT: HL CONTAINS <LOW ADDRESS> OF 'D' COMMAND
		2440	; OUTPUT:
		2441	; MODIFIED: A
		2442	; STACK USAGE:
		2443	DADR:
FE07	7C	2444	MOV     A,H                    ; PRINT MSB OF LOW ADDRESS
FE08	CD0CFE	2445	CALL    DBYTE
FE0B	7D	2446	MOV     A,L                    ; PRINT LSB OF LOW ADDRESS
		2447	;*****JMP    DBYTE
		2448	;/;;/
		2449	; 'DBYTE' - ENTERED VIA CALLS FROM 'D' COMMAND AND 'DADR' ROUTINE
		2450	;                    ENTERED VIA FALL-THRU FROM 'DADR' ROUTINE
		2451	; PROCESS: LIST A BYTE ON THE LIST DEVICE AS TWO ASCII CHARACTERS
		2452	; INPUT: A CONTAINS THE BYTE TO BE LISTED
		2453	; OUTPUT:
		2454	; MODIFIED:
		2455	; STACK USAGE:
		2456	DBYTE:
FE0C	F5	2457	PUSH    PSW                    ; SAVE A COPY OF A-REG
FE0D	0F	2458	RRC
FE0E	0F	2459	RRC
FE0F	0F	2460	RRC
FE10	0F	2461	RRC
FE11	CD4FD	2462	CALL    CONV                    ; WANT TO LOOK ONLY AT BITS 4-7 OF A-REG
FE14	CD14FD	2463	CALL    LOM                    ; CONVERT 4 MSB OF ORIGINAL A-REG TO 1 ASCII CHAR
FE17	F1	2464	POP     PSW                    ; OUTPUT ON LIST DEVICE
FE18	CD4FD	2465	CALL    CONV                    ; RETRIEVE ORIGINAL VALUE
FE1B	C314FD	2466	CALL    CONV                    ; CONVERT 4 LSB OF ORIGINAL A-REG TO 1 ASCII CHAR
			JMP     LOM                    ; OUTPUT ON LIST DEVICE

LOC	OBJ	LINE	SOURCE STATEMENT
		2467	;//
		2468	; 'DELAY' - ENTERED VIA CALL FROM 'RI' ROUTINE
		2469	; PROCESS: 1.0 MS. DELAY
		2470	; INPUT: ONEMS
		2471	; OUTPUT: ROUTINE IDLES FOR 1.0 MS.
		2472	; MODIFIED: C, FLAGS
		2473	; STACK USAGE: 2 BYTES
		2474	DELAY:
FE1E	0E70	2475	MVI     C,ONEMS                   ; LOAD 1 MS.CONSTANT (USE 3BH IN ICE ENVIRONMENT)
		2476	DLY1:
FE20	0D	2477	DCR     C                         ; DECREMENT COUNTER
FE21	C220FE	2478	JNZ     DLY1                     ; JUMP IF NOT EXPIRED
FE24	C9	2479	RET                               ; RETURN
		2480	;//
		2481	; 'DREG' - ENTERED VIA CALL FROM 'X' COMMAND
		2482	; PROCESS: DISPLAY THE CONTENTS OF A USER REGISTER
		2483	; INPUT: HL POINTS TO CHARACTER IN ACTBL OF 'X' COMMAND
		2484	; OUTPUT: HL POINTS TO NEXT CHARACTER IN ACTBL,
		2485	DE CONTAINS ADDRESS OF REGISTER LOCATION
		2486	B CONTAINS REGISTER PRECISION
		2487	; MODIFIED:
		2488	; STACK USAGE:
		2489	DREG:
FE25	23	2490	INX     H                         ; HL POINTS TO LOCATION ENTRY IN ACTBL OF 'X' COMMAND
FE26	5E	2491	MOV     E,M                      ; INCREMENT HL TO POINT AT DISPLACEMENT
FE27	3A0500	2492	LDA     MEMTOP+1
FE2A	57	2493	MOV     D,A                      ; D := MSB OF ADDRESS OF TOP PAGE OF MEMORY
		2494	; DE POINTS TO THAT PART OF THE EXIT TEMPLATE
		2495	;         CONTAINING SAVED REGISTER VALUES
FE2B	23	2496	INX     H                         ; HL POINTS TO PRECISION IN ACTBL
FE2C	46	2497	MOV     B,M                      ; PRECISION, 0=8 BITS, 1=16 BITS
FE2D	23	2498	INX     H                         ; POINT AT NEXT REGISTER IDENTIFIER
FE2E	1A	2499	LDAX    D                         ; 8/16 BIT DISPLAY AND MODIFICATION
FE2F	CD5BFE	2500	CALL    LBYTE                    ; MSB OF 16 BIT REG, ALL OF 8 BIT REG
FE32	05	2501	DCR     B                         ; TEST PRECISION
FE33	F8	2502	RM                                ; 8 BIT DISPLAY, RETURN
FE34	1B	2503	DCX     D
FE35	1A	2504	LDAX    D
FE36	C35BFE	2505	JMP     LBYTE                    ; LSB OF 16 BIT REG
		2506	;//
		2507	; 'EXPR' - ENTERED VIA CALLS FROM 'D','E','F','H','M','R','W' COMMANDS
		2508	; PROCESS: EVALUATE EXPRESSION "<EXPR>,<EXPR>,<EXPR>"
		2509	; INPUT: C-REG CONTAINS THE NUMBER OF PARAMETERS REQUIRED (1,2, OR 3)
		2510	; OUTPUT: STACK CONTAINS THE PARAMETERS IN REVERSE ORDER
		2511	; MODIFIED: F,C,H,L,SP
		2512	; STACK USAGE:
		2513	EXPR:
FE39	CD74FE	2514	CALL    PARAM                    ; GET A HEXADECIMAL PARAMETER, RETURNED IN HL
FE3C	E3	2515	XTHL                              ; PUT THE PARAMETER IN THE STACK; HL NOW
		2516	;         CONTAINS RETURN ADDRESS OF CALL TO 'EXPR'
FE3D	E5	2517	PUSH    H                         ; PUT RETURN ADDRESS ON TOP OF STACK
FE3E	0D	2518	DCR     C                         ; DECREMENT PARAMETER COUNT; CARRY BIT UNAFFECTED
FE3F	D246FE	2519	JNC     EX0                      ; JUMP IF COMMA ENTERED (PARAM CALLS PCHK)
FE42	C247F8	2520	JNZ     ERROR                    ; INCORRECT PARAM COUNT
FE45	C9	2521	RET

```

LOC  OBJ          LINE      SOURCE STATEMENT
                                2522 EX0:
FE46 C239FE      2523          JNZ      EXPR          ; GET ANOTHER PARAMETER
FE49 C347F8      2524          JMP      ERROR          ; NOT TERMINATED WITH CR
                                2525 ;////////////////////////////////////
                                2526 ; 'HILO' - ENTERED VIA CALLS FROM 'D','P','M','W' COMMANDS
                                2527 ; PROCESS: COMPARE HL WITH DE
                                2528 ; INPUT: ADDRESS VALUES IN HL AND DE
                                2529 ; OUTPUT: IF HL <= DE THEN CARRY = 0;
                                2530 ;          IF HL > DE THEN CARRY = 1
                                2531 ; MODIFIED: HL,A,F
                                2532 ; STACK USAGE:
                                2533 HILO:
FE4C 23          2534          INX      H          ; INCREMENT HL ADDRESS
FE4D 7C          2535          MOV      A,H          ; TEST FOR HL = 0
FE4E B5          2536          ORA      L          ; ZERO BIT SET IF H=L=00, I.E. HL MUST
                                2537 ;          HAVE BEEN FFFFH
FE4F 37          2538          STC          ; CARRY := 1
FE50 C8          2539          RZ
FE51 7B          2540          MOV      A,E          ; DE - HL, SET/RESET CARRY
FE52 95          2541          SUB      L          ; (LSB OF HIGH ADDR) - (MSB OF LOW ADDR)
FE53 7A          2542          MOV      A,D
FE54 9C          2543          SBB      H          ; (MSB OF HIGH ADDR) - (MSB OF LOW ADDR)
FE55 C9          2544          RET          ; RETURN
                                2545 ;////////////////////////////////////
                                2546 ; 'LADR' - ENTERED VIA CALLS FROM 'H' COMMAND AND 'RESTART' ROUTINE
                                2547 ; PROCESS: PRINT CONTENTS OF HL IN HEX ON LOCAL CONSOLE DEVICE
                                2548 ; INPUT: HL CONTAINS THE HEX VALUE TO BE OUTPUT(16 BITS)
                                2549 ; OUTPUT:
                                2550 ; MODIFIED: H,L,A
                                2551 ; STACK USAGE:
                                2552 LADR:
FE56 7C          2553          MOV      A,H
FE57 CD5BF8      2554          CALL     LBYTE          ; PRINT 8 MSB OF HEX VALUE ON CONSOLE
FE5A 7D          2555          MOV      A,L
                                2556 ;*****JMP     LBYTE          ; PRINT 8 LSB OF HEX VALUE ON CONSOLE
                                2557 ;////////////////////////////////////
                                2558 ; 'LBYTE' - ENTERED VIA CALLS FROM 'S' COMMAND AND 'DREG','LADR' ROUTINES
                                2559 ;          ENTERED VIA JUMP FROM 'DREG' ROUTINE
                                2560 ;          ENTERED VIA FALL-THRU FROM 'LADR' ROUTINE
                                2561 ; PROCESS: LIST A BYTE AS TWO ASCII CHARACTERS
                                2562 ; INPUT: A-REG CONTAINS THE 8 BITS TO BE CONVERTED TO ASCII
                                2563 ; OUTPUT:
                                2564 ; MODIFIED: A,F
                                2565 ; STACK USAGE: 6 BYTES
                                2566 LBYTE:
FE5B F5          2567          PUSH     PSW          ; SAVE A-REG
FE5C 0F          2568          RRC
FE5D 0F          2569          RRC
FE5E 0F          2570          RRC
FE5F 0F          2571          RRC          ; LOOK ONLY AT 4 MSB OF THE BYTE VALUE
FE60 CD64FE      2572          CALL     HXD          ; CONVERT IT TO ONE ASCII CHAR AND OUTPUT IT
FE63 F1          2573          POP      PSW          ; RETRIEVE ORIGINAL VALUE
                                2574 ;*****JMP     HXD          ; CONVERT 4 LSB OF BYTE TO ASCII AND OUTPUT IT
                                2575 ;////////////////////////////////////
                                2576 ; 'HXD' - ENTERED VIA CALL FROM 'LBYTE' ROUTINE

```

```

LOC  OBJ          LINE      SOURCE STATEMENT
                2577 ;          ENTERED VIA FALL-THRU FROM 'LBYTE' ROUTINE
                2578 ; PROCESS: CONVERT 4 LSB IN A-REG INTO ONE ASCII CHAR IN A-REG, PRINT IT
                2579 ;          ON LOCAL CONSOLE DEVICE
                2580 ; INPUT: NIBBLE TO BE CONVERTED IS IN BITS 0-3 OF A-REG
                2581 ; OUTPUT:
                2582 ; MODIFIED: A-REG
                2583 ; STACK USAGE:
                2584 HXD:
FE64  CDF4FD      2585          CALL    CONV          ; CONVERT 4 BITS TO ONE 8-BIT ASCII CHAR
FE67  C395FC      2586          JMP     COM          ; OUTPUT ON LOCAL CONSOLE
                2587 ; //////////////////////////////////////
                2588 ; 'LCRLF' - ENTERED VIA CALL FROM 'D' COMMAND
                2589 ; PROCESS: PRINT <CR>,<LF> ON LIST DEVICE
                2590 ; INPUT:
                2591 ; OUTPUT:
                2592 ; MODIFIED: C
                2593 ; STACK USAGE: 4 BYTES
                2594 LCRLF:
FE6A  0E0D        2595          MVI     C,CR
FE6C  CD14FD      2596          CALL    LOM          ; OUTPUT <CR> TO LIST DEVICE
FE6F  0E0A        2597          MVI     C,LF
FE71  C314FD      2598          JMP     LOM          ; OUTPUT <LF> TO LIST DEVICE
                2599 ; //////////////////////////////////////
                2600 ; 'PARAM' - ENTERED VIA CALLS FROM 'G','S' COMMANDS AND 'EXPR' ROUTINE
                2601 ; 'PA0' - ENTERED VIA CALLS FROM 'G','S','X' COMMANDS
                2602 ; PROCESS: COLLECT A HEXADECIMAL PARAMETER
                2603 ; INPUT:
                2604 ; OUTPUT: HEXADECIMAL PARAMETER IN HL
                2605 ; MODIFIED: A,F,B,H,L
                2606 ; STACK USAGE:
                2607 PARAM:
FE74  CDC5FE      2608          CALL    PCHK         ; GET FIRST CHARACTER
FE77  CA47F8      2609          JZ     ERROR        ; DISALLOW NULL PARAMETERS
                2610 PA0:
FE7A  210000      2611          LXI     H,0          ; INITIALIZE HL := 0000
                2612 PA1:
FE7D  47          2613          MOV     B,A          ; SAVE CHAR IN CASE IT'S A DELIMITER
FE7E  CD98FE      2614          CALL    NIBBLE       ; CONVERT THE ASCII CHARACTER TO HEX; MUST BE
                2615          ;          0-9,A-F; IF NOT THE CARRY BIT IS SET
FE81  DA90FE      2616          JC     PA2         ; NOT LEGAL CHAR, TREAT AS DELIMITER
FE84  29          2617          DAD    H          ; *2
FE85  29          2618          DAD    H          ; *4
FE86  29          2619          DAD    H          ; *8
FE87  29          2620          DAD    H          ; *16 --- SHIFT THE OLD HEX VALUES 4 PLACES TO LEFT
FE88  B5          2621          ORA    L          ; PUT NEW HEX VALUE IN 4 LSB OF L-REG
FE89  6F          2622          MOV     L,A
FE8A  CD61FF      2623          CALL    TI          ; GET SUBSEQUENT CHARACTERS
FE8D  C37DFE      2624          JMP     PA1         ; DECODE NEXT CHARACTER
                2625 PA2:
FE90  78          2626          MOV     A,B          ; A := B := DELIMITER CHARACTER
FE91  CDC8FE      2627          CALL    P2C         ; IS IT A VALID DELIMITER?
FE94  C247F8      2628          JNZ    ERROR        ; JUMP TO ERROR IF IT ISN'T
FE97  C9          2629          RET
                2630 ; //////////////////////////////////////
                2631 ; 'NIBBLE' - ENTERED VIA CALLS FROM 'BYTE','PARAM','PA0' ROUTINES

```

LOC	OBJ	LINE	SOURCE STATEMENT
		2632	; PROCESS: DECODE 8-BIT ASCII CHAR IN A-REG INTO 4-BIT HEX DIGIT IN A-REG,
		2633	;            FILTER OUT ALL CHARACTERS NOT IN THE ASCII CODING SEQUENCE
		2634	;            0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F.
		2635	; INPUT: 8-BIT ASCII CHAR IN A-REG
		2636	; OUTPUT: VALID HEX EQUIVALENT IN A-REG AND CARRY = 0, OTHERWISE
		2637	;            GARBAGE IN A-REG AND CARRY = 1 (INDICATING ILLEGAL CHARACTER)
		2638	; MODIFIED: A, FLAGS
		2639	; STACK USAGE: 2 BYTES
		2640	NIBBLE:
FE98	D630	2641	SUI        '0'                    ; IF THE ASCII CHAR IS BETWEEN 00 AND 2FH,
FE9A	D8	2642	RC                                ;        THEN RETURN WITH CARRY = 1
FE9B	C6E9	2643	ADI        '0' - 'G'             ; IF THE ASCII CHAR IS GREATER THAN 46H,
FE9D	D8	2644	RC                                ;        THEN RETURN WITH CARRY = 1
FE9E	C606	2645	ADI        6                     ; ORIGINAL ASCII CHAR WAS BETWEEN 30H AND 46H INCL.
FEA0	F2A6FE	2646	JP        NI0                    ; JUMP IF IT WAS 41H THRU 46H (I.E. A-F)
FEA3	C607	2647	ADI        7                     ; ORIGINAL ASCII CHAR WAS BETWEEN 30H AND 40H INCL.
FEA5	D8	2648	RC                                ; RETURN WITH CARRY = 1 IF ASCII CHAR WAS
		2649	;            BETWEEN 3AH AND 40H INCLUSIVE
		2650	NI0:                              ; VALID VALUE: 30H-39H,41H-46H
FEA6	C60A	2651	ADI        10                    ; A-REG NOW CONTAINS HEX EQUIV.(0-9,A-F)
FEA8	B7	2652	ORA        A                     ; CLEAR ERROR FLAG (I.E. RESET CARRY BIT)
FEA9	C9	2653	RET                               ; RETURN WITH VALID VALUE
		2654	;/;;/
		2655	; 'PADR' - ENTERED VIA CALLS FROM 'E','W' COMMANDS
		2656	; PROCESS: PUNCH CONTENTS OF HL IN HEX ON PUNCH DEVICE
		2657	; INPUT: HL CONTAINS 8-BIT LOAD ADDRESS
		2658	; OUTPUT:
		2659	; MODIFIED: A
		2660	; STACK USAGE: 4 BYTES
		2661	PADR:
FEAA	7C	2662	MOV        A,H                   ; A := MSB OF LOAD ADDRESS
FEAB	CDAFFE	2663	CALL      PBYTE                 ; EMIT FRAMES 3 & 4
FEAE	7D	2664	MOV        A,L                   ; A := LSB OF LOAD ADDRESS
		2665	;*****JMP      PBYTE           ; EMIT FRAMES 5 & 6
		2666	;/;;/
		2667	; 'PBYTE' - ENTERED VIA CALLS FROM 'E','W' COMMANDS AND 'PADR' ROUTINE
		2668	;            ENTERED VIA FALL-THRU FROM 'PADR' ROUTINE
		2669	; PROCESS: PUNCH A BYTE AS 2 ASCII CHARACTERS
		2670	; INPUT: A-REG CONTAINS BYTE TO BE CONVERTED, D CONTAINS RUNNING CHECKSUM
		2671	; OUTPUT: D CONTAINS UPDATED CHECKSUM
		2672	; MODIFIED: A,F,D,E
		2673	; STACK USAGE:
		2674	PBYTE:
FEAF	5F	2675	MOV        E,A                   ; SAVE BYTE TO BE CONVERTED IN E-REG
FEB0	0F	2676	RRC                               ;
FEB1	0F	2677	RRC                               ;
FEB2	0F	2678	RRC                               ;
FEB3	0F	2679	RRC                               ;
FEB4	CDF4FD	2680	CALL      CONV                 ; LOOK ONLY AT 4 MSB OF THE BYTE
FEB7	CDE9FC	2681	CALL      PO                    ; CONVERT IT TO 1 ASCII CHARACTER
FEBA	7B	2682	MOV        A,E                   ; NOW LOOK ONLY AT 4 LSB OF BYTE
FEBB	CDF4FD	2683	CALL      CONV                 ; CONVERT IT TO ONE ASCII CHAR
FEBE	CDE9FC	2684	CALL      PO                    ; PUNCH IT
FEC1	7B	2685	MOV        A,E
FEC2	82	2686	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 ;/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
2690 ; 'PCHK' - ENTERED VIA CALLS FROM 'G','S','X' COMMANDS AND 'PARAM' ROUTINE
2691 ; 'P2C' - ENTERED VIA CALLS FROM 'PARAM','PA0' 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 = 0
2696 ;           IF <CR>, THEN ZERO = 1 AND CARRY = 1
2697 ;           IF NONE OF ABOVE, THEN ZERO = 0 AND CARRY = 0
2698 ; MODIFIED: A, FLAGS
2699 ; STACK USAGE: 4 BYTES
2700 PCHK:
FEC5 CD61FF  2701      CALL    TI           ; GET A CHARACTER
2702 P2C:
FEC8 FE20   2703      CPI     ' '
FECA C8     2704      RZ                   ; IF SPACE, THEN ZERO = 1 & CARRY = 0
FECB FE2C   2705      CPI     ', '
FECD C8     2706      RZ                   ; IF COMMA, THEN ZERO = 1 & CARRY = 0
FECE FE0D   2707      CPI     CR
FED0 37     2708      STC
FED1 C8     2709      RZ                   ; IF <CR>, THEN ZERO = 1 & CARRY = 1
FED2 3F     2710      CMC
FED3 C9     2711      RET                   ; IF NONE OF THE THREE, THEN ZERO=CARRY=0
2712 ;/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
2713 ;/ 'RESTART' - ENTERED VIA JUMP FROM LOCATION 0 //
2714 ;/ PROCESS: BREAKPOINT/INTERRUPT/RESTART PROCESSING //
2715 ;/ INPUT: //
2716 ;/ OUTPUT: //
2717 ;/ MODIFIED: //
2718 ;/ EXPLANATION: //
2719 ;/ THIS ROUTINE IS ENTERED VIA A RESTART 0 (RST 0) INSTRUCTION. THE //
2720 ;/ 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 0 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 //
2730 ;/ 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 0 INSTRUCTION OR WHEN CONSOLE //
2739 ;/ INTERRUPT). ALSO, RESTORE THE TRAP VALUES AT THE PROPER //
2740 ;/ TRAP ADDRESSES. //
2741 ;/ 5. RETURN CONTROL TO THE MONITOR (BY JUMPING TO START). //

```

LOC	OBJ	LINE	SOURCE STATEMENT
		2742	;/
		2743	;/
		2744	RESTART:
FED4	F3	2745	DI    ; DISABLE IF SOFTWARE TRAP
		2746	; SAVE USER'S ENVIRONMENT
FED5	E5	2747	PUSH    H                                 ; SAVE H,L
FED6	D5	2748	PUSH    D                                 ; SAVE D,E
FED7	C5	2749	PUSH    B                                 ; SAVE B,C
FED8	F5	2750	PUSH    PSW                              ; SAVE A,FLAGS
FED9	D1	2751	POP      D                                ; TEMPORARILY SAVE PSW IN D & E
FEDA	E5	2752	PUSH    H                                ; DUMMY PUSH TO RESERVE SPACE IN STACK FOR
		2753	;     CURRENT INTERRUPT MASK AND CONFIGURATION
		2754	;     BYTE
FEDB	2A0400	2755	LHLD    MEMTOP
FEDE	2ECC	2756	MVI     L,ILOC-1 AND 0FFH; HL NOW POINTS TO CONFIGURATION BYTE IN
		2757	;     EXIT CODE IN TOP PAGE OF RAM
FEE0	6E	2758	MOV     L,M                              ; L NOW CONTAINS THIS CONFIGURATION BYTE
FEE1	DBFC	2759	IN      SOCP1                            ; INPUT CURRENT INTERRUPT MASK REGISTER ---
		2760	;     THIS MASK IS THE USER'S, SO SAVE IT
FEE3	67	2761	MOV     H,A                              ; H NOW CONTAINS THIS INTERRUPT MASK
FEE4	E3	2762	XTHL                                     ; THE INTERRUPT MASK AND CONFIGURATION BYTE
		2763	;     ARE NOW ON TOP OF THE USER STACK
FEE5	D5	2764	PUSH    D                                ; NOW PUT THE ORIGINAL PSW ON TOP OF THE STACK
FEE6	3EFE	2765	MVI     A,NOT INT0                      ; SET MONITOR'S DEFAULT INTERRUPT MASK
FEE8	D3FC	2766	OUT     SOCP1                            ; OUTPUT NEW MASK
FEEA	2A0400	2767	LHLD    MEMTOP
FEED	2ED2	2768	MVI     L,EXIT AND 0FFH ; HL NOW POINTS TO EXIT CODE AT TOP OF RAM
FEF0	EB	2769	XCHG                                     ; SO NOW DE POINTS TO EXIT CODE AT TOP OF RAM
FEF0	210C00	2770	LXI     H,12                             ; H := 00, L := 0C (DECIMAL VALUE 12)
FEF3	39	2771	DAD     SP                               ; EFFECT OF THIS IS TO CUT BACK THE USER'S
		2772	;     STACK TO WHAT IT WAS BEFORE ENTERING
		2773	;     THIS RESTART ROUTINE AND BEFORE THE PC
		2774	;     WAS PUSHED ON BY RST 0 OR INTERRUPT.
		2775	;     HL CONTAINS THIS 'OLD' STACK ADDRESS.
FEF4	0605	2776	MVI     B,5                              ; COUNT FOR TRANSFER OF MACHINE STATE
		2777	;     TO EXIT TEMPLATE STORAGE (MOVE THE STACK)
FEF6	EB	2778	XCHG                                     ; HL NOW POINTS TO EXIT CODE AT TOP OF RAM
		2779	; DE NOW POINTS TO USER STACK AS IT WAS
		2780	;     PRIOR TO RST 0 OR CONSOLE INTERRUPT.
		2781	;
		2782	RST0:                                    ; MOVE THE MACHINE STATE FROM THE USER'S STACK
		2783	; TO THE RESERVED AREA IN THE EXIT TEMPLATE
		2784	; IN TOP PAGE OF RAM.
		2785	;     B=5     !     B=4     !     B=3     !     B=2     !     B=1
		2786	-----!-----!-----!-----!-----
FEF7	2B	2787	DCX     H                                ;     !     !     !     !     !
FEF8	72	2788	MOV     M,D                              ;SLOC=MSB(SP)!ALOC=A !ILOC=INT!BLOC=B!DLOC=D
FEF9	2B	2789	DCX     H                                ;     !     !     !     !     !
FEFA	73	2790	MOV     M,E                              ;     =LSB(SP)!FLOC=FLG!     =FLG!CLOC=C!ELOC=E
FEFB	D1	2791	POP     D                                ;DE=AF     !DE=INT,F!DE=BC     !DE=DE !DE=HL
FEFC	05	2792	DCR     B                                ;B=4     !B=3     !B=2     !B=1     !B=0
FEFD	C2F7FE	2793	JNZ     RST0
		2794	;
		2795	; AT THIS POINT, HL POINTS TO THE BASE OF
		2796	;     THE MONITOR STACK (TOS) IN TOP PAGE OF



LOC	OBJ	LINE	SOURCE STATEMENT		
		2797			; RAM. DE CONTAINS THE H & L VALUES THE
		2798			; USER HAD PRIOR TO ENTERING THE RESTART
		2799			; ROUTINE.
FF00	C1	2800	POP	B	; BC = OLD PC (PUSHED ON USER STACK BY
		2801			; RST 0 OR INTERRUPT)
FF01	0B	2802	DCX	B	; DECREMENT TO POINT AT TRAPPED CODE
FF02	F9	2803	SPHL		; SP NOW POINTS TO TOS (BASE OF MONITOR STACK)
FF03	2A0400	2804	LHLD	MEMTOP	
FF06	2EE2	2805	MVI	L,TLOC AND 0FFH	; HL NOW POINTS TO TLOC IN TOP PAGE OF RAM
		2806			; I.E. LSB OF TRAP 1 ADDRESS
FF08	7E	2807	MOV	A,M	; TEST IF THIS IS A PROGRAMMED RESTART OR A
FF09	91	2808	SUB	C	; LOCAL CONSOLE INTERRUPT BY COMPARING THE
		2809			; PC VALUE WITH TRAP 1 ADDRESS
		2810			; A := LSB OF TRAP 1 ADDRESS
FF0A	23	2811	INX	H	; HL POINTS TO MSB OF TRAP 1 ADDRESS
FF0B	C213FF	2812	JNZ	RSTA	; PC DID NOT MATCH TRAP 1 ADDRESS
FF0E	7E	2813	MOV	A,M	; A := MSB OF TRAP 1 ADDRESS
FF0F	98	2814	SBB	B	
FF10	CA25FF	2815	JZ	RST1	; PC MATCHES TRAP 1 --- A PROGRAMMED RESTART
		2816	RSTA:		; REPEAT SAME STEPS AS ABOVE BUT SEE IF PC
		2817			; MATCHES 2ND BREAKPOINT (TRAP 2 ADDRESS)
FF13	23	2818	INX	H	; HL POINTS TO TRAP 1 OPCODE VALUE
FF14	23	2819	INX	H	; HL POINTS TO LSB OF TRAP 2 ADDRESS
FF15	7E	2820	MOV	A,M	; A := LSB OF TRAP 2 ADDRESS
FF16	91	2821	SUB	C	
FF17	23	2822	INX	H	; HL POINTS TO MSB OF TRAP 2 ADDRESS
FF18	C220FF	2823	JNZ	RSTB	; PC DID NOT MATCH TRAP 2 ADDRESS
FF1B	7E	2824	MOV	A,M	; A := MSB OF TRAP 2 ADDRESS
FF1C	98	2825	SBB	B	
FF1D	CA25FF	2826	JZ	RST1	; PC MATCHES TRAP 2 --- A PROGRAMMED RESTART
		2827	RSTB:		; NOT A PROGRAMMED RESTART, BUT A
FF20	3E20	2828	MVI	A,EOI	; CONSOLE INTERRUPT SO SEND EOI TO 8259
FF22	D3FD	2829	OUT	SOCPO	
FF24	03	2830	INX	B	; ADJUST PC FOR LOCAL CONSOLE RESTART
		2831			; I.E. GET READY TO POINT PC TO
		2832			; RESUMPTION POINT IN CODE IT WAS
		2833			; EXECUTING WHEN INTERRUPTED
		2834			; BC POINTS TO NEXT INSTR TO BE EXECUTED
		2835			; WHEN CONTROL IS RETURNED TO USER PROGRAM
		2836	RST1:		; PROGRAMMED RESTART AT A BREAKPOINT (TRAP)
		2837			; ALSO FALLTHROUGH FROM CONSOLE INTERRUPT
FF25	2A0400	2838	LHLD	MEMTOP	
FF28	2EDC	2839	MVI	L,LLOC AND 0FFH	; HL NOW POINTS TO LLOC IN EXIT CODE IN TOP OF RAM
FF2A	73	2840	MOV	M,E	; USER'S L VALUE PRIOR TO RESTART IS STORED IN LLOC
FF2B	23	2841	INX	H	
FF2C	72	2842	MOV	M,D	; USER'S H VALUE PRIOR TO RESTART IS STORED IN HLOC
		2843			-----
FF2D	2EE0	2844	MVI	L,PLOC-1 AND 0FFH	; HL POINTS TO LSB OF JMP INSTR IN EXIT CODE
FF2F	71	2845	MOV	M,C	; SAVE LSB OF USER'S PC
FF30	23	2846	INX	H	
FF31	70	2847	MOV	M,B	; SAVE MSB OF USER'S PC. EFFECT IS TO LOAD THE
		2848			; PROPER ADDRESS INTO THE EXIT TEMPLATE FOR THE
		2849			; JUMP BACK TO THE USER'S PROGRAM.
		2850			-----
FF32	C5	2851	PUSH	B	

LOC	OBJ	LINE	SOURCE STATEMENT
FF33	CDDEFC	2852	CALL COMC
FF36	23	2853	DB '#'
FF37	E1	2854	POP H ; RETRIEVE OLD PC FOR DISPLAY
FF38	CD56FE	2855	CALL LADR ; DISPLAY PC
		2856 ;	-----
		2857	; CLEAR TRAPS
FF3B	2A0400	2858	LHLD MEMTOP
FF3E	2EE2	2859	MVI L,TLOC AND 0FFH ; HL NOW POINTS TO TLOC IN TOP PAGE OF RAM
FF40	1602	2860	MVI D,2 ; SET COUNT FOR TWO TRAPS
		2861 RST2:	
FF42	4E	2862	MOV C,M ; C := LSB OF TRAP ADDRESS
FF43	AF	2863	XRA A
FF44	77	2864	MOV M,A ; ZERO OUT LSB OF TRAP ADDRESS
FF45	23	2865	INX H
FF46	46	2866	MOV B,M ; B := MSB OF TRAP ADDRESS
FF47	77	2867	MOV M,A ; ZERO OUT MSB OF TRAP ADDRESS
FF48	23	2868	INX H ; HL NOW POINTS TO TRAP VALUE
FF49	79	2869	MOV A,C ; BC CONTAINS THE TRAP ADDRESS
FF4A	B0	2870	ORA B ; TEST FOR VALID TRAP
FF4B	CA50FF	2871	JZ RST3 ; TRAP ADDRESS IS 0, SO NO TRAP TO RESTORE
FF4E	7E	2872	MOV A,M ; GET OPCODE BYTE, I.E. TRAP VALUE
FF4F	02	2873	STAX B ; PUT IT BACK IN CORRECT PLACE IN USER PROGRAM,
		2874	; I.E. REPLACE THE RST 0 INSTR WITH ORIGINAL
		2875	; OPCODE.
		2876 RST3:	
FF50	23	2877	INX H ; POINT TO TRAP 2 ADDRESS IF D=2
FF51	15	2878	DCR D
FF52	C242FF	2879	JNZ RST2 ; REPEAT FOR TRAP 2
FF55	C355F8	2880	JMP START ; ENTER MONITOR (INTERRUPTS STILL DISABLED)
		2881 ;	////////////////////////////////////
		2882 ;	'RIX' - ENTERED VIA CALLS FROM 'R' COMMAND AND 'BYTE' ROUTINE
		2883 ;	PROCESS: GET A CHARACTER FROM READER, MASK OFF PARITY BIT
		2884 ;	INPUT:
		2885 ;	OUTPUT: CHARACTER IN A-REG, BIT 7 IS 0
		2886 ;	MODIFIED: A,F
		2887 ;	STACK USAGE:
		2888 RIX:	
FF58	CD0FFC	2889	CALL RI ; GET CHARACTER FROM READER DEVICE
FF5B	DA47F8	2890	JC ERROR ; READER TIMEOUT ERROR
FF5E	E67F	2891	ANI 7FH ; MASK OUT THE PARITY BIT
FF60	C9	2892	RET ; RETURN
		2893 ;	////////////////////////////////////
		2894 ;	'TI' - ENTERED VIA CALLS FROM 'A','N','Q' COMMANDS AND 'START','PARAM'
		2895 ;	'PA0','PCHK' ROUTINES
		2896 ;	ENTERED VIA JUMP FROM 'BREAK'
		2897 ;	PROCESS: INPUT FROM LOCAL CONSOLE, ECHO, RETURN IN A-REG
		2898 ;	INPUT:
		2899 ;	OUTPUT: CHARACTER IN A-REG
		2900 ;	MODIFIED: A,F
		2901 ;	STACK USAGE:
		2902 TI:	
FF61	C5	2903	PUSH B ; SAVE STATE OF B- & C-REGS
FF62	CDBEFB	2904	CALL CI ; GET A CHARACTER FROM THE CONSOLE
FF65	E67F	2905	ANI 7FH ; MASK OFF PARITY BIT
FF67	CD76FF	2906	CALL UC ; CONVERT TO UPPER CASE

```

LOC  OBJ          LINE      SOURCE STATEMENT

FF6A FE03        2907      CPI      ETX          ; TEST FOR BREAK
FF6C CA47F8      2908      JZ       ERROR       ; ABORT COMMAND
FF6F 4F          2909      MOV      C,A         ; MOVE INPUT CHARACTER TO C-REG
FF70 CD9FFC      2910      CALL    CO           ; ECHO IT
FF73 79          2911      MOV      A,C         ;
FF74 C1          2912      POP      B           ; RESTORE STATE OF B & C
FF75 C9          2913      RET                    ; RETURN
2914 ;////////////////////////////////////
2915 ; 'UC' - ENTERED VIA CALL FROM 'TI' ROUTINE
2916 ; PROCESS: CONVERT CHARACTER IN A-REG FROM LOWER CASE TO UPPER CASE
2917 ; INPUT: LOWER OR UPPER CASE CHAR IN A-REG
2918 ; OUTPUT: UPPER CASE CHARACTER IN A-REG
2919 ; MODIFIED: A,F
2920 ; STACK USAGE:
2921 UC:
FF76 FE61        2922      CPI      'A'+20H
FF78 F8          2923      RM                    ; CHAR < LC(A) , I.E. IF THE CHAR IN A-REG
2924 ; IS NOT LOWER CASE, THEN IT HAS VALUE
2925 ; < 61H, SO A - 61H WILL BE MINUS. IF
2926 ; IT IS IN LOWER CASE, THE RESULT WILL
2927 ; BE POSITIVE.
FF79 FE7B        2928      CPI      'Z'+20H+1
FF7B F0          2929      RP                    ; CHAR > LC(Z) , I.E. WE KNOW THE A-REG IS
2930 ; UPPER CASE OR SPECIAL CHAR. IF IT IS A
2931 ; SPECIAL CHAR, A - 78H WILL BE 0 OR
2932 ; GREATER SO RETURN.
FF7C E6DF        2933      ANI     NOT 20H      ; FORCE UPPER CASE
FF7E C9          2934      RET
2935 ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
2936 ;*
2937 ;*      I/O CONTROLLER INTERFACE DRIVERS
2938 ;*
2939 ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
2940 ; 'IOCDR1' - ENTERED VIA CALLS FROM 'CI','CSTS' ROUTINES
2941 ; PROCESS: GET DEVICE STATUS OR GET DATA FROM PERIPHERAL
2942 ; INPUT: B CONTAINS THE COMMAND (STATUS REQUEST OR INPUT DATA REQUEST)
2943 ; OUTPUT: A CONTAINS THE REQUESTED INFORMATION
2944 ; MODIFIED: A,FLAGS,B
2945 ; STACK USAGE:
2946 IOCDR1:
FF7F CDA6FF      2947      CALL    IOCCOM       ; OUTPUT 'GET DEVICE STATUS COMMAND' OR
2948 ; 'INPUT DATA COMMAND' TO IOC CONTROL
2949 ; PORT
2950 IOCXXX:
FF82 DBC1        2951      IN      IOCS         ; INPUT DBB STATUS
FF84 E607        2952      ANI     IBF OR OBF OR F0 ; MASK OFF STATUS FLAGS
FF86 FE01        2953      CPI     OBF          ; TEST FOR SLAVE DONE; SOMETHING FOR THE MASTER
FF88 C282FF      2954      JNZ     IOCXXX       ; IF NOT, CONTINUE TO LOOP
FF8B DBC0        2955      IN      IOCI         ; OTHERWISE, INPUT THE DATA FROM THE DBB
FF8D F5          2956      PUSH   PSW          ; SAVE A-REG
FF8E 3E05        2957      MVI     A,ENABL     ; ENABLE INTERRUPTS
FF90 D3FF        2958      OUT    CPUC         ;
FF92 F1          2959      POP     PSW          ; RESTORE A-REG
FF93 C9          2960      RET
2961 ;-----

```

```

LOC  OBJ          LINE          SOURCE STATEMENT
                2962 ; 'IOCDR2' - ENTERED VIA CALLS FROM 'BLK','COM','CO','CRTOUT' ROUTINES
                2963 ; PROCESS: OUTPUT DATA TO THE PERIPHERAL DEVICE
                2964 ; INPUT: B CONTAINS THE COMMAND TO OUTPUT THE DATA
                2965 ;           C CONTAINS THE DATA TO BE OUTPUT
                2966 ; OUTPUT:
                2967 ; MODIFIED: A,FLAGS,B,C
                2968 ; STACK USAGE:
                2969 IOCDR2:
FF94  CDA6FF      2970          CALL    IOCCOM          ; OUTPUT 'OUTPUT DATA COMMAND' TO IOC
                2971                               ;   CONTROL PORT
                2972 IOCYYY:
FF97  DBC1        2973          IN      IOCS              ; INPUT DBB STATUS
FF99  E607        2974          ANI    IBF OR F0 OR OBF ; TEST FOR SLAVE PROCESSOR READY
FF9B  C297FF      2975          JNZ    IOCYYY          ; CONTINUE TO LOOP UNTIL IT IS READY
FF9E  79          2976          MOV    A,C              ; LOAD DATA TO BE WRITTEN
FF9F  D3C0        2977          OUT    IOCO              ; OUTPUT DATA TO THE DBB
FFA1  3E05        2978          MVI    A,ENABL         ; ENABLE INTERRUPTS
FFA3  D3FF        2979          OUT    CPUC
FFA5  C9          2980          RET
                2981 ; -----
                2982 ; 'IOCCOM' - COMMON ROUTINE TO IOC DRIVERS
                2983 ;           ENTERED VIA CALLS FROM 'IOCDR1' AND 'IOCDR2'
                2984 ; PROCESS: OUTPUT COMMAND TO THE IOC
                2985 ; INPUT: B CONTAINS THE COMMAND
                2986 ; OUTPUT:
                2987 ; MODIFIED: A,FLAGS
                2988 ; STACK USAGE:
                2989 IOCCOM:
FFA6  3E0D        2990          MVI    A,DISABL       ; BLOCK ALL INTERRUPTS
FFA8  D3FF        2991          OUT    CPUC
                2992 IOCZZZ:
FFAA  DBC1        2993          IN      IOCS              ; INPUT DBB STATUS
FFAC  E607        2994          ANI    F0 OR IBF OR OBF ; TEST FOR SLAVE PROCESSOR IDLE
FFAE  C2AAFF      2995          JNZ    IOCZZZ          ; LOOP UNTIL IT IS IDLE
FFB1  78          2996          MOV    A,B              ; LOAD COMMAND
FFB2  D3C1        2997          OUT    IOCC              ; OUTPUT COMMAND TO IOC CONTROL PORT
FFB4  C9          2998          RET
                2999 ; *-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
                3000 ; *
                3001 ; *           PARALLEL I/O INTERFACE DRIVERS
                3002 ; *
                3003 ; *-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
                3004 ; 'PIODR1' - ENTERED VIA CALLS FROM 'RI','PO','POC','LO','UPPS'
                3005 ; 'PIODR2' - ENTERED VIA CALLS FROM 'UI','UPPS' ROUTINES
                3006 ; PROCESS: GET DEVICE STATUS OR GET DATA FROM A PERIPHERAL
                3007 ; INPUT: B CONTAINS THE COMMAND (STATUS REQUEST OR INPUT DATA REQUEST)
                3008 ; OUTPUT: A CONTAINS THE REQUESTED INFORMATION
                3009 ; MODIFIED: A, FLAGS, B
                3010 ; STACK USAGE:
                3011 PIODR1:
FFB5  CDE4FF      3012          CALL    PIOCUM          ; OUTPUT 'GET DEVICE STATUS COMMAND' OR
                3013                               ;   'INPUT DATA COMMAND' OR OTHER SUCH
                3014                               ;   COMMAND TO THE PIO CONTROL PORT
                3015 PIODR2:
FFB8  3E0D        3016          MVI    A,DISABL       ; BLOCK ALL INTERRUPTS

```

```

LOC  OBJ          LINE          SOURCE STATEMENT

FFBA D3FF          3017          OUT      CPUC
FFBC DBF9          3018          IN       PIOS                ; INPUT DBB STATUS
FFBE E607          3019          ANI     F0 OR IBF OR OBF; MASK OFF STATUS FLAGS
FFC0 FE01          3020          CPI     OBF                ; TEST FOR SLAVE DONE; SOMETHING FOR THE MASTER
FFC2 C2B8FF       3021          JNZ     PIODR2             ; LOOP UNTIL SLAVE IS READY
FFC5 DBF8          3022          IN       PIOI                ; OTHERWISE INPUT THE DATA FROM THE DBB
FFC7 F5           3023          PUSH    PSW                ; SAVE A-REG
FFC8 3E05          3024          MVI     A,ENABL           ; ENABLE INTERRUPTS
FFCA D3FF          3025          OUT      CPUC
FFCC F1           3026          POP     PSW                ; RESTORE A-REG
FFCD C9           3027          RET
3028 ; -----
3029 ; 'PIODR3' - ENTERED VIA CALLS FROM 'POC','PO','LO','UI','UO' ROUTINES
3030 ; 'PIODR4' - ENTERED VIA CALLS FROM 'UI','UO'
3031 ; PROCESS: OUTPUT DATA TO A PERIPHERAL DEVICE
3032 ; INPUT: B CONTAINS THE COMMAND TO OUTPUT THE DATA
3033 ; C CONTAINS THE DATA TO BE OUTPUT
3034 ; OUTPUT:
3035 ; MODIFIED: A,FLAGS,B, C
3036 ; STACK USAGE:
3037 PIODR3:
FFCE CDE4FF       3038          CALL    PIOCOM            ; OUTPUT 'OUTPUT DATA COMMAND' TO PIO
3039 PIODR4:
FFD1 3E0D          3040          MVI     A,DISABL         ; BLOCK ALL INTERRUPTS
FFD3 D3FF          3041          OUT     CPUC
FFD5 DBF9          3042          IN       PIOS                ; INPUT DBB STATUS
FFD7 E607          3043          ANI     F0 OR IBF OR OBF; TEST FOR SLAVE PROCESSOR READY
FFD9 C2D1FF       3044          JNZ     PIODR4             ; LOOP UNTIL IT IS READY
FFDC 79           3045          MOV     A,C                ; LOAD DATA TO BE WRITTEN
FFDD D3F8          3046          OUT     PIOO                ; OUTPUT DATA TO THE DBB
FFDF 3E05          3047          MVI     A,ENABL           ; ENABLE INTERRUPTS
FFE1 D3FF          3048          OUT     CPUC
FFE3 C9           3049          RET
3050 ; -----
3051 ; 'PIOCOM' - COMMON ROUTINE OF PIO DRIVERS
3052 ; ENTERED VIA CALLS FROM 'PIODR1', 'PIODR3', 'RI' ROUTINES
3053 ; INPUT: B CONTAINS THE COMMAND
3054 ; OUTPUT:
3055 ; MODIFIED: A,FLAGS
3056 ; STACK USAGE:
3057 PIOCUM:
FFE4 3E0D          3058          MVI     A,DISABL         ; BLOCK ALL INTERRUPTS
FFE6 D3FF          3059          OUT     CPUC
3060 PIOZZZ:
FFE8 DBF9          3061          IN       PIOS                ; INPUT DBB STATUS
FFEA E607          3062          ANI     F0 OR IBF OR OBF; TEST FOR SLAVE PROCESSOR IDLE
FFEC C2E8FF       3063          JNZ     PIOZZZ             ; LOOP UNTIL IT IS IDLE
FFEF 78           3064          MOV     A,B                ; LOAD THE COMMAND
FFF0 D3F9          3065          OUT     PIOC                ; OUTPUT THE COMMAND TO THE PIO CONTROL PORT
FFF2 3E05          3066          MVI     A,ENABL           ; ENABLE INTERRUPTS
FFF4 D3FF          3067          OUT     CPUC
FFF6 C9           3068          RET
3069 ;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
FFFF          3070          ORG     0FFFFH
FFFF 6C          3071          MNCISM: DB    06CH                ; CHKSUM MONITOR TO 01EH

```

```

LOC OBJ          LINE          SOURCE STATEMENT
FFFE 00          3072          DB          00          ; UNUSED BYTE
FFFF 01          3073          DB          01          ; 0, IF SERIES I MONITOR
3074          ; 1, IF SERIES II MONITOR
3075 ; *-----*
3076 ;
3077 ; END OF PROGRAM
3078 ;
3079 ; *-----*
3080          END
  
```

PUBLIC SYMBOLS

EXTERNAL SYMBOLS

USER SYMBOLS

@USER	A	FC8C	ACHRM	A	007F	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	AS0	A	F8BE	AS1	A	F8CD	AS2	A	F8E5
AS3	A	F8F2	ASSIGN	A	F8B6	B0110	A	02BA	B2400	A	0020	B9600	A	0007	BASE	A	F800	BATCH	A	0002
BBASE	A	E800	BCDC	A	0001	BDLY	A	EA23	BDLY1	A	EA25	BEGIN	A	F800	BLK	A	FC93	BLOC	A	EACB
BOVROF	A	0001	BOVRON	A	0009	BREAK	A	FC07	BS0	A	E806	BS1	A	E83B	BS10	A	E913	BS11	A	E920
BS12	A	E94E	BS13	A	E959	BS14	A	E97B	BS1X	A	E84F	BS2	A	E862	BS3	A	E86A	BS4	A	E8AC
BS5	A	E8C0	BS6	A	E8C6	BS7	A	E8DC	BS8	A	E8E2	BS9	A	E8EA	BSX1	A	E98E	BSX10	A	EA18
BSX2	A	E9A4	BSX3	A	E9B2	BSX4	A	E9BD	BSX5	A	E9CC	BSX6	A	E9EA	BSX8	A	E9FE	BSX9	A	EA0D
BTCKSM	A	EA64	BTDGOF	A	0004	BTDGON	A	000C	BYTE	A	FDD8	CCRT	A	0001	CI	A	FBBE	CI0	A	FBD0
CI1	A	FBE1	CI2	A	FBEB	CI3	A	FBEC	CI4	A	FBFD	CILOC	A	EAE8	CL5	A	0000	CL6	A	0004
CL7	A	0008	CL8	A	000C	CLERR	A	0010	CLOC	A	EACA	CMSK	A	00FC	CNOTD	A	0008	CO	A	FC9F
CO0	A	FCB2	COLOC	A	EAE8	COM	A	FC95	COMC	A	FCDE	COMD	A	0025	CONC	A	00C1	CONI	A	00C0
CONO	A	00C0	CONS	A	00C1	CONV	A	FDF4	COP	A	F809	CPUC	A	00FF	CPUS	A	00FE	CR	A	00D0
CRLF	A	FDFE	CRTC	A	0010	CRTOT1	A	FCCA	CRTOT2	A	FCD5	CRTOUT	A	FCBE	CRTS	A	0011	CS0	A	FD53
CS1	A	FD6B	CS2	A	FD74	CS3	A	FD79	CSLOC	A	EAFD	CSMEM	A	0008	CSTS	A	FD44	CTBL	A	F882
CTR0P	A	00F0	CTR0S	A	0000	CTR1P	A	00F1	CTR1S	A	0040	CTR2P	A	00F2	CTR2S	A	0080	CTTY	A	0000
CUSE	A	0003	DADR	A	FE07	DATE	A	0103	DBYTE	A	FE0C	DECHO	A	0007	DELAY	A	FE1E	DI0	A	F938
DI1	A	F93E	DI2	A	F956	DIAGBT	A	EB03	DIAGMN	A	EB00	DISABL	A	000D	DISAXP	A	0000	DISP	A	F933
DLOC	A	EAC9	DLY1	A	FE20	DPRNT	A	0008	DREG	A	FE25	DSR	A	0080	DSTAT	A	0003	DSTS	A	0078
DTR	A	0002	ELOC	A	EAC8	ENABL	A	0005	ENAXP	A	0008	ENDX	A	EB00	ENHM	A	0080	EOF	A	F95F
EOI	A	0020	ERESET	A	0001	ERMSG	A	EA56	ERROR	A	F847	ETX	A	0003	EX0	A	FE46	EXIT	A	EAD2
EXPR	A	FE39	F0	A	0004	FALSE	A	0000	FDOC	A	0004	FI0	A	F984	FILL	A	F97D	FLOC	A	EACE
FRDY	A	0001	FSTOP	A	00E7	FSTP	A	00F7	GO0	A	F9A4	G01	A	F9AA	GO2	A	F9BA	GO3	A	F9C2
GO4	A	F9D1	GOTO	A	F98C	HEXN	A	F9D5	HI	A	007A	HILO	A	FE4C	HLOC	A	EADD	HMSK	A	00FF
HXD	A	FE64	IBF	A	0002	ICFG	A	0041	ICNP	A	0041	ICRTI	A	0020	ICRTO	A	0010	ICW1	A	0012
ICW2	A	0000	IICP0	A	00FB	IICP1	A	00FA	ILOC	A	EACD	ILPT	A	0040	INIT	A	E803	INITIO	A	0006
INT0	A	0001	INT1	A	0002	INT2	A	0004	INT3	A	0008	INT4	A	0010	INT5	A	0020	INT6	A	0040
INT7	A	0080	INTA	A	0000	IOBYT	A	0003	IOCC	A	00C1	IOCCOM	A	FFA6	IOCDP1	A	F821	IOCDP2	A	F844
IOCDR1	A	FF7F	IOCDR2	A	FF94	IOCHK	A	FD83	IOCI	A	00C0	IOCO	A	00C0	IOCP0	A	00FB	IOCP1	A	00FA
IOCS	A	00C1	IOCXXX	A	FF82	IOCYYY	A	FF97	IOCZZZ	A	FFAA	IODEF	A	FD94	IOPB	A	EA34	IOSSET	A	FD87
IPTP	A	0004	IPTR	A	0008	ITCP	A	00F3	ITIMO	A	00FF	ITTYI	A	0002	ITTYO	A	0001	KEYC	A	0012
KINT	A	0014	KRDY	A	0001	KSTS	A	0013	L1LOC	A	EAFD	LADR	A	FE56	LBMK	A	00FF	LBYTE	A	FESB
LCRLF	A	FE6A	LCRT	A	0040	LCT	A	001A	LCTR	A	0000	LERM	A	000E	LF	A	000A	LLOC	A	EADC
LLPT	A	0080	LMSK	A	003F	LO	A	FD1E	LOM	A	FD14	LOWW	A	0079	LP0	A	FD33	LPTC	A	0014
LPTRY	A	0001	LSTC	A	0015	LSTE	A	0040	LTBL	A	F903	LTTY	A	0000	LUSE	A	00C0	LVER	A	001B
MEMCHK	A	FD8C	MEMTOP	A	0004	MENB	A	0080	MLP	A	E902	MNCKSM	A	FFFD	MODE0	A	0000	MODE1	A	0002
MODE2	A	0004	MODE3	A	0006	MODE4	A	0008	MODE5	A	000A	MOVBOT	A	0002	MOVE	A	F9F0	MV0	A	F9F7

NI0	A FEA6	NIBBLE	A FE98	NLEADX	A FA0B	NREGS	A 000C	NU0	A FA09	NULL	A FA01	OBF	A 0001
OCW3	A 000B	ONEMS	A 0070	OPCPL	A 0004	PILOC	A EAF4	P2C	A FEC8	P2LOC	A EAF7	PA0	A FE7A
PA1	A FE7D	PA2	A FE90	PACIFY	A 0000	PADR	A FEAA	PARAM	A FE74	PARML	A 0004	PBYTE	A FFAF
PCHK	A FEC5	PCOMP	A 0002	PENB	A 0010	PEVEN	A 0020	PGRDY	A 0001	PIOC	A 00F9	PIOCOM	A FFE4
PIODR1	A FFB5	PIODR2	A FFB8	PIODR3	A FCCE	PIODR4	A FFD1	PIOI	A 00F8	PIOO	A 00F8	PIOS	A 00F9
PIOZZZ	A FFE8	PLOC	A EAE1	PMSK	A 00CF	PNIB	A 0010	PO	A FCE9	PO0	A FCF7	PO1	A FD08
POC	A FCE5	PPTP	A 0010	PRTM	A EA2A	PSOCK	A 0020	PSTC	A 0013	PTPRY	A 0001	PTRADV	A 0040
PTRDY	A 0001	PTRREV	A 0060	PTTY	A 0000	PUNC	A 0012	PUSE1	A 0020	PUSE2	A 0030	Q0	A FA21
Q1	A FA3D	Q2	A FA48	QUERY	A FA14	R16X	A 0002	R1LOC	A EAE6	R1X	A 0001	R2LOC	A EAF1
R64X	A 0003	RADCT	A 0028	RDBC	A 0019	RDBCC	A 001A	RDRC	A 0010	RDSTS	A 001C	READ	A FA52
RED0	A FA59	RED1	A FA7B	RED2	A FA93	RED3	A FA9E	RED4	A FAB3	RESET	A 0000	RESTAR	A FED4
RFR	A 0020	RI	A FC0F	RI0	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 0010
RLLM	A 0030	RLMB	A 0020	RMSK	A 00F3	ROV	A 0010	RPAR	A 0008	RPPC	A 0017	RPSTC	A 0018
RPTR	A 0004	RRDY	A 0002	RRSTS	A 001B	RST0	A FEF7	RST1	A FF25	RST2	A FF42	RST3	A FF50
RSTA	A FF13	RSTB	A FF20	RSTC	A 0011	RSTS	A 007B	RTCC	A 04CD	RTOCT	A 00FA	RTS	A 0020
RTTY	A 0000	RUSE1	A 0008	RUSE2	A 000C	RXEN	A 0004	SBCH	A 0008	SICP0	A 00FD	SICP1	A 00FC
SINT	A 000A	SLOC	A EAD1	SOCPO	A 00FD	SOCPL	A 00FC	SRQ	A 0006	SRQACK	A 0005	SRQDAK	A 0004
ST1	A 0040	ST15	A 0080	ST2	A 00C0	START	A F855	START0	A F851	SU0	A FAC3	SU1	A FAD9
SUBS	A FABF	SYNC	A 0000	SYND	A 0040	SYSTAT	A 0002	TADV	A 0027	TI	A FF61	TLOC	A EAE2
TOS	A EAC8	TOUT	A 00FA	TRAM	A 0009	TRDY	A 0001	TRK0	A 3000	TRKL	A 0D00	TRUE	A FFFF
TTYC	A 00F5	TTYI	A 00F4	TTYIN	A FBC6	TTYO	A 00F4	TTYOUT	A FCA7	TTYS	A 00F5	TXBE	A 0004
TXEN	A 0001	UC	A FF76	UCI	A 0000	UCO	A 0001	UCS	A 0007	UI	A FDAD	UL1	A 0006
UO	A FD8E	UP1	A 0004	UP2	A 0005	UPPS	A FDCE	UR1	A 0002	UR2	A 0003	USCC	A 00F7
USCI	A 00F6	USCO	A 00F6	USCS	A 00F7	USER	A EAC0	USRST	A 0040	VER	A 000D	VERH	A 0013
VERS	A EA3B	WDBC	A 0017	WDBCC	A 0018	WPBC	A 0015	WPBCC	A 0016	WPPC	A 0016	WR0	A FAE5
WR1	A FAED	WR2	A FAF8	WR3	A FB07	WRITE	A FADD	X	A FB26	X0	A FB31	X1	A FB3F
X2	A FB42	X3	A FB5F	X4	A FB60	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#							











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#					
PTY	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		

RI0	1912#	1915						
RI1	1919#	1922						
RI2	1926#	1932						
RI3	1933#	1967						
RI4	1929	1937#						
RI4B	1936	1940#						
RI5	1904	1950#						
RI6	1959#	1966						
RI7	1963	1968#						
RI8	1952	1977#						
RIX	1523	2390	2397	2888#				
RLLB	201#							
RLLM	202#	455	542	553				
RLMB	200#							
RMSK	335#	1182	1903					
ROV	168#							
RPAR	167#							
RPPC	273#	2324						
RPSTC	274#	2369						
RPTR	346#	1202	1951					
RRDY	165#	732	739	1830	1853	1928	2194	2213
RRSTS	244#	685						
RST0	2782#	2793						
RST1	2815	2826	2836#					
RST2	2861#	2879						
RST3	2871	2876#						
RSTA	2812	2816#						
RSTB	2823	2827#						
RSTC	267#	1960						
RSTS	314#	771						
RTCC	390#	457						
RTOCT	176#	1925						
RTS	158#	177	178	551	562			
RTTY	345#	1201						
RUSE1	347#	1203	1979					
RUSE2	348#	1204						
RXEN	155#	177	178	551	562			
SBCH	156#							
SICP0	119#	446						
SICP1	120#	449						
SINT	298#							
SLOC	527	884#	1786					
SOCP0	121#	2829						
SOCP1	122#	452	891	2759	2766			
SRQ	291#							
SRQACK	290#							
SRQDAK	289#							
ST1	149#							
ST15	148#							
ST2	147#	549	560					
START	1039#	1046	1050	2880				
START0	973	1036#						
SU0	1627#	1642						
SU1	1634	1640#						
SUBS	1098	1624#						
SYNC	140#							
SYND	170#							



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	BOORG 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 0000001B ;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	
00F8		54	PIOI EQU 0F8H ;PIO INPUT DATA (FROM DBB) PORT
00F8		55	PIOO EQU 0F8H ;PIO OUTPUT DATA (TO DBB) PORT
00F9		56	PIOS EQU 0F9H ;PIO STATUS PORT
00F9		57	PIOC EQU 0F9H ;PIO COMMAND PORT
		58	
0020		59	INT5 EQU 00100000B ;INTERRUPT LEVEL 5
0040		60	INT6 EQU 01000000B ;INTERRUPT LEVEL 6
0080		61	INT7 EQU 10000000B ;INTERRUPT LEVEL 7
		62	
F809		63	CO EQU 0F809H ;CO MONITOR FUNCTION
F81B		64	MEMCHK EQU 0F81BH ;MEMCHK MONITOR FUNCTION
		65	
		66	
		67	
		68	
		69	; GLOBALS
		70	
		71	
0010		72	FFLAG EQU 010H ;MAJOR TEST FAILURE FLAG
		73	; 0 = NO FAILURES IN TEST
		74	; 0FFH = TEST HAS FAILED
0011		75	TOFLAG EQU 011H ;TIMEOUT FLAG
		76	; 0 = NO TIMEOUT
		77	; 0FFH = TIMEOUT HAS OCCURRED
		78	
		79	
		80	
		81	
		82	; ENTRY POINTS
		83	
		84	
EB00		85	ORG 0EB00H ;BEGINNING OF DIAGNOSTIC
EB00 C324EB		86	JMP MIMODE ;MONITOR'S ENTRY POINT
EB03 C306EB		87	JMP BIMODE ;BOOT ENTRY POINT
		88	
		89	\$ EJECT

LOC	OBJ	SEQ	SOURCE STATEMENT
		90	;;; BIMODE - BOOT INVOKED MODE CONTROL ROUTINE
		91	
		92	
EB06	CD71ED	93	BIMODE: CALL INIT ;SAVE ENVIRONMENT
EB09	CD67EB	94	CALL CHKSUM ;CHECKSUM ROMS
		95	
EB0C	0E55	96	MVI C,55H ;CHECK FOR IOC PRESENCE
EB0E	CDE6EB	97	CALL IOCDRA
EB11	211100	98	LXI H,TOFLAG ;TEST TIME OUT FLAG
EB14	B6	99	ORA M ;AND RESULT FLAG
EB15	CC9FEB	100	CZ IOCTST ;RUN THE IOC TEST IF IOC PRESENT
EB18	3E00	101	MVI A,0 ;RESET TIMEOUT FLAG
EB1A	321100	102	STA TOFLAG
		103	
EB1D	CD2DEC	104	CALL PIOTST ;RUN THE PIO TEST
		105	
EB20	CDA8ED	106	CALL RESTOR ;RESTORE THE ENVIRONMENT
EB23	C9	107	RET ;RETURN TO THE BOOT
		108	
		109	
		110	
		111	
		112	;;; MIMODE - MONITOR INVOKED MODE CONTROL ROUTINE
		113	
		114	
EB24	CD71ED	115	MIMODE: CALL INIT ;SAVE ENVIRONMENT
EB27	014CEE	116	LXI B,SIGNON ;PRINT SIGN ON MESSAGE
EB2A	CD9EED	117	CALL PRINTL
		118	
		119	; CHECKSUM ROMS
		120	
EB2D	0180EE	121	LXI B,MIM1 ;'CHECKSUM TEST' MESSAGE
EB30	CDBFED	122	CALL SETUP ;PRINT MESSAGE AND INITIALIZE FFLAG
EB33	CD67EB	123	CALL CHKSUM ;CHECKSUM ROMS
EB36	CD5AED	124	CALL FINISH ;CHECK FFLAG
		125	
		126	; TEST IOC
		127	
EB39	018AEE	128	LXI B,MIM2 ;'IOC TEST' MESSAGE
EB3C	CDBFED	129	CALL SETUP ;PRINT MESSAGE AND INITIALIZE FFLAG
EB3F	CD9FEB	130	CALL IOCTST ;TEST IOC
EB42	CD5AED	131	CALL FINISH ;CHECK FFLAG
		132	
		133	; TEST PIO
		134	
EB45	018EEE	135	LXI B,MIM3 ;'PIO TEST' MESSAGE
EB48	CDBFED	136	CALL SETUP ;PRINT MESSAGE AND INITIALIZE FFLAG
EB4B	CD2DEC	137	CALL PIOTST ;TEST PIO
EB4E	CD5AED	138	CALL FINISH ;CHECK FFLAG
		139	
		140	; TEST RAM
		141	
EB51	0192EE	142	LXI B,MIM4 ;'RAM TEST' MESSAGE
EB54	CDBFED	143	CALL SETUP ;PRINT MESSAGE AND INITIALIZE FFLAG
EB57	CDBBEC	144	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	2100F8	160	CHKSUM: LXI H,MONORG ;SET UP TO CHECKSUM MONITOR
EB6A	110008	161	LXI D,MONLEN
EB6D	3E1E	162	MVI A,MONCHK
EB6F	CD8AEB	163	CALL SUM ;CHECKSUM MONITOR
EB72	0196EE	164	LXI B,CHKM1 ;'MONITOR CHECKSUM' MESSAGE
EB75	CDDBED	165	CALL TEST
		166	
EB78	2100E8	167	LXI H,BOOORG ;SET UP TO CHECKSUM BOOT
EB7B	110008	168	LXI D,BOOLEN
EB7E	3E55	169	MVI A,BOOCHK
EB80	CD8AEB	170	CALL SUM ;CHECKSUM BOOT
EB83	01A7EE	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	;
		180	;
		181	PARAMETERS:
		182	HL = ORIGIN OF ROM TO BE CHECKSUMMED
		183	DE = LENGTH OF ROM
		184	A = EXPECTED CHECKSUM
		185	;
		186	RETURNS:
		187	A = SUCCESS FLAG
		188	0 = CHECKSUM OK
		189	OFFH = CHECKSUM FAILED
		190	;
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	200	INX H ;STEP TO NEXT WORD
EB95	1B	201	DCX D ;DECREMENT COUNT
EB96	C38CEB	202	JMP SUM1 ;LOOP
		203	
EB99	78	204	SUM2: MOV A,B ;GET SUM
EB9A	B7	205	ORA A ;TEST FOR ZERO SUM
EB9B	C8	206	RZ ;IF ZERO, RETURN SUCCESS
		207	
EB9C	3EFF	208	MVI A,0FFH ;RETURN FAILURE
EB9E	C9	209	RET
		210	
		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
		220	
EB9F	0E55	221	MVI C,55H ;TRY TO ECHO A 55H
EBA1	CDE6EB	222	CALL IOCDRA
EBA4	211100	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	01B5EE	226	LXI B,IOCM1 ;FAILURE MESSAGE
EBAE	CDDBED	227	CALL TEST
EBB1	C9	228	RET ;DO NOT DO OTHER TESTS IF NOT PRESENT
		229	
		230	; IOC CHECKSUM TEST
		231	
EBB2	0608	232	IOC1: MVI B,CSMEM ;CHECKSUM COMMAND
EBB4	CDF7EB	233	CALL IOCDRB
EBB7	01D2EE	234	LXI B,IOCM2
EBBA	CDDBED	235	CALL TEST
		236	
		237	; IOC RAM TEST
		238	
EBBD	0609	239	MVI B,TRAM ;TEST RAM COMMAND
EBBF	CDF7EB	240	CALL IOCDRB
EBC2	01DFEE	241	LXI B,IOCM3
EBC5	CDDBED	242	CALL TEST
		243	
		244	; IOC INTERRUPT TEST
		245	
EBC8	0EBF	246	MVI C,NOT INT6 ;SET UP MASKS
EBCA	CD40EE	247	CALL SETINT
EBCD	0606	248	MVI B,SRQ ;TURN ON IOC INTERRUPT
EBCF	CDFEEB	249	CALL IOCDRC
EBD2	CD13EE	250	CALL CHKINT ;CHECK INTERRUPTS
EBD5	F5	251	PUSH PSW ;SAVE RESULT
EBD6	0605	252	MVI B,SRQACK ;RESET IOC INTERRUPT
EBD8	CDFEEB	253	CALL IOCDRC
EBDB	CD2EEE	254	CALL RESET ;RESTORE INTERRUPTS TO NORMAL
		255	
EBDE	F1	256	POP PSW ;GET RESULT FLAG
EBDF	01E7EE	257	LXI B,IOCM4 ;LOAD MESSAGE POINTER
EBE2	CDDBED	258	CALL TEST
EBE5	C9	259	RET
		260	
		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:
		270 ;	C = DATA TO BE ECHOED
		271 ;	
		272 ;	RETURNS:
		273 ;	A = SUCCESS FLAG
		274 ;	0 = PASSED
		275 ;	0FFH - FAILED
		276 ;	
		277 ;	CALLS:
		278 ;	IOCCOD
		279 ;	IOCDID
		280 ;	IOCDOD
		281	
		282	
EBE6	0607	283	IOCDRA: MVI B,DECHO ;PUT OUT DATA ECHO COMMAND
EBE8	CDFEEB	284	CALL IOCCOD
EBEB	CD0FEC	285	CALL IOCDOD
EBEE	CD07EC	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	290	MVI A,0FFH ;OTHERWISE,RETURN FAILURE
EBF6	C9	291	RET
		292	
		293	
		294	
		295	
		296 ;;;	IOCDRB - ISSUE COMMAND AND READ DATA
		297 ;	
		298 ;	PARAMETER:
		299 ;	B = COMMAND
		300 ;	
		301 ;	RETURNS:
		302 ;	A = DATA
		303 ;	
		304 ;	CALLS:
		305 ;	IOCCOD
		306 ;	IOCDID
		307	
		308	
EBF7	CDFEEB	309	IOCDRB: CALL IOCCOD ;PUT OUT COMMAND
EBFA	CD07EC	310	CALL IOCDID ;READ DATA
EBFD	C9	311	RET
		312	
		313	
		314	
		315	
		316 ;;;	IOCDRC - ISSUE COMMAND
		317 ;	
		318 ;	PARAMETER:
		319 ;	B = COMMAND
		320 ;	
		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	
		330	;; IOCCOD - IOC COMMAND OUT DRIVER
		331	;
		332	PARAMETER:
		333	B = COMMAND
		334	;
		335	MODIFIES A,B,E,HL
		336	
EBFE	1E00	337	IOCCOD: MVI E,0 ;TEST FOR ZERO STATUS
EC00	CD18EC	338	CALL IOCWT ;WAIT FOR STATUS OR TIMEOUT
EC03	78	339	MOV A,B ;OUTPUT COMMAND
EC04	D3C1	340	OUT IOCC
EC06	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
		350	;
		351	RETURNS:
		352	A = DATA
		353	;
		354	MODIFIES A,E,HL
		355	
EC07	1E01	356	IOCDID: MVI E,0BF ;TEST FOR OBF STATUS
EC09	CD18EC	357	CALL IOCWT
EC0C	DBC0	358	IN IOCI ;READ DATA
EC0E	C9	359	RET
		360	
		361	
		362	
		363	
		364	;; IOCDOD - IOC DATA OUT DRIVER
		365	;
		366	PARAMETER:
		367	C = DATA
		368	;
		369	MODIFIES A,E,HL
		370	
EC0F	1E00	371	IOCDOD: MVI E,0 ;TEST FOR ZERO STATUS
EC11	CD18EC	372	CALL IOCWT ;WAIT FOR READY STATUS
EC14	79	373	MOV A,C ;WRITE DATA
EC15	D3C0	374	OUT IOCC
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
		409	;;; PIOTST - PIO TEST
		410	
		411	
		412	PIOTST:
		413	
		414	; ECHO TEST
		415	
EC2D	0E55	416	MVI C,55H ;TRY TO ECHO A 55H
EC2F	CD74EC	417	CALL PIODRA
EC32	211100	418	LXI H,TOFLAG ;TEST TIME-OUT AND RESULT FLAGS
EC35	B6	419	ORA M
EC36	CA40EC	420	JZ PIO1 ;IF RESULT OK
EC39	01F6EE	421	LXI B,PIOM1 ;FAILURE MESSAGE
EC3C	CDBBED	422	CALL TEST
EC3F	C9	423	RET ;DO NOT DO OTHER TESTS IF NOT PRESENT
		424	
		425	; PIO CHECKSUM TEST
		426	
EC40	0608	427	PIO1: MVI B,CSMEM ;CHECKSUM COMMAND
EC42	CD85EC	428	CALL PIODRB
EC45	0109EF	429	LXI B,PIOM2
EC48	CDBBED	430	CALL TEST
		431	
		432	; PIO RAM TEST
		433	
EC4B	0609	434	MVI B,TRAM ;TEST RAM COMMAND
EC4D	CD85EC	435	CALL PIODRB
EC50	0116EF	436	LXI B,PIOM3
EC53	CDBBED	437	CALL TEST
		438	
		439	; PIO INTERRUPT TEST
		440	
EC56	0EDF	441	MVI C,NOT INT5 ;SET UP MASKS
EC58	CD40EE	442	CALL SETINT
EC5B	0606	443	MVI B,SRQ ;TURN ON PIO INTERRUPT
EC5D	CD8CEC	444	CALL PIODRC
EC60	CD13EE	445	CALL CHKINT ;CHECK INTERRUPTS
EC63	F5	446	PUSH PSW ;SAVE RESULT FLAG
EC64	0605	447	MVI B,SRQACK ;RESET PIO INTERRUPT
EC66	CD8CEC	448	CALL PIODRC
EC69	CD2EEE	449	CALL RESET ;RESTORE INTERRUPTS TO NORMAL
		450	
EC6C	F1	451	POP PSW ;GET RESULT FLAG
EC6D	011EEF	452	LXI B,PIOM4 ;LOAD MESSAGE POINTER
EC70	CDBBED	453	CALL TEST
		454	
EC73	C9	455	RET
		456	
		457	
		458	
		459	
		460	;;; 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 ;	0 = PASSED
		471 ;	0FFH - FAILED
		472 ;	
		473 ;	CALLS:
		474 ;	PIOCOD
		475 ;	PIODID
		476 ;	PIODOD
		477 ;	
		478 ;	
EC74	0607	479	PIODRA: MVI B,DECHO ;PUT OUT DATA ECHO COMMAND
EC76	CD8CEC	480	CALL PIOCOD
EC79	CD9DEC	481	CALL PIODOD
EC7C	CD95EC	482	CALL PIODID ;READ BACK DATA
EC7F	2F	483	CMA ;PIO RETURNS COMPLEMENTED DATA
EC80	91	484	SUB C ;CHECK IF ECHO EQUALS ORIGINAL
EC81	C8	485	RZ ;RETURN IF OK
EC82	3EFF	486	MVI A,0FFH ;OTHERWISE,RETURN FAILURE
EC84	C9	487	RET
		488 ;	
		489 ;	
		490 ;	
		491 ;	
		492 ;;;	PIODRB - ISSUE COMMAND AND READ DATA
		493 ;	
		494 ;	PARAMETER:
		495 ;	B = COMMAND
		496 ;	
		497 ;	RETURNS:
		498 ;	A = DATA
		499 ;	
		500 ;	CALLS:
		501 ;	PIOCOD
		502 ;	PIODID
		503 ;	
		504 ;	
EC85	CD8CEC	505	PIODRB: CALL PIOCOD ;PUT OUT COMMAND
EC88	CD95EC	506	CALL PIODID ;READ DATA
EC8B	C9	507	RET
		508 ;	
		509 ;	
		510 ;	
		511 ;	
		512 ;;;	PIODRC - ISSUE COMMAND
		513 ;	
		514 ;	PARAMETER:
		515 ;	B = COMMAND
		516 ;	
		517 ;	CALLS:
		518 ;	PIOCOD

LOC	OBJ	SEQ	SOURCE STATEMENT
		519	
		520	;PIODRC EQU PIOCOD ;THIS IS JUST THE COMMAND OUT DRIVER
		521	;ACTUAL DEFINITION FOLLOWS PIOCOD
		522	
		523	
		524	
		525	
		526	;; PIOCOD - PIO COMMAND OUT DRIVER
		527	;
		528	PARAMETER:
		529	B = COMMAND
		530	;
		531	MODIFIES A,B,E,HL
		532	
EC8C	1E00	533	PIOCOD: MVI E,0 ;WAIT FOR 0 STATUS
EC8E	CDA6EC	534	CALL PIOWT ;CALL WAIT ROUTINE
EC91	78	535	MOV A,B ;OUTPUT COMMAND
EC92	D3F9	536	OUT PIOC
EC94	C9	537	RET
		538	
		539	
EC8C		540	PIODRC EQU PIOCOD ;DEFINITION HERE DUE TO FORWARD REFERENCE
		541	
		542	
		543	
		544	
		545	;; PIODID - PIO DATA IN DRIVER
		546	;
		547	RETURNS:
		548	A = DATA
		549	;
		550	MODIFIES A,E,HL
		551	
EC95	1E01	552	PIODID: MVI E,0BF ;WAIT FOR OBF STATUS
EC97	CDA6EC	553	CALL PIOWT ;WAIT
EC9A	DBF8	554	IN PIOI ;READ DATA
EC9C	C9	555	RET
		556	
		557	
		558	
		559	
		560	;; PIODOD - PIO DATA OUT DRIVER
		561	;
		562	PARAMETER:
		563	C = DATA
		564	;
		565	MODIFIES A,E,HL
		566	
EC9D	1E00	567	PIODOD: MVI E,0 ;WAIT FOR 0 STATUS
EC9F	CDA6EC	568	CALL PIOWT
ECA2	79	569	MOV A,C ;WRITE DATA
ECA3	D3F8	570	OUT PIOO
ECA5	C9	571	RET
		572	
		573	

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:
		580 ;	E = STATUS TO WAIT FOR
		581 ;	
		582 ;	RETURNS:
		583 ;	TOFLAG UPDATED TO 0FFH IF A TIMEOUT OCCURS
		584 ;	
		585 ;	MODIFIES:
		586 ;	TOFLAG,HL
		587	
		588	
ECA6	210010	589	PIOWT: LXI H,01000H ;WAIT COUNT
ECA9	DBF9	590	PIOWT1: IN PIOS ;CHECK STATUS
ECAB	E607	591	ANI F0 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
ECB0	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,0FFH ;TIMED OUT; UPDATE TOFLAG
ECB7	321100	599	STA TOFLAG
ECBA	C9	600	RET
		601	
		602	
		603	
		604	
		605 \$	TITLE ('RAMTST - RAM TEST')
		606 \$	EJECT

LOC	OBJ	SEQ	SOURCE STATEMENT
		607 ;;;	RAMTST - TAM TEST
		608	
		609	
ECBB	211200	610	RAMTST: LXI H,012H ;FIRST WORD TO FILL
ECBE	11FFE7	611	LXI D,0E7FFH ;BOTTOM OF BOOT ROM
ECC1	CD2DED	612	CALL FILL
ECC4	2100F0	613	LXI H,0F000H ;TOP OF BOOT/DIAGNOSTIC ROM
ECC7	11FFF7	614	LXI D,0F7FFH ;BOTTOM OF MONITOR ROM
ECCA	CD2DED	615	CALL FILL
		616	
		617 ;	TEST BANK 0-32K
		618	
ECCD	211200	619	LXI H,012H ;FIRST WORD TO TEST
ECD0	11FF7F	620	LXI D,07FFFH ;LAST WORD TO TEST
ECD3	CD08ED	621	CALL CHECK
ECD6	012DEF	622	LXI B,RAMM1 ;'BANK 0-23K FAILURE' MESSAGE
ECD9	CDBBED	623	CALL TEST
		624	
		625 ;	TEST BANK 32-48K
		626	
ECDC	210080	627	LXI H,08000H ;FIRST WORD TO TEST
ECDF	11FFBF	628	LXI D,0BFFFH ;LAST WORD TO TEST
ECE2	CD08ED	629	CALL CHECK
ECE5	013CEF	630	LXI B,RAMM2 ;'BANK 32-48K FAILURE' MESSAGE
ECE8	CDBBED	631	CALL TEST
		632	
		633 ;	TEST BANK 48-62K
		634	
ECEB	2100C0	635	LXI H,0C000H ;FIRST WORD TO TEST
ECEE	11FFE7	636	LXI D,0E7FFH ;BOTTOM OF BOOT/DIAGNOSTIC ROM
ECF1	CD08ED	637	CALL CHECK
ECF4	B7	638	ORA A ;TEST FOR FAILURE
ECF5	C201ED	639	JNZ RAM1 ;IF A FAILURE
ECF8	2100F0	640	LXI H,0F000H ;TOP OF BOOT/DIAGNOSTIC ROM
ECFB	11FFF7	641	LXI D,0F7FFH ;BOTTOM OF MONITOR ROM
ECFE	CD08ED	642	CALL CHECK
ED01	014CEF	643	RAM1: LXI B,RAMM3 ;'BANK 48-62K FAILURE' MESSAGE
ED04	CDBBED	644	CALL TEST
		645	
ED07	C9	646	RET
		647	
		648	
		649	
		650	
		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 = 0 IF TEST SUCCESSFUL
		659 ;	A = 0FFH AT FIRST FAILURE
		660 ;	
		661 ;	CALLS:

LOC	OBJ	SEQ	SOURCE STATEMENT
		662 ;	SETLIM
		663	
		664	
ED08	CD3BED	665	CHECK: CALL SETLIM ;SET UP LIMITS TO TAKE MEMCHK INTO ACCOUNT
		666	
ED0B	7A	667	CHECK1: MOV A,D ;CHECK IF ALREADY DONE
ED0C	B3	668	ORA E
ED0D	CA27ED	669	JZ CHECK2 ;IF ALREADY DONE
		670	
ED10	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 0FFH IS THERE
ED19	C22AED	679	JNZ CHECK3
		680	
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	C30BED	688	JMP CHECK1 ;LOOP
		689	
ED27	3E00	690	CHECK2: MVI A,0 ;RETURN SUCCESS
ED29	C9	691	RET
		692	
ED2A	3EFF	693	CHECK3: MVI A,0FFH ;RETURN FAILURE
ED2C	C9	694	RET
		695	
		696	
		697	
		698	
		699 ;;;	FILL - FILL MEMORY WITH BACKGROUND, TAKING MEMCHK INTO ACCOUNT.
		700 ;	
		701 ;	FILL WILL PUT BACKGROUND INTO MEMORY STARTING AT FIRST WORD ADDRESS
		702 ;	AND ENDING AT MEMCHK OR LAST MEMORY ADDRESS, WHICHEVER IS ENCOUNTERED
		703 ;	FIRST
		704 ;	
		705 ;	PARAMETERS:
		706 ;	HL = FIRST WORD ADDRESS
		707 ;	DE = LAST WORD ADDRESS
		708 ;	
		709 ;	CALLS:
		710 ;	SETLIM
		711	
		712	
ED2D	CD3BED	713	FILL: CALL SETLIM
		714	
ED30	7A	715	FILL1: MOV A,D ;CHECK IF COUNT=0
ED31	B3	716	ORA E



LOC	OBJ	SEQ	SOURCE STATEMENT
ED32	C8	717	RZ
ED33	7C	718	MOV A,H ;GENERATE PATTERN
ED34	AD	719	XRA L
ED35	77	720	MOV M,A ;STORE PATTERN
ED36	23	721	INX H ;INCREMENT POINTER TO NEXT LOCATION
ED37	1B	722	DCX D ;DECREMENT COUNTER
ED38	C330ED	723	JMP FILL1 ;LOOP
		724	
		725	
		726	
		727	
		728	;;; SETLIM - SET LIMITS
		729	;
		730	;
		731	;
		732	;
		733	;
		734	;
		735	;
		736	;
		737	;
		738	;
		739	;
		740	;
		741	;
		742	;
		743	;
		744	;
		745	;
		746	;
		747	;
ED3B	E5	748	SETLIM: PUSH H ;SAVE HL AND DE DURING CALL TO MEMCHK
ED3C	D5	749	PUSH D
		750	
ED3D	CD1BF8	751	CALL MEMCHK
ED40	4F	752	MOV C,A ;BC=MEMCHK
		753	
ED41	D1	754	POP D ;RESTORE HL AND DE
ED42	E1	755	POP H
		756	
ED43	95	757	SUB L ;SUBTRACT FWA FROM MEMCHK
ED44	78	758	MOV A,B
ED45	9C	759	SBB H
ED46	DA52ED	760	JC SETLM1 ;JUMP IF MEMCHK < FWA
		761	
ED49	7B	762	MOV A,E ;SUBTRACT MEMCHK FROM LWA
ED4A	91	763	SUB C
ED4B	7A	764	MOV A,D
ED4C	98	765	SBB B
ED4D	DA52ED	766	JC SETLM1 ;JUMP IF LWA < MEMCHK
		767	
ED50	50	768	MOV D,B ;MEMCHK IS WITHIN RANGE; USE IT AS LWA
ED51	59	769	MOV E,C
		770	
ED52	7B	771	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 0
		786	;
		787	ACCESSES:
		788	FFLAG
		789	;
		790	CALLS:
		791	PRINT
		792	;
		793	;
ED5A	3A1000	794	FINISH: LDA FFLAG ;TEST FFLAG
ED5D	B7	795	ORA A
ED5E	C0	796	RNZ ;RETURN IF TEST FAILED
ED5F	0166ED	797	LXI B,FINA ;PRINT ' -- PASSED' MESSAGE
ED62	CD9EED	798	CALL PRINTL
ED65	C9	799	RET
		800	;
		801	;
ED66	202D2D20	802	FINA: DB ' -- PASSED',0
ED6A	50415353		
ED6E	4544		
ED70	00		
		803	
		804	
		805	
		806	
		807	;;; INIT - SAVE INVIRONMENT ON STACK
		808	;
		809	INIT SAVES THE INTERRUPT MASKS OF BOTH 8257'S AND THE
		810	CONTENTS OF FFLAG. IT IS INTENDED TO BE USED WITH RESTOR,
		811	AND MUST BE CALLED AT THE SAME NEST LEVEL AS RESTOR
		812	;
ED71	D1	813	INIT: POP D ;SAVE RETURN SINCE STACK TO BE MODIFIED
ED72	F5	814	PUSH PSW ;SAVE A AND FLAGS
ED73	2A3800	815	LHLD INT7V ;SAVE INTERRUPT 7 VECTOR
ED76	E5	816	PUSH H
ED77	2A3A00	817	LHLD INT7V+2 ;SAVE REST OF VECTOR
ED7A	E5	818	PUSH H
ED7B	2A1000	819	LHLD 010H ;SAVE TOFLAG AND FFLAG
ED7E	E5	820	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	3E00	826	MVI A,0 ;INITIALIZE FFLAG AND TOFLAG
ED88	321000	827	STA FFLAG
ED8B	321100	828	STA TOFLAG
ED8E	D5	829	PUSH D ;RETURN
ED8F	C9	830	RET
		831	;
		832	;
		833	;
		834	;
		835	;;; PRINT - PRINT STRING
		836	;

LOC	OBJ	SEQ	SOURCE STATEMENT
		837 ;	PARAMETER:
		838 ;	BC = POINTER TO STRING TERMINATED WITH A NULL.
		839 ;	
		840 ;	CALLS:
		841 ;	CO
		842	
		843	
ED90	C5	844	PRINT: PUSH B ;SAVE POINTER ON THE STACK
		845	
ED91	E1	846	PRINT1: POP H ;LOAD POINTER INTO HL
ED92	4E	847	MOV C,M ;READ NEXT CHARACTER
ED93	79	848	MOV A,C ;PREPARE FOR TERMINATOR CHECK
ED94	B7	849	ORA A ;CHECK FOR STRING TERMINATOR
ED95	C8	850	RZ ;RETURN IF NULL
ED96	23	851	INX H ;INCREMENT POINTER
ED97	E5	852	PUSH H ;RESTORE ON STACK
ED98	CD09F8	853	CALL CO ;PRINT CHARACTER
ED9B	C391ED	854	JMP PRINT1 ;LOOP UNTIL DONE
		855	
		856	
		857	
		858	
		859 ;;;	PRINTL - PRINT MESSAGE WITH A CR-LF ADDED AT THE END
		860 ;	
		861 ;	PARAMETER:
		862 ;	BC = POINTER TO STRING TERMINATED WITH A NULL
		863 ;	
		864 ;	CALLS:
		865 ;	PRINT
		866	
		867	
ED9E	CD90ED	868	PRINTL: CALL PRINT ;PRINT ORIGINAL STRING
EDA1	0149EE	869	LXI B,CRLF ;PRINT CR-LF
EDA4	CD90ED	870	CALL PRINT
EDA7	C9	871	RET
		872	
		873	
		874	
		875	
		876 ;;;	RESTOR - RESTOR ENVIRONMENT
		877 ;	RESTOR IS THE COMPLEMENT OF INIT
		878	
		879	
EDA8	D1	880	RESTOR: POP D ;SAVE RETURN ADDR WHILE PLAYING WITH STACK
EDA9	C1	881	POP B ;READ INTERRUPT MASKS
EDAA	79	882	MOV A,C ;RESTORE SYSTEM INTERRUPT MASK
EDAB	D3FC	883	OUT SPICMR
EDAD	78	884	MOV A,B ;RESTORE IO INTERRUPT MASK
EDAE	D3FA	885	OUT IPICMR
EDB0	E1	886	POP H ;RESTORE TOFLAG AND FFLAG
EDB1	221000	887	SHLD 010H
EDB4	E1	888	POP H ;RESTORE INTERRUPT VECTOR
EDB5	223A00	889	SHLD INT7V+2
EDB8	E1	890	POP H
EDB9	223800	891	SHLD INT7V

LOC	OBJ	SEQ	SOURCE STATEMENT
EDBC	F1	892	POP PSW ;RESTORE A AND FLAGS
EDBD	D5	893	PUSH D ;RETURN
EDBE	C9	894	RET
		895	
		896	
		897	
		898	
		899	;;; SETUP - SET UP FOR TEST
		900	;
		901	;
		902	;
		903	;
		904	;
		905	;
		906	;
		907	;
		908	;
		909	;
		910	;
		911	;
		912	;
		913	;
EDBF	C5	914	SETUP: PUSH B ;SAVE MESSAGE POINTER
EDC0	01D0ED	915	LXI B,SETA ;PRINT 'TESTING '
EDC3	CD90ED	916	CALL PRINT
EDC6	C1	917	POP B ;PRINT MESSAGE
EDC7	CD90ED	918	CALL PRINT
EDCA	3E00	919	MVI A,0 ;ZERO OUT FFLAG
EDCC	321000	920	STA FFLAG
EDCF	C9	921	RET
		922	
		923	
EDD0	20205445	924	SETA: DB ' TESTING ',0
EDD4	5354494E		
EDD8	4720		
EDDA	00		
		925	
		926	
		927	
		928	
		929	;;; TEST - TEST RESULT FLAG OF A TEST
		930	;
		931	;
		932	;
		933	;
		934	;
		935	;
		936	;
		937	;
		938	;
		939	;
		940	;
		941	;
		942	;
		943	;

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 UNDESIREED 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
		1035	
		1036	
		1037	
		1038	;;; SETINT - SET UP FOR INTERRUPT TEST
		1039	;
		1040	PARAMETER:
		1041	C = MASK FOR IO PIC
		1042	;
		1043	EXIT CONDITIONS:
		1044	PIC'S SET UP SO THIS IS THE ONLY INTERRUPT ENABLEABLE
		1045	
		1046	
EE40	F3	1047	SETINT: DI
EE41	3E7F	1048	MVI A,NOT INT7 ;INITIALIZE SYSTEM PIC FOR ONLY IO INTERRUPTS
EE43	D3FC	1049	OUT SPICMR
EE45	79	1050	MOV A,C ;SET UP IO PIC
EE46	D3FA	1051	OUT IPICMR
EE48	C9	1052	RET
		1053	
		1054	
		1055	\$ TITLE ('MESSAGES')
		1056	\$ EJECT



## MESSAGES

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		
EEDF	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

BIMODE A EB06	BOOCHK A 0055	BOOLEN A 0800	BOOORG A E800	BOOSUM A 00A3	CHECK A ED08	CHECK1 A ED0B
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 F809	CR A 000D	CRLF A EE49	CSMEM A 0008	DECHO A 0007	EOI A 0020
F0 A 0004	FFLAG A 0010	FILL A ED2D	FILL1 A ED30	FINA A ED66	FINISH A ED5A	IBF A 0002
INIT A ED71	INT5 A 0020	INT6 A 0040	INT7 A 0080	INT7V A 0038	IOC1 A EBB2	IOCC A 00C1
IOCCOD A EBFE	IOCDID A EC07	IOCDOD A EC0F	IOCDRA A EBE6	IOCDRB A EBF7	IOCDRC A EBFE	IOCI A 00C0
IOCM1 A EEB5	IOCM2 A EED2	IOCM3 A EEDF	IOCM4 A EEE7	IOCO A 00C0	IOCS A 00C1	IOCTST A EB9F
IOCWT A EC18	IOCWT1 A EC1B	IPICCR A 00FB	IPICMR A 00FA	LF A 000A	MEMCHK A F81B	MIM1 A EE80
MIM2 A EE8A	MIM3 A EE8E	MIM4 A EE92	MIMODE A EB24	MONCHK A 001E	MONLEN A 0800	MONORG A F800
OBF A 0001	PIO1 A EC40	PIOC A 00F9	PIOCOD A EC8C	PIODID A EC95	PIODOD A EC9D	PIODRA A EC74
PIODRB A EC85	PIODRC A EC8C	PIOI A 00F8	PIOM1 A EEF6	PIOM2 A EF09	PIOM3 A EF16	PIOM4 A EF1E
PIOO A 00F8	PIOS A 00F9	PIOTST A EC2D	PIOWT A ECA6	PIOWT1 A ECA9	POLL A 000C	PRINT A ED90
PRINT1 A ED91	PRINTL A ED9E	RAM1 A ED01	RAMM1 A EF2D	RAMM2 A EF3C	RAMM3 A EF4C	RAMTST A ECBB
RESET A EE2E	RESTOR A EDAB	SETA A EDD0	SETINT A EE40	SETLIM A ED3B	SETLM1 A ED52	SETUP A EDBF
SGNOFF A EE71	SIGNON A EE4C	SPICCR A 00FD	SPICMR A 00FC	SRQ A 0006	SRQACK A 0005	SUM A EB8A
SUM1 A EB8C	SUM2 A EB99	TEST A EDDB	TEST1 A EDF3	TESTA A EE03	TOFLAG A 0011	TRAM A 0009

ASSEMBLY COMPLETE, NO ERRORS



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

Printed in U.S.A.