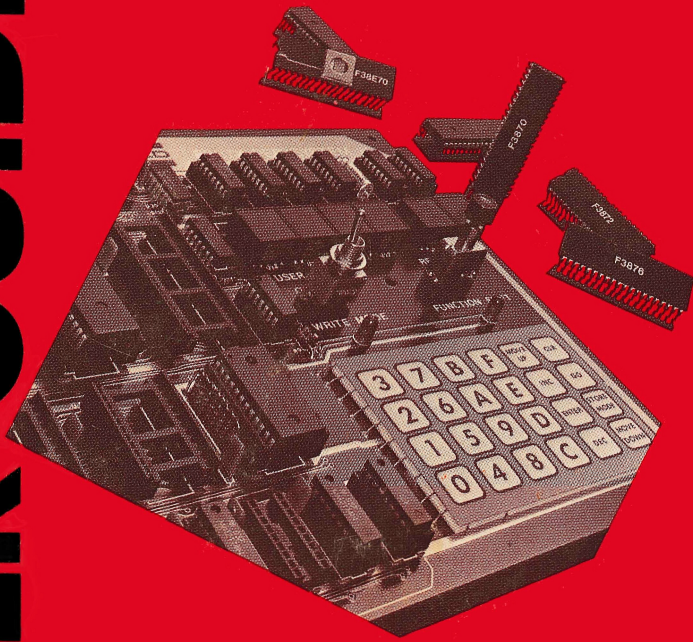


# PEPBUG USER GUIDE



**FAIRCHILD**







## TABLE OF CONTENTS

Section	Page
1.0 General Description.....	1
1.1 Electrical Specifications.....	1
2.0 PEPBUG Overview.....	1
2.1 PEPBUG Commands.....	3
3.0 Detailed Command Explanation.....	7
3.1 PEPBUG Command Usage.....	11
3.2 Input/Output.....	11
3.3 Load from Parallel Input Device.....	12
4.0 PEPBUG Subroutines.....	13
5.0 Programming Examples.....	14

### Figures

1. PEPBUG Keyboard Display Diagram.....	5
2. RAM Utilization by PEPBUG.....	6
3. Flow Diagram of Load Routine.....	12

### Appendices

Appendix A: ASCII Character Codes.....	A1
Appendix B: PEPBUG Port Assignments .....	A1
Appendix C: Formatted Tape (Loader Formats) and Checksum Calculations.....	A2 A4
Appendix D: PEPBUG Prom Punch Format.....	A5
Appendix E: PEPBUG Subroutines.....	A6
Appendix F: F2716 and F38E70 Prom Program IO Interconnect Diagram.....	A9
Keyboard/Display Diagram.....	A10
IO Ports to Keypad and TTY Interconnect Diagram.....	A11
Appendix G: PEPBUG Examples.....	A12
Appendix H: PEPBUG Program Listing.....	A15



## 1.0 General Description

A special F38T56 PSU with a debug monitor (PEPBUG) has been developed by Fairchild to provide the user with a convenient and powerful programming debug facility to aid in the development of F8 or F387X programs. The debugging program provides the user with an interactive system via a teletype terminal or via a 4 x 6 keypad.

The PEPBUG PSU is assigned memory addresses '8000'-'87FF', with entry point being '8080'. The port assignments are as follows:

I/O PORT A	'28'
I/O PORT B	'29'
Local Int. Control	'2A'
Timer	'2B'

The Interrupt address vector for the timer is '0300' and for an External Interrupt is '0380'.

## 1.1 Electrical Specifications

Since this is just a standard PSU with a special program on it, the dc and ac electrical characteristics of the F38T56 are identical to those described in the F38T56 data sheet.

## 2.0 PEPBUG Overview

PEPBUG provides the following capabilities:

- Display or Alter Memory Locations
- Display or Alter Scratchpad Registers
- Display or Alter I/O Ports
- Display or Alter Accumulator, ISAR, Status(W Register)
- Display or Alter PC0, PC1, DC0, DC1
- Load Formatted Paper Tape (FAIR-BUG or Formulator Format)
- Punch Formatted Paper Tape (Formulator Format)
- Punch Paper Tape in PROM Format ( 8-Bit Binary Format)
- Entry from Keyboard or by Program Instruction
- I/O Subroutines Available to User
- Breakpoint
- Block Memory Move
- Hexadecimal Arithmetic
- Program F38E70 or F2716 EPROM.

PEPBUG can be entered in several ways. Execution of a program instruction such as PI '8080', JMP '8080', LR,PO,Q, or PK (when Q or K contains '8080') can be used to achieve entry from another software module. Another technique is to use hardware to decode the RESET state on the ROMC lines and force the high order bit on the Data Bus to a '1' at this time.

PEPBUG will save the state of the machine upon entry and will restore it upon return to the user's program. The interrupt is disabled by PEPBUG and may be re-enabled by the user if desired. RAM at memory locations H'2B80' to H'2BE5' is used as save and work area by PEPBUG. Address 2B80 was chosen for RAM address for two reasons:

1. Keep it out of user's first 8K address space
2. Minimize change of chip select logic necessary to switch from KD-BUG to PEPBUG. KD-BUG uses RAM Bxx or 3xx.

An ideal part to use for this RAM is an F6810 128 x 8 Static RAM.

Two I/O ports (28 and 29) are available to the user when PEPBUG is not executing, as are the Timer and External Interrupt facilities.(PEPBUG does not use either of these.) During PEPBUG execution Port 28 is used for serial input/output and control functions while Port 29 is used for parallel input from a high speed paper tape reader. Assignments for Port 28 are:

<u>Bit</u>	<u>Function</u>		
7	Serial input (0 Volts= MARK)		
6	Character Ready Input (Parallel Device) (+5 V=READY)		
5	TTY or KEYBOARD IO (0 Volts = KEYPAD)		
4	Device Ready Input (Parallel Device) (+5 V=READY)		
3	Step Reader Output (Parallel Device) (0 Volts=Step)		
2-1	<u>Bit 2</u>	<u>Bit 1</u>	<u>Baud Rate</u>
	0	0	110 Baud for Serial I/O
	0	1	300 Baud for Serial I/O
	1	0	1200 Baud for Serial I/O
0	Serial Output (0 Volts=MARK)		

If Port 29 is not utilized by PEPBUG, then Pins 3, 4, and 6 of Port 28 are also available to the user. If Port 29 is used for parallel input, the parallel input data should have logic '1' corresponding to a +5 V electrical level.



Bits 1 and 2 of Port 28 are examined when PEPBUG is entered to initialize the Baud Delay Counter. If Bits 2 and 3 are at ground so that the Baud Delay count is being obtained from memory location H'2BC3', then the total delay count (time between bits for a bit serial I/O device) can be adjusted as follows:

Bit time interval=(256-count) x 36 us + 55 us  
Thus, 110 Baud requires a count of 06  
300 Baud requires a count of A4  
1200 Baud requires a count of EA

The above equation assumes a clock of 2MHz. If a different clock frequency is used the 36 usec loop time must be adjusted as well as the 55 usec overhead time. The count must then be recalculated.

## 2.1 PEPBUG Commands

When PEPBUG is entered a prompt character (?) is sent to the output device. The user then has the option of using any of the debug commands. After each execution the user is again prompted with (?). All data and input parameters are in hexadecimal notation. (C/R) following a command indicates a carriage return.

The keyboard also has a prompt which is a LED. The prompt lite is on when a command is requested, no shift is required. I.e. if PROMPT is on and key 7 is depressed the program will assume ACCumlator display and not the digit 7.

After a command key is depressed the prompt lite will go off and then the keys will be recognized as hexadecimal digits.

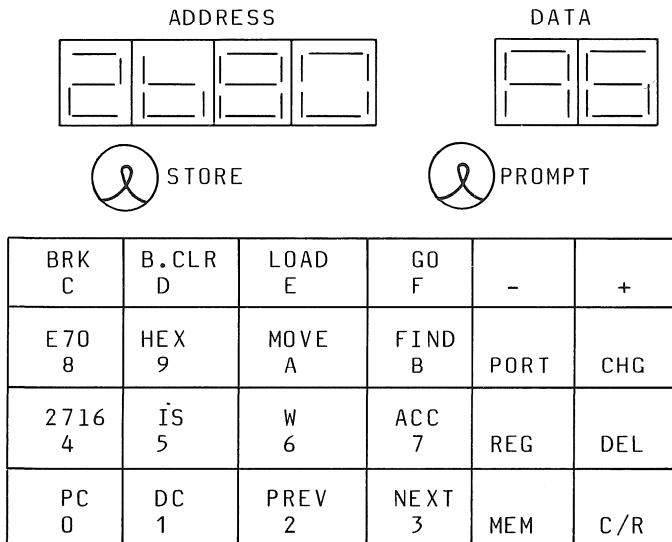
If the CHG key is depressed the STORE lite will go on and remain on until two consecutive C/R keys are detected.

The commands from the keypad are the same as they would be from the TTY. However, several functions are not useful; i.e. MO-F on the TTY would print memory locations 0-F; this keypad instruction would not be meaningful.

Command Type	Command	Function
Display	A(C/R)	Display the contents of the Accumulator
	DO(C/R)	Display the contents of DC0
	D1(C/R)	Display the contents of DC1
	I(C/R)	Display the contents of ISAR
	Mxxxx(C/R)	Display Memory Location xxxx
	Mxxxx-yyy(C/R)	Display Memory Location xxxx to yyyy
	Oxx(C/R)	Display Port xx Data
	PO(C/R)	Display the contents of PC0
	P1(C/R)	Display the contents of PC1
	Rxx(C/R)	Display the contents of Register xx
	Rxx-yy(C/R)	Display the contents of Register xx to yy
Change	S(C/R)	Display the contents of W Register, status
	W(C/R)	Display the contents of W Register, status
	Cxx(C/R)(C/R)	Change the previously displayed memory location or port or register to xx
Examine	Cxx(C/R)yy(C/R)..	Change the sequential registers or memory locations to xx,yy..
	(C/R)(C/R)	Change the PC or DC to xxxx.
Previous	E(C/R)	Display the last addressed register or memory location or port.
Next	B(C/R)	Display the previous register or memory location or port.
Load	N(C/R)	Display the next register or memory location or port.
	L(C/R)	Load formatted object paper tape. If (CK) prints, then checksum error has occurred on block last read.
Hexadecimal Block Move	H(C/R)	Load formatted object paper tape from the high-speed reader.
	Xaaaa+bbbb=	Add or subtract value bbbb from value aaaa.
	Mdddd(C/R)	Move memory block starting at ssss and ending at address eeee to memory destination address dddd
Punch	Vssss-eeee(C/R)	Binary Punch PROM 8 bit format; xxxx is starting page address and yyyy is ending page address. z is block length code 0=256,1=512. To Punch 0 to BFF then enter BO-C00-0.
	Jxxxx-yyy-y-z	Formulator Formatted punch for future load.
Go To	G(C/R)	Go to address of PO.
	Gaaaa(C/R)	Change PO to address aaaa, then go to aaaa and continue execution.
Delete Breakpoint	[	Delete command and start a new command.
	Uaaaa(C/R)	Set breakpoint at address aaaa.
Find	T(C/R)	Clear breakpoint restore user instruction.
	Kxx(C/R)	Find the matching byte starting with the last memory address displayed and searching through memory.

Figure 1 depicts the keypad organization supported by PEPBUG software routines and Appendix E shows the response a user would receive if utilizing the subroutine SCAN. The schematic diagram for the keyboard and display is in Appendix F. Ports 20 and 21 are used for the keyboard and display I/O functions. Appendix H has the complete PEPBUG program listing.

Figure 2 is a memory map of the area used by PEPBUG. Appendix G demonstrates how this memory area is changed with specific PEPBUG commands.



**Fig. 1 PEPBUG Keyboard Display Diagram**

RAM Address	Memory Contents							
2B80	R0	R1	R2	R3	R4	R5	R6	R7
2B88	R8	R9	R'A'	R'B'	R'C'	R'D'	R'E'	R'F'
2B90	R'10'	R'11'	R'12'	R'13'	R'14'	R'15'	R'16'	R'17'
2B98	R'18'	R'19'	R'1A'	R'1B'	R'1C'	R'1D'	R'1E'	R'1F'
2BA0	R'20'	R'21'	R'22'	R'23'	R'24'	R'25'	R'26'	R'27'
2BA8	R'28'	R'29'	R'2A'	R'2B'	R'2C'	R'2D'	R'2E'	R'2F'
2BB0	R'30'	R'31'	R'32'	R'33'	R'34'	R'35'	R'36'	R'37'
2BB8	R'38'	R'39'	R'3A'	R'3B'	R'3C'	R'3D'	R'3E'	R'3F'
2BC0	ACC	IS	W	BAUD	DO	DO	P1	P1
2BC8	D1	D1	EI/DI	JMP	PO	PO	--	--
2BD0	←BRK ADDR →		←USER INST SAVE →				--	--
2BD8	←PROM PROGRAM PARAMETERS →							
2BE0	←PORT DISPLAY + CHANGE WORK →						--	--
2BE8 -2BFF	←NOT USED →							

**Fig. 2 RAM Utilization by PEPBUG**

Note: See Appendix G PEPBUG examples for further clarification of the RAM utilization.  
 (--) indicates RAM space not used by PEPBUG and is available to the user.

### 3.0 Detailed Command Explanation

In the following list of commands the TTY key is in ( ) when different from keypad. All data and addresses are in hexadecimal notation.

- C/R            Carriage return. The keypad C/R operates the same as a TTY C/R. It is the termination of all command strings, except for the change command which is explained below. All commands except delete must be terminated with C/R.
- DEL([ ])      The Delete command cancels any previous command string that has not yet been terminated with a C/R.
- CHG(C)        The change command will change the last displayed register, memory location, register, or port. If the previous display was PC or DC then the change expects four ASCII digits. If a port was the last displayed item then only the last keyed field will be accepted; port changes cannot be chained. If PC or DC change is chained then memory starting at the PC address will be changed in 2-byte segments.
- + or -        These characters are delimiters only. They are only valid for the commands Hex, Block Move, Punch, and multiple display of Mem, Reg, or Port.
- REG (R)       Display the register as addressed. RXX will display register XX. RXX-YY will display the block of register that include XX-YY. If XX is 13 and YY is 21, then registers 10-2F will be displayed. The same block addressing also applies to Memory and Port displays.
- Port(P)       Display the port as addressed. This command builds an executable command sequence in Memory and then executes the instructions. See the Memory Map and also refer to the examples in the Appendix. Port change also builds a sequence of instructions to be executed. The same address blocks apply as in the Reg command. A port change will immediately change the contents of a port; register changes are done in RAM and the real register changes occur when a GO command is issued.

BRK(U) Set a breakpoint at the given memory address. The sequence of instructions stored in the users program are:

1. OUTS 'C' to save the accumulator.
2. JMP to PEPBUG address H'80E0'.

The breakpoint address and the user's original instructions are saved in the PEPBUG RAM area. Only the information from the last breakpoint is saved. If a user requires multiple breakpoints the 4 bytes of code may be stored; however address and data must be logged and the code manually restored where the breakpoints were stored.

Since this is a 4 byte sequence care must be used as to where a breakpoint may be stored. It must be the first byte of an instruction; branch or JMP instructions into the middle of the 4-byte sequence will result in erroneous execution. See section 4.0 for further information.

C.BRK(T) Clear breakpoint will restore the 4 bytes of user code that were saved by the last set breakpoint command.

LOAD(L) Load from the RS232 device an object formatted program. Normally this would be a TTY paper tape. However it may also be data being down-loaded from an RS232 line from the Formulator or from any other computer system. The two formats which are accepted are Fairbug or Formulator formats as detailed in Appendix C.

(H) Load from a high speed paper tape. There is no corresponding key pad instruction; however, a GO H'84B8' will execute the high speed load routine.

GO(G) GO to PC0 address and execute after restoring all registers. The Accumulator is not restored so it is necessary to be careful as to what instruction will execute next.

MEM(M) Display the memory as addressed. The same address blocks apply as in the Reg Command.

NEXT(N) Display the next register, port or memory location.

PREV(B) Display the previous register, port, or memory location.

(E) Examine current register, port or memory location. This TTY command has no corresponding command on the keypad.

ACC(A) Display the contents of the accumulator.

W Display the contents of the status register.

Bit	Meaning	
0	Sign	- 1 means positive or zero
1	Carry	- 1 means carry from bit 7 on previous arithmetic instruction
2	Zero	- 1 means zero result
3	Overflow	- 1 means overflow
4	ICB	- 1 means interrupt control is enabled.

When entering PEPBUG the ICB is set to zero before the status register is saved in RAM. The user may set ICB to 1 prior to executing a GO command. The GO sequence will restore the low 4 bits of the status register then execute an Enable or Disable Interrupt instruction prior to executing a JMP to the PCO address.

IS(I) Display the contents of the Indirect Scratchpad Address Register(ISAR)

PC(P) Display the contents of the Program Counter (PCO) on the Stack Register (PCI)

DC(D) Display the contents of either Data Counter (DC0 or DC1).

HEX(X) Add or Subtract the two hexadecimal numbers and display the answer.

MOVE(V) Move a block of memory. This command must be preceded by a Memory display command which displays the first byte of the destination where the memory block is to be moved. Then the Move command provides the starting and ending address of data to be moved. The block may be moved up or down and may overlap itself. There are no restrictions on the move.

FIND (K) Find the matching bit pattern. Start the search at the last displayed Memory address. If the last displayed item was not memory this command will not work correctly. When a match is displayed a subsequent key in of the save command will start a new search at the next memory location.

E70 Program the F38E70. This command may be accomplished from the TTY by a GO H'8745'. Before executing there are several things to be accomplished:

1. Set Port 0 to H'FF'
2. Set Port 1 to zero
3. Connect 25V power supply
4. Store memory starting address at H'2BD8' & H'2BD9'
5. Store Eprom starting address at H'2BDA' & H'2BDB'
6. Store Eprom ending address at H'2BDC' & H'2BDD'
7. Store at H'2BDE' a H'FF' for blank check or store zero to skip the blank check.

Note: Since the addresses are supplied to the EPROM programmer small segments of data may be put on the EPOM and more added later. This is useful when developing subroutines; after a routine is debugged it can be put into EPOM and executed. This eliminates reloading that portion of code. A library of routines can be built in this manner. See Appendix F for the pinouts required.

2716 Program the F2716 EPROM. This command may be accomplished from the TTY by executing a GO H'8741' command. The same rules apply to 2716 programming as to the E70 Programming.



### 3.1 PEPBUG Command Usage

A brief study of Appendix B will assist the reader in understanding the TTY command utilization. From these examples many advantages over Keypad debugging are quite apparent. First and most important a history is recorded of the steps taken and the results displayed. Second is the simple access from the console to the user.

One advantage that is not so apparent is that when a user wishes to end a debug session the modified program may be saved by punching the memory using the F command. In a subsequent debug session this "updated" program may be loaded and debugging continued. If program patches were extensive much time and effort is saved. Of course the program may be re-assembled after editing the changes in order to obtain an updated program.

Another advantage and also not so apparent is the fact the TTY input is much less error prone. If the one uses a little care and examines the keyed input prior to the carriage return, errors can almost be totally eliminated. In addition, a debug session achieves more results in a quicker time.

### 3.2 Input/Output

The input/output routines assume that Port A of the U4 socket is available and is configured as shown in Appendix F. In order to communicate with an 11-bit serial type device such as a Teletype ASR 33 or a compatible TTY or CRT is required. PEPBUG has options to vary the baud rate by changing the counters for delay loops between bits. The timer is not utilized so the user is not deprived of this valuable resource. The counter value for 110 Baud is six which counts by incrementing an 8-bit register until zero; the six gives a loop count of 250. For 300 Baud the counter is initially H'A4' giving a loop count of H'5C' or 92 decimal. Other baud rates can be achieved by modifying the counter. These counts assumed a system clock of 2MHz. For a faster clock, the counter must be changed to produce more delay loops, while for a slower clock the counter must be changed to produce less delay loops. This is due to the fact that each instruction time will change and the total loop time will change while the device spelled is always constant. When the baud rate is not set to default to either 110 or 300 or 1200 then the Baud Rate must be put into RAM location H'2BC3'.

### 3.3 Load from Parallel Input Device

The loader in PEPBUG can load from either a teletype or from a parallel source. The usual parallel source would be a high-speed paper tape reader which reads 100 to 300 characters per second. The parallel device is controlled using a "handshaking" protocol. The handshaking eliminates synchronizing problems because it forces the device to wait for the microprocessor and forces the microprocessor to wait for the device.

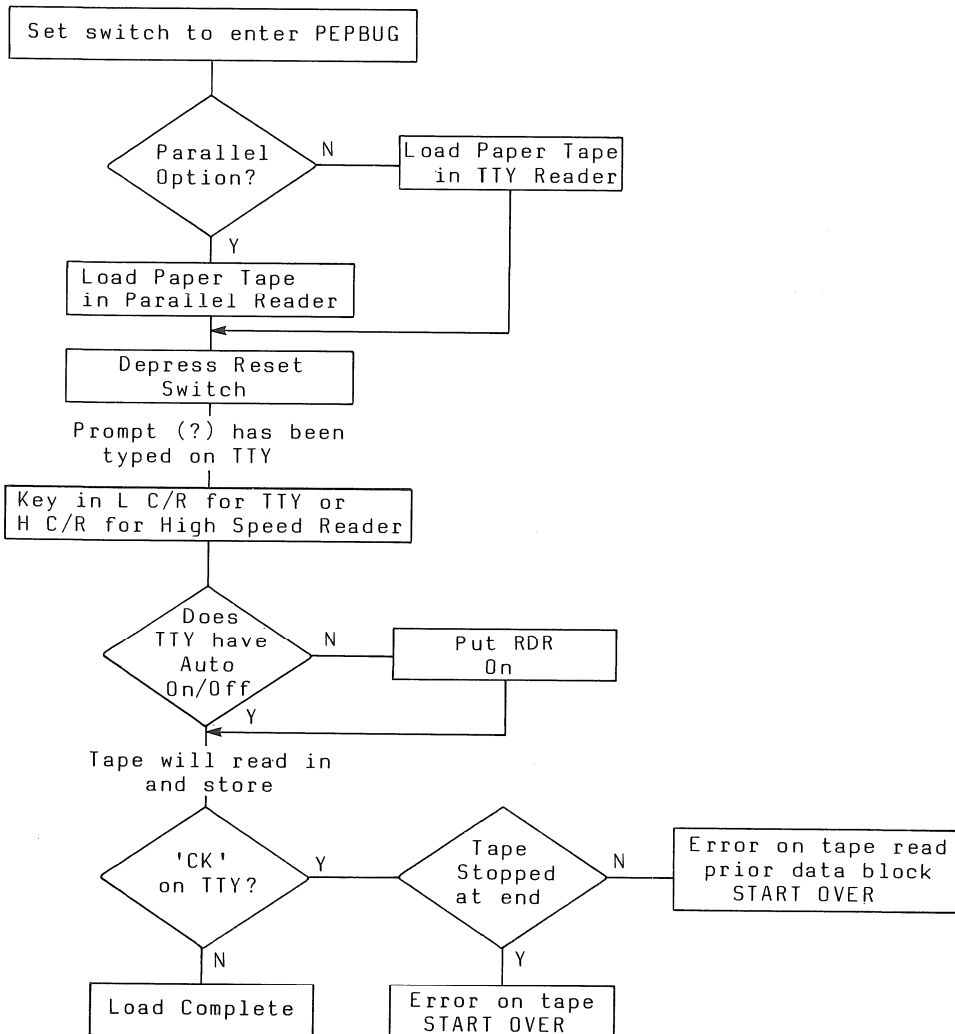


Fig. 3

The PEPBUG parallel read routine examines DEVICE READY and waits for the ready signal, looks for Character Ready and delays 100us after detecting the ready, and then reads a character before the output of Step Reader. This sequence is repeated for each character. Only the formats shown in Appendix C can be read by PEPBUG with the Load Command. However, the user may use the subroutine PINP to read other formats. Bits 1-2 are examined when is entered to initialize the baud delay counter. See Appendix A for the I/O port pin assignments. The flow chart in Figure 3 indicates the steps necessary to load a paper tape program formatted as shown in Appendix C. If a checksum error occurs the FAIR-BUG format has no recovery capability since the address is on the beginning of the tape.

#### 4.0 PEPBUG Subroutines

I/O Subroutines on the PSU are available to the user. These are listed below and documented in Appendix E.

NAME	ENTRY ADDRESS	FUNCTION
ITX	85C7	Input 1 byte from TTY type device (11 bits serial/character).
TTYX	8617	Output 1 byte to TTY type device (11 bits serial/character).
TCRX	85F9	Output CR,LF & delay 250 ms using TTYX subroutine.
PINP	85A3	Input 1 byte from the parallel IP device (150usec minimum delay between characters).
FOP1	812A	Output 1 or 2 hexadecimal digits in ASCII format from a memory location.
FOP2	812C	Output 1 or 2 hexadecimal digits in ASCII format from register QL.
BYTE	857B	Input 2 ASCII digits from a parallel or serial IP device; then covert them to one hexadecimal byte.
SCAN	8636	Read keyboard until new character is received; also output to the the six 7-segment displays and two LED's.
DISPX	87E2	Output to the six 7-segment displays and two LED's.

## 5.0 Programming Examples

The following program linkages to PEPBUG are cited as examples as how to utilize PEPBUG as a "snap shot" display or breakpoint. vehicle.

The two general ways to enter PEPBUG are either through programmed instructions or by manual intervention from the console. The manual entry is normally used for the following situations:

- For initial program loading
- For program punch and save
- To start tracing a runaway program
- To display or take further action following a BR\*

Programmed entry to PEPBUG can be achieved by building trace routines into the source program prior to assembly or else by patching the object program after loading it. In either case the following instructions provide the necessary linkage.

JMP H'8080'	This is a 3-byte instruction that destroys the accumulator and does not save the program counter.
PI H'8080'	This is a 3-byte instruction that destroys the accumulator and pushes the program counter (PC0) to the stack (PC1).
LR P0,Q	This is a 1-byte instruction that does not destroy the accumulator and does not save the program counter.
PK	This is a 1-byte instruction that does not destroy the accumulator and pushes the program counter (PC0) to the stack (PC1).

These instructions are explained in detail in the F8 USERS GUIDE. It should be noted that all except the LR PO,Q are privileged instructions and that an interrupt cannot be serviced following these instructions. Furthermore, the first instruction executed at H'8080' is a disable interrupt which inhibits interrupts until an Enable instruction is issued.

Examination of these instructions discloses that the PK instruction is the most desirable to use since it is 1 byte, saves the accumulator, and saves the PC0; however, it does require the K register (R12 and R13) be preset. If K is not used in the program being tested this is an ideal instruction. A recommended procedure is to code in to the users program at location 0 the following instructions:

```
LI H'80'  
LR KU,A  
LR KL,A
```

If one of the above housekeeping procedures is used, patching the user's program for tracing or display purposes becomes an easier proposition. The simpler PK may be used and the prompt(?) will indicate that a desired point has been reached. Display of PC1 will then determine which (if more than one) of the trace points has been reached. The disadvantage of using PK or PI is that the object program PC1 has been lost. If this is detrimental for a particular section of code being debugged the JMP or LR PO,Q instructions are available. If LR PO,Q is used the Q register must be preset in the same fashion as above, substituting Q for K. The type of breakpoint to be used will depend on which registers are not used in the portion of the code being debugged.

It can be noted that the options are numerous and the user must decide which one or which combination is most desirable for a given purpose.

It is suggested that the user try a small program as shown in Appendix G and utilize all the options of PEPBUG until all the options are understood. This will save much time and effort in future debug sessions.



## APPENDIX A ASCII CHARACTER CODES

Character	Hex Code 7 Bit	Character	Hex Code 7 Bit	Character	Hex Code 7 Bit
(Space)	20	0	30	H	48
!	21	1	31	I	49
"	22	2	32	J	4A
#	23	3	33	K	4B
\$	24	4	34	L	4C
%	25	5	35	M	4D
&	26	6	36	N	4E
'(Quote)	27	7	37	O	4F
(	28	8	38	P	50
)	29	9	39	Q	51
*	2A	:	3A	R	52
+	2B	;	3B	S	53
,(Comma)	2C	>	3C	T	54
-	2D	=	3D	U	55
.	2E	<	3E	V	56
/	2F	?	3F	W	57
Line Feed	0A	@	40	X	58
Carriage RTN	0D	A	41	Y	59
Bell	87	B	42	Z	5A
Punch ON	92	C	43	[	5B
Punch OFF	94	D	44	\	5C
Reader ON	91	E	45	]	5D
Reader OFF	93	F	46	↑	5E
Null	7F	G	47	←	5F
Null	FF				

## APPENDIX B PEPBUG PORT ASSIGNMENTS

### Assignments for Port 28

Bit	Function
7	Serial input
6	Character Ready (Parallel Device)
5	Enter Key/Display Monitor
4	Device Ready(Parallel Device)
3	Step Reader(Parallel Device)

Bit 2	Bit 1	Baud Rate
0	0	110 baud
0	1	300 baud
1	0	1200 baud
1	1	Baud delay counter in memory at H'2BC3'

### Assignments for Port 29

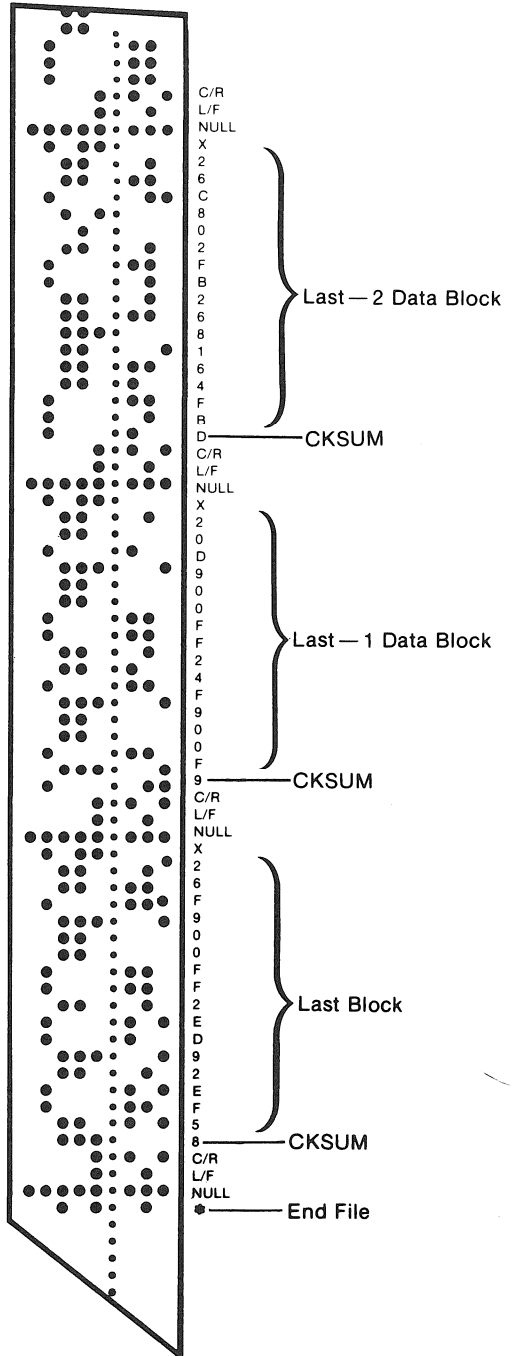
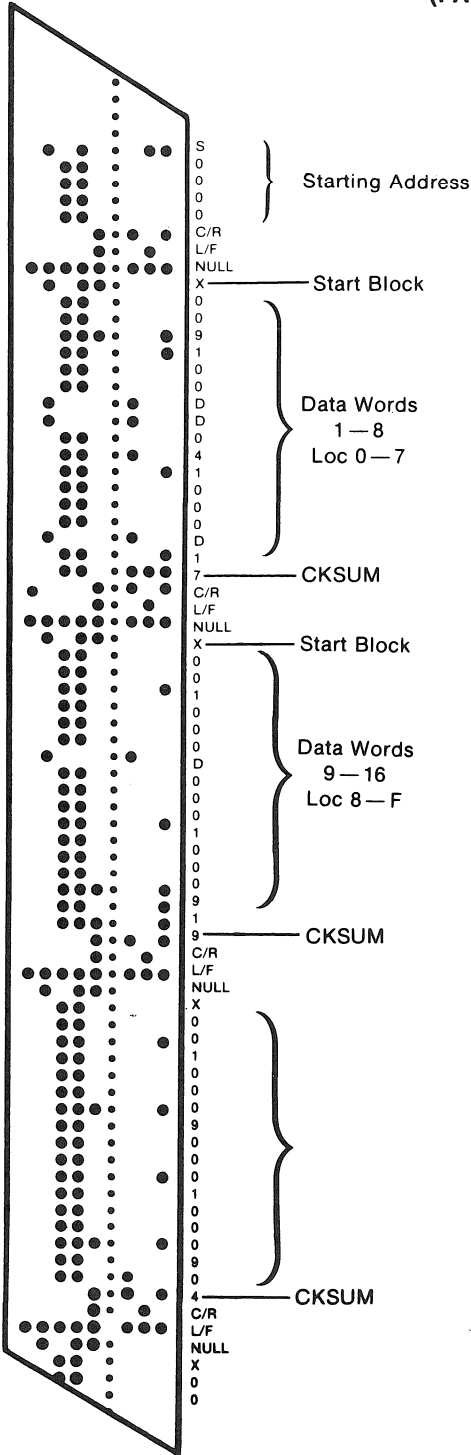
Bit	Function
7	Parallel input byte - MSB
6	Parallel input byte
5	Parallel input byte
4	Parallel input byte
3	Parallel input byte
2	Parallel input byte
1	Parallel input byte
0	Parallel input byte - LSB

Note: 0=+5;1-GND,0 V.

0 Serial Output

# APPENDIX C FORMATTED TAPE (FAIRBUG FORMAT)

Note: This example was loaded (L) by the instruction shown in Appendix B. This is ASCII 7 bit format.

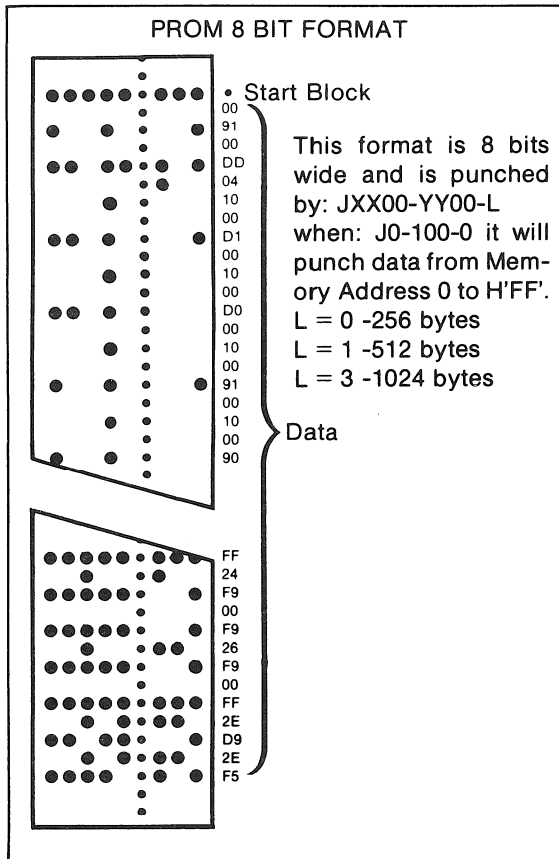








**APPENDIX D  
PEPBUG  
(PROM PUNCH FORMAT)**



Input Command JXX00-YY00-L	Decimal Memory Addresses Punched	# Blocks	Block Length
J 0-100-0	0-255	1	256
J 0-400-0	0-1023	4	256
J 100-200-0	256-511	1	256
J 0-400-1	0-1023	2	512
J 0-1000-1	0-4095	8	512
J 200-400-1	512-1023	1	512

## APPENDIX E PEPBUG SUBROUTINES

The following INPUT and OUTPUT subroutines exist in PEPBUG and may be called by the user's program. All subroutines should be entered by: (PI Address).

TTX - Input 1 byte from TTY type device, without echo.  
Data is 11 bits/character being received on Port 28  
Pin 7.

Address: H'85C7'  
Enter: R0 Delay Counter  
Exit: W Reg Destroyed  
PC1 User return address  
Accum Input byte  
R0 Unchanged  
R1 Input byte  
R2 -1

TTYX- Output 1 byte to TTY type device. Data transmitted  
is 11 bits/character being output on Port 28 Pin 0.

Address: H'8617'  
Enter: R0 Delay Counter  
R1 Byte to output  
Exit: W Reg Destroyed  
PC1 User return address  
Accum 0  
R0 Unchanged  
R1 -1  
R2 0

TCRX- Output CR/LF/NULL to TTY type device; subroutine  
TTYO is called.

Address: H'85F9'  
Enter: R0 Delay Counter  
Exit: W Reg Destroyed  
PC1 Destroyed  
Accum 0  
K Reg User return address  
R0 Unchanged  
R1 -1  
R2 0

PINP- Input 1 byte from parallel input device; minimum  
delay between characters is 150 us. Byte is  
received on Port 29 with control bits on Port 28,  
pins 3,4, and 6. See Appendix B.

Address: H'85A3'  
Enter: No setup  
Exit: W Reg Destroyed  
PC1 User return address  
Accum Input byte  
R1 Input byte

## APPENDIX E (continued)

BYTE - Input 2 ASCII hexadecimal characters and convert to 1 byte; also accumulate the checksum. If input is not ASCII characters 0-9 or A-F meaningless results will be returned. Either TTYI or PINP is called as input routine.

```
Address:      H'857B'  
Enter:        Q      H'85A3'(for parallel input)  
              Q      H'85C7'(serial input) R0=Delay  
              Counter  
              R7      Previously accumulated checksum  
Exit:         W Reg   Destroyed  
              PC1     Destroyed  
              Accum   Input byte  
              K       User return address  
              Q       Unchanged  
              R0      Unchanged  
              R1      Destroyed  
              R2      -1 (if serial IP), unchanged for  
              Parallel IP  
              R7      Checksum  
              R8      0  
              R11     Input byte
```

FOP1 - Output byte of data from memory to TTY type device using TTYO subroutine. Byte is converted to 1 or 2 ASCII hexadecimal characters.

```
Address:      H'812A'  
Enter:        R0      Delay Counter  
              R8      Flag pos# = OP Hi 4 bits, then Lo 4  
              as ASCII  
              Neg#    OP Lo 4 bits as ASCII  
              DCO     Memory address of data  
Exit:         W Reg   Destroyed  
              PC1     Destroyed  
              Accum   Destroyed  
              DCO     DCO + 1  
              K Reg   User return address  
              QL      Data byte  
              R0      Unchanged  
              R1      -1  
              R2      0  
              R7      Checksum (low 4 bits significant)
```

FOP2 - Output byte of data from QL. Same routine as FOP1 except DCO is not used.

```
Address      H'812C'  
Enter:       R0      Delay Counter  
              R8      Same as FOP1  
              QL      Data byte to output  
Exit:        Same as FOP1
```

## APPENDIX E (continued)

DISPX Display data to six 7-segment displays and two LED's.

```

Address:    H'87E2'
Enter:      R'30'to'35'    Low 4 bits from each
                                register are displayed.
                                R'36'    Bit 7 is Prompt LED
                                Bit 6 is Change LED
Exit:       ISAR          is destroyed
            R1           is destroyed
            Port '20'&'21' are destroyed
            ACC          is destroyed
            W           is destroyed
    
```

SCAN Display data to six 7-segment displays and two LED's and scan keyboard for new key in. This routine will wait for new key before returning to the user.

```

Address:    H'8636'
Enter:      R'30'to'35'    Low 4 bits from each
                                register are displayed.
                                R'36'    Bit 7 is Prompt LED
                                Bit 6 is Change LED
Exit:       R'36'        is changed
            R'37'        is number of C/Rs expected
                                to complete change command
            DCO          is destroyed
            W           is destroyed
            R2          is keypad location code in Hex
            Port '20'&'21' are destroyed
            R'28'to'2F'  are destroyed
            ACC and R1  new key character in ASCII
    
```

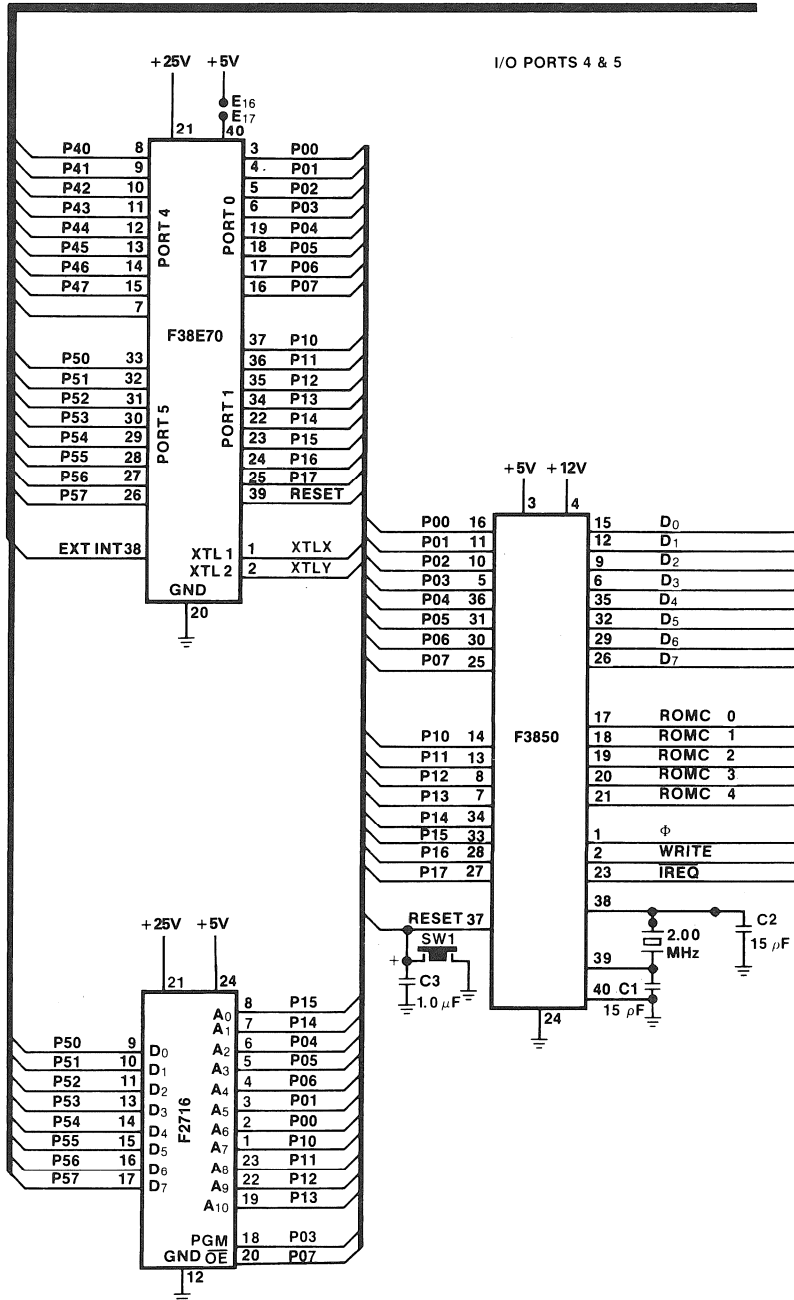
The characters returned by this routine are dependent upon the setting of Bit 7 of Reg H'36'. This flag bit determines when a keypad shift is invoked. This bit is set to a 1 when Del is keyed in or when C/R is keyed in. It also causes the Prompt LED to be lit. It will be set to 0 on the next key in except on Del. The following tables show the returned characters.

	Results in R1 in ASCII	Results in R2 in Hex
Reg '36'= 0	C D E F - + 8 9 A B 0 C 4 5 6 7 R H'5B' 0 1 2 3 M H'0D'	3 7 B F 13 17 2 6 A E 12 16 1 5 9 D 11 15 0 4 8 C 10 14
Reg '36'=1	U T L G - + * X V K 0 C * I W A R H'5B' P D B N M H'0D'	1B 1F 23 27 13 17 * 1E 22 26 12 16 * 1D 21 25 11 15 18 1C 20 24 10 14

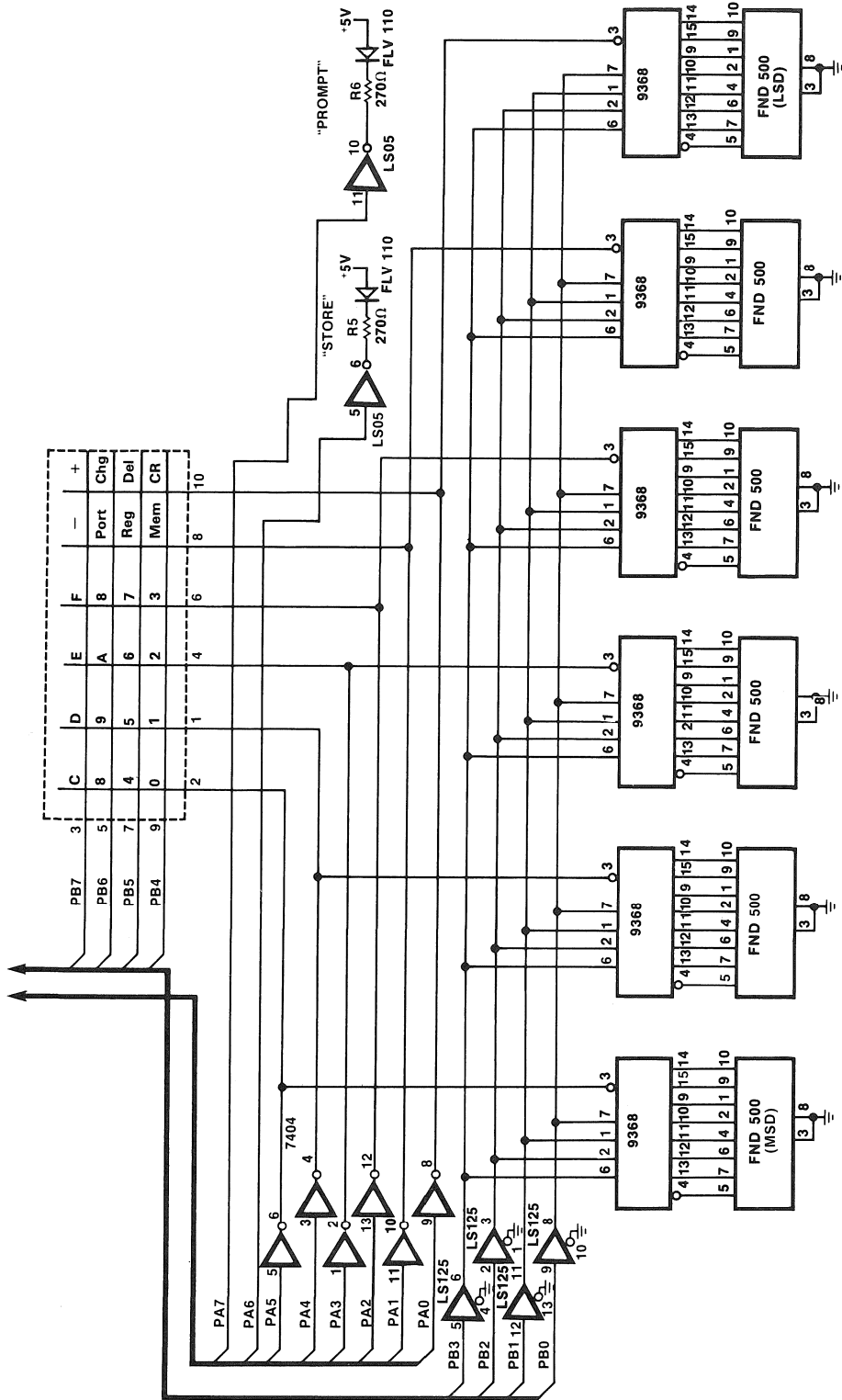
NOTE: \* These keys cause a direct JMP to the E70 or 2716 EPROM routines and do not return to the user.

# APPENDIX F

## F2716 AND F38E70 PROM PROGRAM I/O INTERCONNECT DIAGRAM



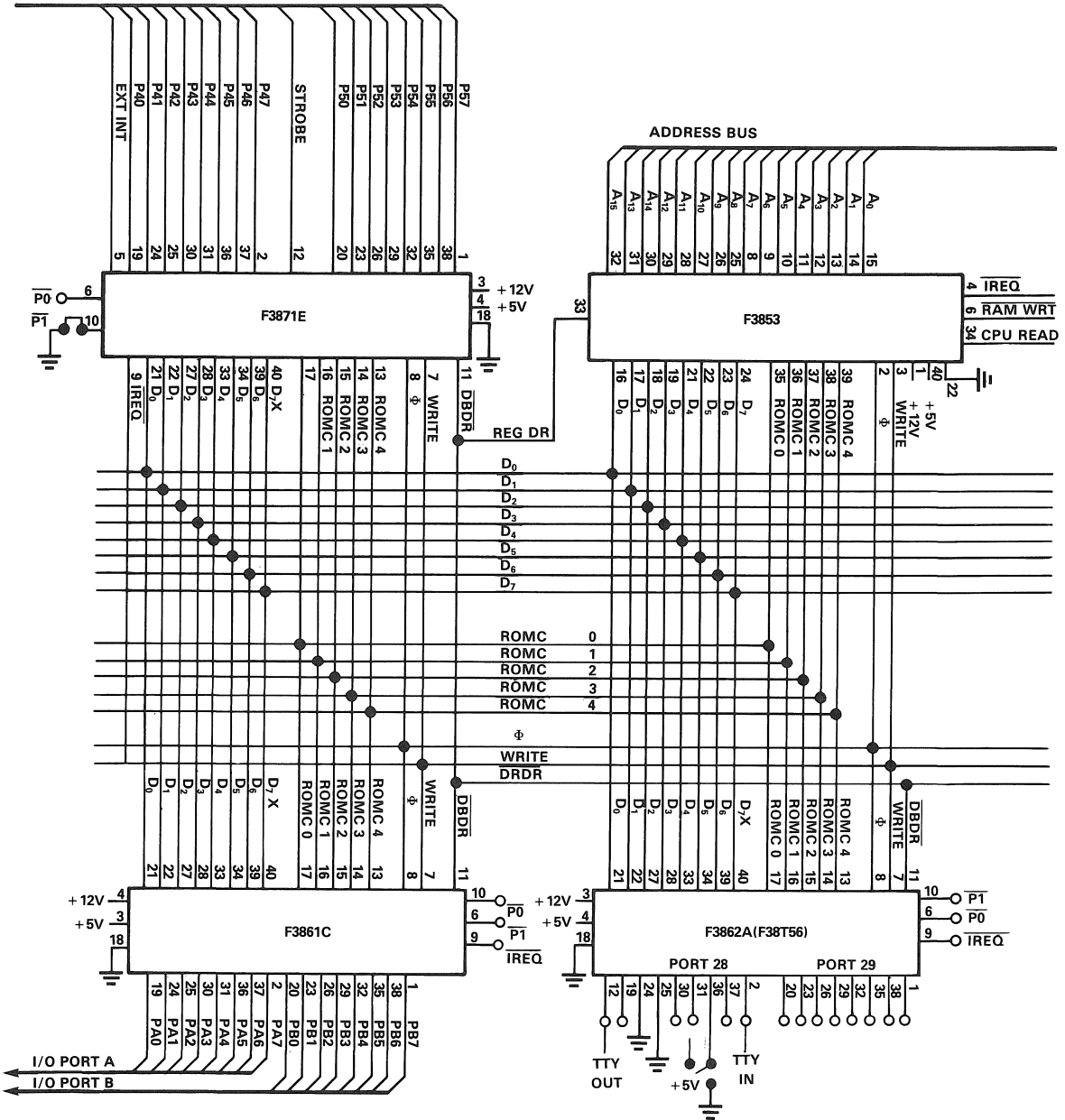
# APPENDIX F (Continued) KEYBOARD/DISPLAY SCHEMATIC DIAGRAM



Notes:  
 PA7 = PORT 20 Bit 7  
 PB7 = PORT 21 Bit 7  
 Resistors are 1/4 watt



# I/O Ports to Keypad and TTY Interconnect Diagram Appendix F (Continued)



## APPENDIX G PEPBUG EXAMPLES

```

MO(CR)
M0000= FF
?C70(CR)(CR)
?N(CR) M0002= 6A
?B(CR) M0001= 8B
?E(CR) M0001= 8B
?COB(CR)(CR)
?E(CR) M0002= 6A
?C5C(CR)(CR)
?E(CR) M0003= 81
?C1F(CR)25(CR)40(CR)94(CR)F9(CR)29(CR)80(CR)80(CR)(CR)
?MO-A(CR)
M0000=(70 0B 5C 1F 25 40 94 F9 29 80 80)A9 B0 89 11 93
?RO-3F(CR)
R0000= 04 00 02 00 00 00 00 00 00 00 00 00 00 00 00 00
R0010= FF 61 EE 5E F7 E4 85 5C FA D2 E6 E6 DD DE BB 90
R0020= 00 00 00 00 02 00 00 00 00 00 02 00 01 00 00 00
R0030= FF D6 FF B3 B6 A3 FE 9A DF D4 7F 76 F7 88 E7 91
?M2B80-2B87(CR)
M2B80= 04 00 02 00 00 00 00 00 00 00 00 00 00 00 00 00
?GO(CR)
← Execute.
← Program did not return, manually reset

?M7(CR)
M0007= F9 ← Change displacement of BR instruction
?CFA(CR)(CR)
?GO(CR) ← Execute
?RO-3F(CR)
R0000= 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
R0010= 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F
R0020= 20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F
R0030= 30 31 32 33 34 35 36 37 38 39 3A 3B 3C 3D 3E 3F
?PO(CR)=0000 ← Go 0 set this address
?P1(CR)=8080 ← Return address not saved
?M8(CR)
M0008= 29
?C28(CR)(CR)
?GO(CR)
?PO(CR)0000
?P1(CR)=000B ← Now OK.
?A(CR)= 80
?I(CR)= 3F
?W(CR)= 07
?DO(CR)=0001
?D1(CR)=0000
?CI234(CR)(CR) ← Change
?D1(CR)=1234
?M2B80-2BFF
M2B80= 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
M2B90= 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F
M2BA0= 20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F
M2BB0= 30 31 32 33 34 35 36 37 38 39 3A 3B 3C 3D 3E 3F
M2BC0= 80 3F 07 06 00 01 00 0B 12 34 1A 29 00 00 FF EB
M2BD0= 66 FF 00 10 89 00 DF EF 76 FF 00 10 83 00 FF EF
M2BE0= 72 FF 00 10 89 02 FF EF 76 FF 00 10 89 00 FF EF
M2BF0= 76 FF 00 10 89 00 FB EF 72 FF 00 10 89 20 FF EF

```

Store a program to put an incrementing counter in scratchpad registers 0-3F. Note the use of the various instructions.

Display the program

Display registers before.

Memory where RO-F is saved.

All registers are changed OK.

Oh! We used a jump Change it to PI. Execute Again.

Now OK.

Displays

See RAM map to verify display of registers.

## APPENDIX G (continued)

```

?00-9(CR)           Display Ports. Note 2-3
00000= 88 FF 8D 8D  80 80 80 80  FF FF FF FF  00 80 FF FF
?01(CR)                                     Port changes not chained.
00001=  FF                                         Port 1 was set to last.
?CAA(CR)BB(CR)CC(CR)DD(CR)EE(CR)FF(CR)(CR) ← Old value FF changed to FF.
00-F(CR)
00000= 88 FF 8D 8D  80 80 80 80  FF FF FF FF  00 80 FF FF
?01(CR)
00001=  FF
?C1(CR)2(CR)3(CR)4(CR)5(CR)6 (CR) (CR) ← Try again to change Port 1.
?00-F(CR)
00000= 88 06 8D 8D  80 80 80 80  FF FF FF FF  00 80 FF FF
?M2BDO-2BE4(CR)
M2BDO= 66 FF 00 10  89 00 DF EF  76 FF 00 10  83 00 FF B1
M2BEO= AF 07 29 84 8D 02 FF EF  76 FF 00 10  89 00 FF EF
?01(CR)           ↑ Last set of instructions executed when display of 00-F.
00001= 06           Port data saved in QL then JMP back into PEPBUG.
?M2BDO-2BE4(CR)
M2BDO= 66 FF 00 10  89 00 DF EF  76 FF 00 10  83 00 FF B1
M2BEO= A1 07 29 84 8D 02 FF EF  76 FF 00 10  89 00 FF EF
?01(CR)
00001= 06           Start of sequence of instructions
?CAA(CR)(CR) ← KILL           to change Port 1. Data from R4
?M2B80-[(CR)           to port then JMP to PEPBUG.
?M2BDO-2BE4(CR)
M2BDO= 66 FF 00 10  89 00 DF EF  76 FF 00 10  83 00 FF B1
M2BEO= 44 B1 29 83 30 02 FF EF  76 FF 00 10  89 00 FF EF
?021 (CR)
00021= 0A           Display sequence for long IO instruction
?M2BDO-2BE4-2BE4(CR)
M2BDO= 66 FF 00 10  89 00 DF EF  76 FF 00 10  83 00 FF B1
M2BEO= 26 21 07 29 84 8D FF EF  76 FF 00 10  89 00 FF EF
?C55 (CR)(CR)
?M2BDF-2BE0 (CR)           Last displayed value was M2BEF, and is changed.
M2BDO= 66 FF 00 10  89 00 DF EF  76 FF 00 10  83 00 FF B1
M2BEO= 26 21 07 29  84 8D FF EF  76 FF 00 10  89 00 FF 55
?021 (CR)
00021= 0A ← Try again to change port 21.
?C55(CR) (CR)
?M2BD4-2BEF (CR)           Instructions for change command.
M2BDO= 66 FF 00 10  89 00 DF EF  76 FF 00 10  83 00 FF 27
M2BEO= 21 44 27 21 29 83 30 EF  76 FF 00 10  89 00 FF 55

?00-21(CR)           Display of ports again with changes noted.
00000= 88 AA 8D 8D  80 80 80 80  FF FF FF FF  00 80 FF FF
00010= FF FF FF FF  FF FF FF FF  FF FF FF FF  FF FF FF FF
00020= 2B 55 FF FF  81 FF FF FF  FF FF FF FF  FF FF FF FF

```

## APPENDIX G (continued)

```

?U7(CR) ←———— Set breakpoint at M7. Note instructions put in user
?M7(CR)           program and data saved in RAM to restore.
M0007=  BC
?M7-A(CR)
M0000=  70 0B 5C 1F  25 40 94 BC 29 80 E0 A9  B0 89 11 93
?M2BC0-2BEF(CR)
M2BC0=  80 3F 07 06  00 01 00 0B  12 34 1A 29  00 00 FF EB
M2BD0=  00 07 FA 28 80 80 DF EF  76 FF 00 10  83 00 FF 27
M2BE0=  21 44 27 21  29 83 30 EF  76 FF 00 10  89 00 FF 55
?T(CR)           Clear breakpoint. Display User instructions.
?M7-A(CR)
M0000=  70 0B 5C 1F  25 40 94 FA 28 80 80 A9  B0 89 11 93
?M2B80-2B8[(CR)
?M2B80-2BFF(CR)
M2B80=  00 01 02 03  04 05 06 07  08 09 0A 0B  0C 0D 0E 0F
M2B90=  10 11 12 13  14 15 16 17  18 19 1A 1B  1C 1D 1E 1F
M2BA0=  20 21 22 23  24 25 26 27  28 29 2A 2B  2C 2D 2E 2F
M2BB0=  30 31 32 33  34 35 36 37  38 39 3A 3B  3C 3D 3E 3F
M2BC0=  80 3F 07 06  00 01 00 0B  12 34 1A 29  00 00 FF EB
M2BD0=  00 07 FA 28  80 80 DF EF  76 FF 00 10  83 00 FF 27
M2BE0=  21 44 27 21  29 83 30 EF  76 FF 00 10  89 00 FF 55
M2BF0=  76 FF 00 10  89 00 FB EF  72 FF 00 10  89 20 FF EF
} Note RAM
} is still
} intact. we
} may still
} return to
} user
} correctly

?U7(CR) ←———— Now set breakpoint & execute the Go and return to
?G7(CR)           PEPBUG users instructions are not restored.
?MO-F(CR)
M0000=  70 0B 5C 1F  25 40 94 BC 29 80 E0 A9  B0 89 11 93
?T(CR) ←———— Now CLR restores users instructions.
?MO-F(CR)
M0000=  70 0B 5C 1F  25 40 94 FA 28 80 80 A9  B0 89 11 93
?F0-7(CR):18000000700B5C1F254094FA288080A9B08911931EC5476A83FA999512
:0000000000000000
}
} Using(CR) leader
} and trailer
}
} Formulator dump format. Note
} long delay before and after.
} These are null characters for
} paper tape leader and trailer.

?X1234-5678BBBC
?X1234-5678=BBBC
?X1234+5678=68AC
?X1234-F034=2200
?X1234-204=1030
} Use of = as delimiter
} is neater

?L(CR) CK ←———— Keyboard load. Keyed in 50000X12345678911223344
                        and 44 is checksum error

?MO-F(CR)
M0000=  12 34 56 78 90 11 22 33 28 80 80 A9  B0 89 11 93
?M3(CR)
M0003=  78 } ————— Execute MOVE of location 7-B to 3-7
?V7-B(CR)
?MO-F(CR)
M0000=  12 34 56 33 28 80 80 A9 28 80 80 A9 B0 89 11 93
?MA(CR)
M000A=  80 } ————— Move 8-B to A-D
?V8-B(CR)
?MO-F(CR)
M0000=  12 34 56 33  28 80 80 A9  28 80 28 80 80 A9 11 93

```

# APPENDIX H PEPBUG PROGRAM LISTING

ERRS	LOC	OBJECT	ADDR	LINE	SOURCE STATEMENT
				0001	*PEPBUG
				0002	*R VATERT
				0003	*FOR 2K PSU F38T56
				0004	* REVISED 2/20/79
	0000	0005	RAM	CORG	0
	0000	0006	XX	RORG	H'8000'
	02CB	0007	A	EQU	(HEX-1)
	0000	0008	BAUD	EQU	0
	0001	0009	CHRS	EQU	1
	0002	0010	BCNT	EQU	2
	0008	0011	HFLG	EQU	8
	0005	0012	CCNT	EQU	5
	0006	0013	XFLG	EQU	6
	0007	0014	CKSM	EQU	7
	0008	0015	FCNT	EQU	8
	0009	0016	FLG	EQU	9
	0007	0017	SIZE	EQU	CKSM
	0003	0018	SA	EQU	3
	0005	0019	EA	EQU	5
	0006	0020	EAL0	EQU	XFLG
	0005	0021	EARI	EQU	CCNT
	000C	0022	00	EQU	12 IO DISPLAY PORT
	000D	0023	PP	EQU	13
	000F	0024	RR	EQU	15 FLG+H'43'=REGISTER
	000A	0025	MM	EQU	10 FLG+H'43'=MEMORY
	000B	0026	CC	EQU	11
	000B	0027	CHR1	EQU	11
	000D	0028	CR	EQU	H'0D'
	000A	0029	LF	EQU	H'0A'
	00EB	0030	NUSE	EQU	H'EB'
	0028	0031	OPOR	EQU	H'28'
	0028	0032	IPOR	EQU	H'28'
	0028	0033	PSTS	EQU	H'28'
	0029	0034	PPRT	EQU	H'29'
	0044	0035	D0	EQU	RAM+68 D0 ADDRESS IN RAM
	0009	0036	RA9	EQU	RAM+9 REG 9 ADDR IN RAM
	0020	0037	PORTA	EQU	H'20'
	0024	0038	PORTB	EQU	H'24'
	0029	0039	PORTC	EQU	H'29'
	004A	0040	EX	EQU	RAM+74
	000A	0041	HII	EQU	10
	000B	0042	LO	EQU	11
	0020	0043	P8	EQU	H'20'
	0021	0044	P9	EQU	H'21'
	0058	0045	BRAM	EQU	RAM+88 MEM FOR BURN PARAMETERS
				0046	*
				0047	*
8000	298000	8000	0048	JMP SAVER	NO ENTRY @8000 GO TO 8080
				0049	*
				0050	*
				0051	*BOOTSTRAP FORMAT PUNCH SUBROUTINE
				0052	* START ADDRESS INDC ON ENTRY
				0053	* END ADDRESS IN R6-7 ON ENTRY
				0054	*WRITE BLANK LEADER
8003	45		0055	FPUN LR	A, EARI
8004	18		0056		COM
8005	55		0057	LR	EARI, A NOW 1'S COMPLIMENT HIGH
8006	46		0058	LR	A, EALO
8007	18		0059		COM NEGATIVE
8008	1F		0060		INC 2'S COMPLIMENT
8009	56		0061	LR	EALO, A

## APPENDIX H (continued)

ERRS	LOC	OBJECT	ADDR	LINE	SOURCE	STATEMENT
	800A	45		0062	LR	A, EAHI
	800B	19		0063	LNK	
	800C	55		0064	LR	EAHI, A
	800D	2092		0065	LI	H'92'
	800F	57		0066	LR	CKSM, A
	8010	51		0067	FPM1	CHRS, A
	8011	288610	8610	0068	PI	TTYO
	8014	37		0069	DS	CKSM
	8015	70		0070	LIS	0
	8016	94F9	8010	0071	BNZ	FPM1
	8018	E8		0072	XS	HFLG
	8019	810A	8024	0073	BP	FCON
				0074	*TRAILER	PUNCHED NOW FINISH
	801B	2094		0075	LI	H'94'
	801D	51		0076	LR	CHRS, A
	801E	288610	8610	0077	PI	TTYO
	8021	298076	8076	0078	JMP	TZE
				0079	*	
				0080	*WRITE	STARTING ADDRESS
	8024	20FF		0081	FCON	LI
	8026	51		0082	LR	CHRS, A
	8027	288610	8610	0083	PI	TTYO
	802A	203A		0084	LI	H'3A'
	802C	51		0085	LR	CHRS, A
	802D	288610	8610	0086	PI	TTYO
	8030	2018		0087	LI	24
	8032	58		0088	LR	HFLG, A
	8033	07		0089	LR	QL, A
	8034	28812C	812C	0090	PI	FOP2
	8037	4A		0091	LR	A, 10
	8038	07		0092	LR	QL, A
	8039	28812C	812C	0093	PI	FOP2
	803C	4B		0094	LR	A, 11
	803D	07		0095	LR	QL, A
	803E	28812C	812C	0096	PI	FOP2
	8041	70		0097	LIS	0
	8042	07		0098	LR	QL, A
	8043	28812C	812C	0099	PI	FOP2
	8046	10		0100	LR	DC, H
	8047	2018		0101	LI	24
	8049	58		0102	LR	HFLG, A
	804A	28812A	812A	0103	FLOP	FOP1
	804D	11		0104	LR	H, DC
	804E	38		0105	DS	HFLG
	804F	94FA	804A	0106	BNZ	FLOP
				0107	*NOW	CHECKSUM
	8051	47		0108	LR	A, CKSM
	8052	18		0109	COM	
	8053	1F		0110	INC	
	8054	07		0111	LR	QL, A
	8055	78		0112	LIS	8
	8056	58		0113	LR	HFLG, A
	8057	28812C	812C	0114	PI	FOP2
	805A	2885F2	85F2	0115	PI	TTCR
				0116	*NOW	CHECK IF ALL MEMORY IS
	805D	4B		0117	LR	A, 11
	805E	C6		0118	AS	ERLO
	805F	4A		0119	LR	A, 10
	8060	19		0120	LNK	
	8061	C5		0121	AS	EAHI
	8062	92C1	8024	0122	BNC	FCON

## APPENDIX H (continued)

ERRS	LOC	OBJECT	ADDR	LINE	SOURCE STATEMENT
	8064	07		0123	LR QL,A ZERO
				0124	*
				0125	*NOW WROTE ZERO RECORD PRIOR TO TRAILER
				0126	*
	8065	203A		0127	LI H'3A' COLON
	8067	51		0128	LR CHR5,A
	8068	288610	8610	0129	PI TTY0 PUNCH VCOLON
	806B	78		0130	LIS 8
	806C	58		0131	LR HFLG,A
	806D	28812C	812C	0132	HERE PI FOP2 PUNCH 2 ZERO CHAR
	8070	38		0133	DS HFLG
	8071	94FB	806D	0134	BNZ HERE REPEAT IF NOT 0
	8073	38		0135	DS HFLG SET TO -1
	8074	9090	8005	0136	BR (FPUN+2) TRAILER
				0137	*
				0138	*TEST FOR KEYBOARD OR TTY
				0139	*
	8076	66		0140	TZE LISU 6
	8077	6E		0141	LISL 6
	8078	2080		0142	LI H'80'
	807A	50		0143	LR I,A
	807B	70		0144	LIS 0
	807C	50		0145	LR I,A
	807D	298160	8160	0146	JMP STRT
				0147	*
				0148	*
				0149	*
				0150	*
				0151	*
				0152	*ENTRY POINT TO SAVE ALL REGISTERS EXCEPT PC0
				0153	* IF ENTRY IS FROM RESET MOST REGISTERS
				0154	* WILL BE MEANINGLESS DUE TO SWITCH BOUNCE.
				0155	*
				0156	*
	8080	1A		0157	SAVER DI MUST DO
	8081	2720		0158	OUT PORTA SAVE ACC VALID OR NOT
	8083	71		0159	LIS 1
	8084	2728		0160	OUT OPOR MARK BIT FOR TTY TO KEEP QUIET
	8086	4A		0161	LR A,10
	8087	2724		0162	OUT PORTB SAVE R10 TEMP
	8089	4B		0163	LR A,11
	808A	2729		0164	OUT PORTC SAVE R11 TEMP
	808C	11		0165	LR H,DC FREE DC0
				0166	*SAVE R0-R63, IS, W
	808D	2A0000	0000	0167	DCI RAM
	8090	40		0168	LR A,0
	8091	17		0169	ST R0 SAVED
	8092	41		0170	LR A,1
	8093	17		0171	ST R1 SAVED
	8094	49		0172	LR A,9
	8095	51		0173	LR I,A R9 TEMP TO R1
	8096	1E		0174	LR J,W W TEMP TO R9
	8097	0A		0175	LR A,IS
	8098	50		0176	LR 0,A ISAR TEMP TO R0
				0177	*NOW SAVE R2 TO R63
	8099	72		0178	LIS 2
	809A	0B		0179	LR IS,A ISAR TO R2
	809B	4C		0180	ZC LR A,S
	809C	17		0181	ST
				0182	*INC ISAR
	809D	0A		0183	LR A,IS

## APPENDIX H (continued)

ERRS	LOC	OBJECT	ADDR	LINE	SOURCE STATEMENT
	809E	1F		0184	INC
	809F	0B		0185	LR IS, A
				0186	*TEST FOR H'40' IF DONE
	80A0	13		0187	SL 1
	80A1	81F9	809B	0188	BP ZC
				0189	*NOW SAVE ACC, IS, W FROM TEMPS
	80A3	2620		0190	IN PORTA
	80A5	17		0191	ST
	80A6	40		0192	LR A, 0
	80A7	17		0193	ST ISAR NOW SAVED
	80A8	49		0194	LR A, 9
	80A9	17		0195	ST W NOW SAVED
	80AA	16		0196	LM BAUD COUNT TO R0
	80AB	50		0197	LR 0, A MAYBE IT IS VALID
				0198	*NOW SAVE DC0, DC1, PC1
	80AC	2C		0199	XDC
	80AD	0E		0200	LR Q, DC DC1 TO Q
	80AE	2C		0201	XDC
	80AF	08		0202	LR K, P PC1 TO K
	80B0	7A		0203	LIS 10
	80B1	0B		0204	LR IS, A ISAR TO H, K, Q
	80B2	4D		0205	ZD LR A, 1
	80B3	17		0206	ST
	80B4	8FFD	80B2	0207	BR? ZD
	80B6	03		0208	LR A, QL BR? DID NOT FINISH
	80B7	17		0209	ST
	80B8	2A0009	0009	0210	DCI RA9
	80BB	41		0211	LR A, 1
	80BC	17		0212	ST R9 NOW SAVED
	80BD	2621		0213	IN PORTB
	80BF	17		0214	ST R10 NOW SAVED
	80C0	2629		0215	IN PORTC
	80C2	17		0216	ST R11 NOW SAVED
	80C3	0E		0217	LR Q, DC QU=RAM PAGE ADDRESS
				0218	*
				0219	*NOW TEST FOR TTY AND IF SO GET BAUD COUNT
				0220	*
	80C4	70		0221	CLR
	80C5	2728		0222	OUT IPOR
	80C7	2628		0223	IN IPOR PORT0 FROM DEBUG
	80C9	18		0224	COM
	80CA	2106		0225	NI 6 BITS 1, 2 FOR DEFAULT
	80CC	84A9	8076	0226	BZ TZE DEFAULT TO RAM VALUE
	80CE	2306		0227	XI 6
	80D0	12		0228	SR 1
	80D1	2A80DD	80DD	0229	DCI BAUDT TABLE
	80D4	8E		0230	ADC
	80D5	16		0231	LM
	80D6	2A0043	0043	0232	DCI RAM+67 STORE AREA FOR BAUD
	80D9	17		0233	ST
	80DA	50		0234	LR 0, A
	80DB	909A	8076	0235	ZE BR TZE
				0236	*
				0237	*
				0238	*
			80DD	0239	BAUDT EQU * NO 0 VALUE FOR TABLE
	80DD	06		0240	DC H'06' 110 BAUD VALUE
	80DE	A4		0241	DC H'A4' 300 BAUD VALUE
	80DF	EA		0242	DC H'EA' 1200 BAUD VALUE
				0243	*
				0244	* RETURN FROM BREAKPOINT



## APPENDIX H (continued)

ERRS	LOC	OBJECT	ADDR	LINE	SOURCE STATEMENT
				0245	*
	80E0	AC		0246	INS H'C' ACC SAVE
	80E1	909E	8080	0247	BR SAVER
				0248	*
				0249	*
				0250	*
				0251	*RESTORE BEFORE RETURN TO USER, ALL BUT ACC IS OK
				0252	*
				0253	* MOVE D0, P1, D1 TO H, K, Q
				0254	*
	80E3	2A0044	0044	0255	REST DCI D0
	80E6	7A		0256	LIS 10 HU
	80E7	0B		0257	LR IS, A
	80E8	16		0258	2A LM
	80E9	5D		0259	LR I, A
	80EA	8FFD	80E8	0260	BR7 2A
	80EC	16		0261	LM GET D1 LOW
	80ED	5D		0262	LR I, A TO QL WITH 'ISAR' TO 0
				0263	*RESTORE D1, P1
	80EE	60		0264	LISU 0
	80EF	0F		0265	LR DC, Q
	80F0	2C		0266	XDC TO DC1
	80F1	09		0267	LR P, K TO PC1
				0268	*RESTORE R0-R63
	80F2	2A0000	0000	0269	DCI RAM
	80F5	16		0270	2B LM
	80F6	5C		0271	LR S, A
	80F7	0A		0272	LR A, IS
	80F8	1F		0273	INC
	80F9	0B		0274	LR IS, A ISAR +1
	80FA	13		0275	SL 1
	80FB	81F9	80F5	0276	BP 2B IF NOT H'40' DO MORE
				0277	*NOW RESTORE IS, W
	80FD	16		0278	LM SKIP ACC
	80FE	16		0279	LM ISAR
	80FF	0B		0280	LR IS, A
	8100	16		0281	LM
	8101	59		0282	LR S, A W NOW IN J
				0283	*GET D0 IN H
	8102	16		0284	LM SKIP BAUD COUNT
	8103	16		0285	LM
	8104	5A		0286	LR 10, A
	8105	16		0287	LM
	8106	5B		0288	LR 11, A
				0289	*GET ICB BIT AND ADD TO DI INST
	8107	74		0290	LIS 4
	8108	8E		0291	ADC
	8109	49		0292	LR A, 9
	810A	14		0293	SR 4 ICB NOW BIT 0
	810B	241A		0294	AI H'1A' IF ICB ON DI BECOMES EI
	810D	17		0295	ST IN EXECUTION SEQUENCE
	810E	2029		0296	LI H'29' JMP OP CODE
	8110	17		0297	ST RETURN TO USER PC0
				0298	*NOW STRIP ICB FROM STATUS
	8111	7F		0299	LIS 15
	8112	F9		0300	NS 9
	8113	59		0301	LR S, A
				0302	*GET R9-11 AGAIN
	8114	2A0009	0009	0303	DCI RA9
	8117	16		0304	LM
	8118	2720		0305	OUT PORTA SAVE R9

## APPENDIX H (continued)

ERRS	LOC	OBJECT	ADDR	LINE	SOURCE STATEMENT
	811A	16		0306	LM
	811B	272	4	0307	OUT PORTB SAVE R10
	811D	16		0308	LM
	811E	10		0309	LR DC, H DC0 NOW RESTORED
	811F	58		0310	LR 11, A
	8120	262	4	0311	IN PORTE
	8122	5A		0312	LR 10, A
	8123	2620		0313	IN PORTA R9 TO ACC
	8125	1D		0314	LR W, J STATUS RESTORED
	8126	59		0315	LR 9, A FINALLY R9
	8127	29004A	004A	0316	JMP EX NOW EXECUTE EI OR DI THEN JMP
				0317	*
				0318	*
				0319	*
				0320	*
				0321	*
				0322	*****
				0323	*SUBROUTINE TO OUTBYTE 1 BYTE AS 2 4 BIT ASC CHARS
				0324	*CKSM IS ACCUMULATED
				0325	*HFLG MUST BE POSITIVE NUMBER
				0326	*
	812A	16		0327	FOP1 LM
	812B	07		0328	LR QL, A
				0329	*ENTRY WITH CHARACTER IN REG SAVE
	812C	08		0330	FOP2 LR K, P SAVE RETURN INK
	812D	48		0331	FLP1 LR A, HFLG
	812E	18		0332	COM SET STATUS
	812F	58		0333	LR HFLG, A FLIP-FLOP FLAG
	8130	03		0334	LR A, QL
	8131	9102	8134	0335	BM FHI DO HI 4 BITS IF -
	8133	15		0336	SL 4 LOSE HI 4 BITS
	8134	14		0337	FHI SR 4 MOVE TO LO 4 BITS
				0338	*CONVERT TO ASC 0='30', 9='39', A='41', F='46'
	8135	2430		0339	AI H'30'
	8137	2539		0340	CI H'39'
	8139	8103	813D	0341	BP FINT NOT A-F
	813B	2407		0342	AI 7
	813D	51		0343	FINT LR CHRS, A
	813E	288610	8610	0344	PI TTY0 TYPE FRAME
	8141	38		0345	DS HFLG
	8142	91EH	812D	0346	BM FLP1 DO LOW HALF WORD
				0347	*DO CHECKSUM
	8144	03		0348	LR A, QL LAST BYTE OUTPUTTED
	8145	C7		0349	AS CKSM ADD TO PROIR CKSM
	8146	57		0350	LR CKSM, A SAVE UPDATED CKSM
	8147	0C		0351	PK RETURN FROM K REGISTERS
				0352	*
				0353	*
	8148	28		0354	TBLS DC AL1(A1-A) ACCUMULATOR DISPLAY
	8149	B2		0355	DC AL1(B-A) PREVIOUS OR BACK 1 ADDRESS
	814A	33		0356	DC AL1(C-A) CHANGE FIELD
	814B	69		0357	DC AL1(D-A) DC DISPLAY
	814C	6F		0358	DC AL1(E-A) EXAMINE
	814D	75		0359	DC AL1(F-A) FORMATTED PUNCH
	814E	78		0360	DC AL1(G-A) GO TO
	814F	96		0361	DC AL1(H-A) HIGH SPEED LOADER
	8150	87		0362	DC AL1(I-A) ISAR DISPLAY
	8151	93		0363	DC AL1(J-A) 8 BIT PROM PUNCH
	8152	A2		0364	DC AL1(K-A) FIND BIT PATTERN
	8153	9C		0365	DC AL1(L-A) LOAD
	8154	9F		0366	DC AL1(M-A) MEMORY DISPLAY

## APPENDIX H (continued)

ERRS	LOC	OBJECT	ADDR	LINE	SOURCE	STATEMENT
	8155	B7		0367	DC	AL1(N-A) NEXT DISPLAY
	8156	99		0368	DC	AL1(O-A) PORT DISPLAY
	8157	C3		0369	DC	AL1(P-A) PC DISPLAY
	8158	FF		0370	DC	AL1(-1) 0
	8159	F7		0371	DC	AL1(R-A) REGISTER DISPLAY
	815A	F3		0372	DC	AL1(S-A) STATUS DISPLAY
	815B	2F		0373	DC	AL1(T-A) CLEAR BREAK POINT
	815C	31		0374	DC	AL1(U-A) BREAKPOINT
	815D	96		0375	DC	AL1(V-A) MOVE BLOCK
	815E	F3		0376	DC	AL1(S-A) W (STATUS) DISPLAY
	815F	01		0377	HXX DC	AL1(X-A) HEX ARITHMETIC
	8160	4B		0378	STRT LR	A, 11 FREE R11
	8161	07		0379	LR	QL, A
	8162	2885F2	85F2	0380	STR1 PI	TTCR WRITE CR/LF
	8165	203F		0381	LI	C'?' PROMPT CHARACTER
	8167	51		0382	LR	CHRS, A
	8168	288610	8610	0383	PI	TTY0
				0384		*READ INPUT CHARACTER
	816B	2885BE	85BE	0385	PI	TTY1
	816E	53		0386	LR	3, A SAVE CHAR
	816F	288610	8610	0387	PI	TTY0 ECHO CHAR
	8172	43		0388	LR	A, 3
	8173	217F		0389	NI	H'7F' 7 BITS ONLY
	8175	2540		0390	CI	H'40'
	8177	81EA	8162	0391	BP	STR1 LESS THAN 'A' INVALID
	8179	2558		0392	CI	C'X'
	817B	91E6	8162	0393	BM	STR1 GREATER THAN 'W' INVALID
	817D	211F		0394	NI	H'1F' SAVE 5 BITS ONLY
	817F	2A8147	8147	0395	DCI	(TBL5-1)
	8182	8E		0396	ADC	
	8183	16		0397	LM	GET TABLE VALUE
	8184	05		0398	RPTC LR	KL, A SAVE INDEX IN R13
	8185	1F		0399	INC	SET STATUS
	8186	84DB	8162	0400	BZ	STR1 INVALID CONTROL CHAR
				0401		*NOW INPUT PARAMETERS IF ANY
				0402		*FETCH PARAMETERS AFTER LEGIT CODE IS IN
				0403		*MAX FIELD 1 AND 2 IS 4 HEX DIGITS
				0404		*MAX FIELD 3 IS 1 HEX DIGIT
	8188	73		0405	LIS	3
	8189	58		0406	LR	FCNT, A PARAMETER COUNT
	818A	0B		0407	LR	IS, A ISAR=3
	818B	74		0408	FA	LIS 4
	818C	5B		0409	LR	CC, A CHAR CNT=4
				0410	*	
				0411		*ZERO DISPLAY R60-66
	818D	0A		0412	LR	A, IS
	818E	52		0413	LR	2, A TEMP
	818F	66		0414	LISU	6
	8190	6D		0415	LISL	5
	8191	70		0416	LIS	0
	8192	5E		0417	FAA	LR D, A
	8193	8FFE	8192	0418	BR7	FAA
	8195	42		0419	LR	A, 2
	8196	0B		0420	LR	IS, A RESTORE
				0421	*	
				0422		*READ CHARACTER
	8197	2885BE	85BE	0423	FB	PI TTY1 GET CHAR
	819A	217F		0424	NI	H'7F'
	819C	57		0425	LR	7, A
	819D	250D		0426	CI	CR
	819F	844E	81EE	0427	BZ	CORT CORRECT FIELD BOUNDARIES

## APPENDIX H (continued)

ERRS	LOC	OBJECT	ADDR	LINE	SOURCE	STATEMENT
	81A1	288610	8610	0428	PI	TTYO ECHO INPUT CHAR
	81A4	47		0429	LR	A, 7
	81A5	252D		0430	CI	C'-' FIELD SEPARATOR
	81A7	8445	81ED	0431	BZ	CORN
	81A9	252B		0432	CI	C'+'
	81AB	8441	81ED	0433	BZ	CORN
	81AD	253D		0434	CI	C'='
	81AF	843E	81EE	0435	BZ	CORT
	81B1	255B		0436	CI	H'5B' KILL CHAR?
	81B3	84AE	8162	0437	BZ	STR1 KILL INPUT
				0438		*MUST BE HEX 0-9, OR A-F
	81B5	252F		0439	CI	H'2F' LESS THAN ZERO
	81B7	81DF	8197	0440	BP	FB ERROR IGNORE
	81B9	2546		0441	CI	H'46' F?
	81BB	91DB	8197	0442	BM	FB ERROR IGNORE
	81BD	24D0		0443	AI	H'D0'
	81BF	2509		0444	CI	H'9'
	81C1	8107	81C9	0445	BP	FOK
	81C3	2510		0446	CI	H'10'
	81C5	81D1	8197	0447	BP	FB ERROR, IGNORE IT
	81C7	24F9		0448	AI	H'F9'
	81C9	3B		0449	DS	CC
	81CA	91CC	8197	0450	BM	FB FULL FIELD IGNORE THIS
				0451		*ZERO LO 4 BITS, THEN ADD THIS DIGIT
	81CC	52		0452	LR	2, A
				0453	*	
				0454		*SHIFT NEW DIGIT INTO DISPLAY
	81CD	0A		0455	LR	A, IS
	81CE	04		0456	LR	KU, A TEMP
	81CF	66		0457	LISU	6
	81D0	69		0458	LISL	1
	81D1	4E		0459	LR	A, D R61
	81D2	5D		0460	LR	I, A TO R60
	81D3	6A		0461	LISL	2
	81D4	4E		0462	LR	A, D R62
	81D5	5D		0463	LR	I, A TOR61
	81D6	6B		0464	LISL	3
	81D7	4E		0465	LR	A, D R63
	81D8	5D		0466	LR	I, A TO R62
	81D9	42		0467	LR	A, 2 NEW DIGIT
	81DA	5C		0468	LR	S, A
	81DB	00		0469	LR	A, KU
	81DC	0B		0470	LR	IS, A RESTORE
	81DD	4C		0471	LR	A, 5
	81DE	15		0472	SL	4
	81DF	C2		0473	AS	2
	81E0	5D		0474	LR	I, A TO PARAMETER LOC
	81E1	71		0475	LIS	1
	81E2	FB		0476	NS	CC
	81E3	84B3	8197	0477	BZ	FB EVEN CHAR, 2DIGITS, GO READ
	81E5	4E		0478	LR	A, D SET ISAR BACK 1
	81E6	8FB0	8197	0479	BR7	FB GET NEXT DIGIT
	81E8	73		0480	LIS	3
	81E9	9033	821D	0481	BR	COR7 PARAMETERS FULL, NOW EXIT
	81EB	909F	818B	0482	BR	FA LINK TO FA
	81ED	59		0483	CORN	LR 9, A
				0484		*CORRECT HEX FIELD, RIGHT JUSTIFY
				0485	*	IF AB0C THEN 0ABC
				0486	*	IF ABCD THEN IT IS OK
				0487	*	IF A00 THEN 000A
				0488	*	IF AB00 THEN 00AB

## APPENDIX H (continued)

ERRS	LOC	OBJECT	ADDR	LINE	SOURCE STATEMENT		
	81EE	74		0489	CORT	LIS	4
	81EF	EB		0490		XS	CC
	81F0	841F	8210	0491		BZ	COR6 NO PARAMETERS
	81F2	3B		0492		DS	CC
	81F3	911B	820F	0493		BM	COR5 FIELD ABCD, AOK
	81F5	940C	8202	0494		BNZ	COR1
	81F7	7F		0495		LIS	15
	81F8	FC		0496		NS	S
	81F9	5E		0497		LR	D, A
	81FA	4D		0498		LR	A, I
	81FB	15		0499		SL	4
	81FC	EC		0500		XS	S
	81FD	5E		0501		LR	D, A
	81FE	4C		0502		LR	A, S
	81FF	14		0503		SR	4
	8200	900C	820D	0504		BR	COR4
	8202	3B		0505	COR1	DS	CC
	8203	8405	8209	0506		BZ	COR2
	8205	7F		0507		LIS	15
	8206	FD		0508		NS	I
	8207	9003	820B	0509		BR	COR3
	8209	4E		0510	COR2	LR	A, D
	820A	4D		0511		LR	A, I
	820B	5E		0512	COR3	LR	D, A
	820C	70		0513		LIS	0
	820D	5D		0514	COR4	LR	I, A
	820E	4D		0515		LR	A, I
	820F	38		0516	COR5	DS	FCNT
	8210	47		0517	COR6	LR	A, 7
	8211	250D		0518		CI	CR
	8213	8405	8219	0519		BZ	COR8
	8215	253D		0520		CI	C'='
	8217	94D3	81EB	0521		BNZ	FAK LAST CHAR NOT CR OR =
	8219	48		0522	COR8	LR	A, FCNT
	821A	18		0523		COM	
	821B	2404		0524		RI	4
	821D	58		0525	COR7	LR	FCNT, A
	821E	03		0526		LR	A, QL
	821F	5B		0527		LR	11, A RESTORE R11
				0528			*TEST FOR 2 OR 3 PARAMETERS
				0529			* MAKE LO ADDRESS START ON 8 BYTE BOUNDARY
				0530			* MAKE HI ADDRESS END ON 8 BYTE BOUNDARY
	8220	72		0531		LIS	2
	8221	F8		0532		NS	FCNT
	8222	8418	823B	0533		BZ	RTN1 NOT 2 OR 3 PARAMETERS
	8224	01		0534		LR	A, KL
	8225	2596		0535		CI	<V-R> HEX
	8227	8422	824A	0536		BZ	MOVE
	8229	2575		0537		CI	<F-R> DUMP
	822B	840A	8236	0538		BZ	FF
	822D	12		0539		SR	1
	822E	840C	823B	0540		BZ	RTN1 HEX
	8230	46		0541		LR	A, 6
	8231	220F		0542		OI	15
	8233	56		0543		LR	6, A
	8234	20F0		0544		LI	H'F0'
	8236	F4		0545	FF	NS	4
	8237	54		0546		LR	4, A
	8238	5B		0547		LR	11, A
	8239	43		0548		LR	A, 3
	823A	5A		0549		LR	10, A

## APPENDIX H (continued)

ERRS	LOC	OBJECT	ADDR	LINE	SOURCE STATEMENT
				0550	*CALC ADDRESS TO GET TO PROCESS ROUTINE
	823B	2A8248	8248	0551	RTN1 DCI STB
	823E	16		0552	LM
	823F	04		0553	LR KU, A SAVE
	8240	01		0554	LR A, KL
	8241	88		0555	AM
	8242	05		0556	LR KL, A
	8243	00		0557	LR A, KU HI 8 BIT ADDRESS OF 'A'
	8244	19		0558	LNK
	8245	04		0559	LR KU, A
	8246	02		0560	LR A, QU HI ADDRESS IF NEEDED
	8247	0C		0561	PK
	8248	82CB		0562	STB DC AL2(HEX-1)
				0563	*MOVE MEMORY BLOCK
				0564	* TO ACCOMPLISH, DO MXXXXX WHERE XXXX IS DESTINATION START ADDRESS
				0565	* VSSSS-EEEE WHERE SSSS IS SOURCE ADDRESS START
				0566	* AND EEEE IS SOURCE ENDING ADDRESS
				0567	*INPUT
				0568	* R10-11 =DESTINATION START ADDRESS
				0569	* R3-4 =SOURCE START ADDRESS
				0570	* R5-6 =SOURCE END ADDRESS
				0571	*WORK REGISTERS
				0572	* R10-11 =CORRENT SOURCE BYTE ADDRESS
				0573	* R5-6 =SOURCE START IF MOVING FROM END FORWARD
				0574	* =SOURCE END ADDRESS IF MOVE START TO END
	824A	4A		0575	MOVE LR A, 10 DESTINATION HI BYTE
	824B	18		0576	COM
	824C	1F		0577	INC
	824D	C3		0578	AS 3 SOURCE START HI BYTE
	824E	840B	825A	0579	BZ MBY2 EQUAL, TEST LO BYTE
	8250	910F	8260	0580	BM MEND MOVE BACKWARDS
	8252	10		0581	MBGN LR DC, H
	8253	43		0582	LR A, 3
	8254	5A		0583	LR 10, A
	8255	44		0584	LR A, 4
	8256	5B		0585	LR 11, A
	8257	70		0586	LIS 0
	8258	9026	827F	0587	BR MCOM
	825A	4B		0588	MBY2 LR A, 11
	825B	18		0589	COM
	825C	1F		0590	INC
	825D	C4		0591	AS 4
	825E	81F3	8252	0592	BP MBGN
				0593	*CALC LENGTH OF BLOCK
	8260	43		0594	MEND LR A, 3
	8261	18		0595	COM
	8262	51		0596	LR 1, A
	8263	44		0597	LR A, 4
	8264	18		0598	COM
	8265	1F		0599	INC
	8266	1E		0600	LR J, W
	8267	C6		0601	AS 6
	8268	9202	826B	0602	BNC MXX
	826A	1E		0603	LR J, W
	826B	CB		0604	AS 11
	826C	5B		0605	LR 11, A
	826D	41		0606	LR A, 1
	826E	19		0607	LNK
	826F	1D		0608	LR W, J
	8270	19		0609	LNK
	8271	C5		0610	AS 5

## APPENDIX H (continued)

ERRS	LOC	OBJECT	ADDR	LINE	SOURCE STATEMENT
	8272	CA		0611	RS 10
	8273	5A		0612	LR 10,A
				0613	*MOVE TO DC THEN INCREMENT TO ADD 1 TO LENGTH
	8274	10		0614	LR DC,H
				0615	*MOVE SOURCE START ADDRESS TO R10-11
	8275	45		0616	LR A,5
	8276	5A		0617	LR 10,A
	8277	46		0618	LR A,6
	8278	58		0619	LR 11,A
				0620	*NOW MOVE SOURCE START TO R5-6 FOR END COMPARE
	8279	43		0621	LR A,3
	827A	55		0622	LR 5,A
	827B	44		0623	LR A,4
	827C	56		0624	LR 6,A
	827D	20FE		0625	LI -2 MEMORY ADDRESS INCREMENT
	827F	57		0626	MCOM LR 7,A
	8280	2C		0627	XDC
	8281	10		0628	LR DC,H
	8282	11		0629	MLOP LR H,DC SAVE ADDRESS FOR COMPARE
	8283	16		0630	LM
	8284	2C		0631	XDC
	8285	17		0632	ST
	8286	47		0633	LR A,7
	8287	8E		0634	ADC
	8288	2C		0635	XDC
	8289	8E		0636	ADC
				0637	*COMPARE FOR ADDRESS END
	828A	4A		0638	LR A,10
	828B	E5		0639	XS 5
	828C	94F5	8282	0640	BNZ MLOP
	828E	4B		0641	LR A,11
	828F	E6		0642	XS 6
	8290	94F1	8282	0643	BNZ MLOP
	8292	298160	8160	0644	BXX JMP STRT ALL DONE
				0645	*BREAKPOINT TO MEMORY SAVE 4 BYTES OF USER CODE.
				0646	* REPLACE WITH SAVE ACC IN PORT, THEN JUMP TO BRKPT
				0647	* PROCESS. BRKPT STAYS IN MEMORY UNTIL CLEARED.
	8295	20D0		0648	BRBK LI H'D0' SCRATCH MEM LO ADDRESS BYTE
	8297	07		0649	LR 0L,A
	8298	0F		0650	LR DC,Q
	8299	43		0651	LR A,3
	829A	5A		0652	LR 10,A
	829B	17		0653	ST
	829C	44		0654	LR A,4
	829D	58		0655	LR 11,A
	829E	17		0656	ST
				0657	*NOW FETCH USER BYTE
	829F	2C		0658	XDC LR DC,H
	82A0	10		0659	LR DC,H
	82A1	6B		0660	LISL 3
	82A2	16		0661	BLP LM
	82A3	2C		0662	XDC
	82A4	17		0663	ST
	82A5	2C		0664	XDC
	82A6	4E		0665	LR A,D DEC ISAR
	82A7	8FFA	82A2	0666	BR7 BLP
				0667	*NOW STORE IN MEMORY
	82A9	10		0668	LR DC,H
	82AA	20BC		0669	LI H'BC'
	82AC	17		0670	ST
				0671	*NOW CHANGE USER TO RESTORE ADDRESS

## APPENDIX H (continued)

ERRS	LOC	OBJECT	ADDR	LINE	SOURCE STATEMENT
	82AD	2029		0672	LI H'29' JMP INSTRUCTION
	82AF	17		0673	ST
	82B0	20B0		0674	LI H'00'
	82B2	17		0675	ST
	82B3	20E0		0676	LI H'E0' RETURN FROM BREAKPOINT
	82B5	17		0677	ST
	82B6	90DB	8292	0678	BR BXX TO START
				0679	*RESTORE BYTE TO USER MEMORY
	82B8	20D0		0680	TT LI H'D0' SCRATCH AREA
	82BA	07		0681	LR QL, A
	82BB	0F		0682	LR DC, Q
	82BC	16		0683	LM
	82BD	5A		0684	LR 10, A
	82BE	16		0685	LM
	82BF	5B		0686	LR 11, A
	82C0	2C		0687	XDC
	82C1	6B		0688	LISL 3
	82C2	10		0689	LR DC, H USER ADDRESS
	82C3	2C		0690	TTLP XDC
	82C4	16		0691	LM
	82C5	2C		0692	XDC
	82C6	17		0693	ST RESTORE USER CODE
	82C7	4E		0694	LR A, D
	82C8	8FFA	82C3	0695	BR7 TTLP
	82CA	90C7	8292	0696	BR BXX TO START
				0697	*HEX ARITHMETIC
			82CC	0698	X EQU *
	82CC	49		0699	HEX LR A, 9
	82CD	252B		0700	CI C'+'
	82CF	840B	82DB	0701	BZ PLUS
				0702	*COMPLEMENT BEFORE ADD=SUBTRACT
	82D1	46		0703	LR A, 6
	82D2	18		0704	COM
	82D3	1F		0705	INC 2'S COMPLEMENT
	82D4	1E		0706	LR J, W
	82D5	56		0707	LR 6, A
	82D6	45		0708	LR A, 5
	82D7	18		0709	COM
	82D8	1D		0710	LR W, J
	82D9	19		0711	LNK
	82DA	55		0712	LR 5, A
				0713	*ADDITION
	82DB	44		0714	PLUS LR A, 4
	82DC	C6		0715	AS 6
	82DD	54		0716	LR 4, A
	82DE	43		0717	LR A, 3
	82DF	19		0718	LNK
	82E0	C5		0719	AS 5
				0720	*NOW WRITE TO DISPLAY RESULTS
	82E1	07		0721	LR QL, A
	82E2	66		0722	LISU 6
	82E3	68		0723	LISL 0
	82E4	14		0724	SR 4
	82E5	5D		0725	LR I, A
	82E6	03		0726	LR A, QL
	82E7	5D		0727	LR I, A
	82E8	28812C	812C	0728	PI FOP2
	82EB	44		0729	LR A, 4
	82EC	07		0730	LR QL, A
	82ED	14		0731	SR 4
	82EE	5D		0732	LR I, A



## APPENDIX H (continued)

ERRS	LOC	OBJECT	ADDR	LINE	SOURCE STATEMENT	
	82EF	03		0733	LR	A, QL
	82F0	5D		0734	LR	I, A
	82F1	28812C	812C	0735	PI	FOP2
	82F4	909D	8292	0736	BXT	BR BXX
				0737	*PRINT ACCUMULATOR	
	82F6	2040		0738	AL	LI 64 LOC OF ACC
	82F8	905B	8354	0739	BR	I1
				0740	*CLEAR BREAK POINT ENTRY	
	82FA	906D	82B8	0741	T	BR TT
				0742	*BREAKPOINT ENTRY FROM TABLE	
	82FC	9098	8295	0743	U	BR BRBK TEMP NOP
				0744	*CHANGE LAST USED FIELD	
	82FE	38		0745	C	DS FCNT
	82FF	91F4	82F4	0746	BM	BXT NO PARAMETERS SUPPLIED
	8301	49		0747	LR	A, FLG
	8302	10		0748	LR	DC, H
	8303	250C		0749	CI	00 CK FOR PORT
	8305	9404	830A	0750	BNZ	C0 NOT PORT CHANGE
	8307	298546	8546	0751	JMP	PORT GO CHANGE PORT
	830A	250D		0752	C0	CI PP
	830C	910B	8318	0753	BM	C2 REGISTER CHANGE
	830E	9403	8312	0754	BNZ	C1 MEMORY CHANGE 1 WORD
	8310	43		0755	LR	A, 3
	8311	17		0756	ST	PC OR DC 1ST HALF
	8312	44		0757	C1	LR A, 4
	8313	17		0758	ST	
	8314	11		0759	LR	H, DC
	8315	4B		0760	LR	A, 11
	8316	900C	8323	0761	BR	(C3+4)
	8318	2080		0762	C2	LI H'80'
	831A	0B		0763	AS	11
	831B	07		0764	LR	QL, A
	831C	0F		0765	LR	DC, Q
	831D	44		0766	LR	A, 4
	831E	17		0767	ST	
	831F	4B		0768	C3	LR A, 11
	8320	1F		0769	INC	
	8321	213F		0770	NI	H'3F'
	8323	07		0771	LR	QL, A
	8324	2020		0772	C5	LI H'20' BLANK
	8326	51		0773	LR	CHRS, A
	8327	288610	8610	0774	PI	TTYO
	832A	2033		0775	LI	(C-A) TABLE VALUE OF C ENTRY
	832C	05		0776	LR	KL, A
	832D	298185	8185	0777	JMP	(RPTC+1)
	8330	46		0778	C55	LR A, 6
	8331	07		0779	LR	QL, A
	8332	90F1	8324	0780	BR	C5
				0781	*DISPLAY DC	
	8334	5A		0782	D	LR 10, A HI STORE ADDRESS
	8335	44		0783	LR	A, 4 DET LO ADDR OF DC
	8336	13		0784	SL	1 MEM0=2 OR 3
	8337	13		0785	SL	1 MAKE 4 OR 6
	8338	905C	8395	0786	BR	P1
				0787	*EXAMINE THIS ADDRESS	
	833A	7D		0788	E	LIS PP
	833B	E9		0789	XS	FLG
	833C	845D	839A	0790	BZ	P2 PC OR DC
	833E	9047	8386	0791	BR	N1
				0792	*FORMATTED PUNCH, FOR BOOT LOAD INPUT	
	8340	298003	8003	0793	F	JMP FPUN

## APPENDIX H (continued)

ERRS	LOC	OBJECT	ADDR	LINE	SOURCE	STATEMENT
				0794	*GO TO	
	8343	38		0795	G	DS FCNT
	8344	910A	834F	0796		BM G1
				0797	*GET NEW PC FROM PARAMETERS	
	8346	5A		0798	LR	10, A
	8347	20CC		0799	LI	H'CC'
	8349	58		0800	LR	11, A SET PC MEMORY ADDRESS
	834A	10		0801	LR	DC, H
	834B	43		0802	LR	A, 3
	834C	17		0803	ST	PC UPPER
	834D	44		0804	LR	A, 4
	834E	17		0805	ST	PC LOWER
	834F	2980E3	80E3	0806	G1 JMP	REST RESTORE REGISTERS , RETURN TO USER
				0807	*ISAR DISPLAY	
	8352	2041		0808	I	LI 65 REGISTER OF ISAR
	8354	54		0809	I1	LR 4, A
	8355	56		0810	LR	6, A
	8356	70		0811	LIS	0
	8357	53		0812	LR	3, A SET TO REGISTER FETCH
	8358	55		0813	LR	5, A
	8359	7F		0814	LIS	RR
	835A	59		0815	LR	FLG, A
	835B	2983F2	83F2	0816	JMP	R23
				0817	*	
	835E	298444	8444	0818	J	JMP PPUN
				0819	*	
			8361	0820	V	EQU * " JUST SO TABLE VALUE IS NOT FF
				0821	*HIGH SPEED READER LOAD ROUTINE	
	8361	2984B8	84B8	0822	H	JMP HIGH
				0823	*PORT PRINT	
	8364	7C		0824	O	LIS 00 CODE FOR PORT
	8365	905E	83C4	0825	BR	R1 PRINT PORT DATA
				0826	*BOOT LOAD	
	8367	2984AD	84AD	0827	L	JMP LOAD
				0828	*MEMORY PRINT	
	836A	7A		0829	M	LIS MM
	836B	9058	83C4	0830	BR	R1
				0831	*	
				0832	*FIND MATCHING BYTE	
				0833	*	
	836D	10		0834	K	LR DC, H
	836E	16		0835	LM	DUMMY TO SKIP BYTE
	836F	16		0836	F1	LM
	8370	E4		0837	XS	4
	8371	8414	8386	0838	BZ	N1 MATCH DECREMENT DC
	8373	11		0839	LR	H, DC
	8374	70		0840	LIS	0
	8375	EA		0841	XS	10
	8376	94F8	836F	0842	BNZ	F1 CONTINUE SEARCH
	8378	EB		0843	XS	11
	8379	94F5	836F	0844	BNZ	F1 CONTINUE SEARCH
	837B	900A	8386	0845	BR	N1 WRAPPED AROUND
				0846	*	
				0847	*DISPLAY PREVIOUS LOCATION	
				0848	*	
	837D	10		0849	B	LR DC, H
	837E	20FF		0850	LI	H'FF' -1
	8380	9003	8384	0851	BR	N2
				0852	*	
				0853	*NEXT--DISPLAY NEXT MEMORY OR REGISTER LOC	
	8382	10		0854	N	LR DC, H

## APPENDIX H (continued)

ERRS	LOC	OBJECT	ADDR	LINE	SOURCE	STATEMENT
	8383	71		0855	LIS	1
	8384	8E		0856	N2	ADC
	8385	11		0857	LR	H, DC
	8386	4A		0858	N1	LR A, 10
	8387	53		0859	LR	3, A
	8388	55		0860	LR	5, A
	8389	4B		0861	LR	A, 11
	838A	54		0862	LR	4, A
	838B	56		0863	LR	6, A
	838C	9044	83D1	0864	BR	R22
				0865	*PC PRINT	
	838E	5A		0866	P	LR 10, A
	838F	44		0867	LR	A, 4 PC0 OR 1
	8390	13		0868	SL	1
	8391	9403	8395	0869	BNZ	P1 PC1 OK
	8393	2408		0870	AI	8 TO REACH P0
	8395	24C4		0871	P1	AI H'C4' ADD PC0 OR PC1 OR DC
	8397	5B		0872	LR	11, A
	8398	7D		0873	LIS	PP
	8399	59		0874	LR	FLG, A SET TO P FOR 2 WORDS
	839A	10		0875	P2	LR DC, H
	839B	203D		0876	LI	C'=' EQUAL
	839D	51		0877	LR	CHRS, A
	839E	288610	8610	0878	PI	TTV0
	83A1	28812A	812A	0879	PI	FOP1 OP HEX WD, 2 CHAR FROM MEM
	83A4	66		0880	LISU	6
	83A5	68		0881	LISL	0
	83A6	03		0882	LR	A, QL
	83A7	14		0883	SR	4
	83A8	5D		0884	LR	I, A
	83A9	03		0885	LR	A, QL
	83AA	5D		0886	LR	I, A
	83AB	02		0887	LR	A, QU
	83AC	28812A	812A	0888	PI	FOP1 OP HEX WD, 2 CHAR
	83AF	03		0889	LR	A, QL
	83B0	14		0890	SR	4
	83B1	5D		0891	LR	I, A
	83B2	03		0892	LR	A, QL
	83B3	5D		0893	LR	I, A
	83B4	0F		0894	LR	DC, Q
	83B5	16		0895	LM	DATA @ PC/DC ADDRESS
	83B6	07		0896	LR	QL, A TEMP
	83B7	14		0897	SR	4
	83B8	5D		0898	LR	I, A
	83B9	03		0899	LR	A, QL
	83BA	5D		0900	LR	I, A
	83BB	298160	8160	0901	P3	JMP STRT
				0902	*S-FETCH	STATUS REGISTER
	83BE	2042		0903	S	LI 66 LOC OF STATUS STORE
	83C0	9093	8354	0904	BR	I1
				0905	*R-ENTRY	FOR REGISTER DISPLAY
	83C2	200F		0906	R	LI RR
	83C4	59		0907	R1	LR FLG, A SET TO R OR M
	83C5	38		0908	DS	FCNT
	83C6	91F4	83BB	0909	BM	P3 NO PARAMETERS GIVEN
	83C8	9405	83CE	0910	BNZ	R2
				0911	*MAKE HI	ADDR=LO ADDR
	83CA	43		0912	LR	A, 3
	83CB	55		0913	LR	5, A
	83CC	44		0914	LR	A, 4
	83CD	56		0915	LR	6, A

## APPENDIX H (continued)

ERRS	LOC	OBJECT	ADDR	LINE	SOURCE STATEMENT
				0916	*PRINT CR LF BLANK BLANK (R OR M) BLANK XXXX=
	83CE	2885F2	85F2	0917	R2 PI TTCR WRITE CR/LF
	83D1	2020		0918	R22 LI H'20' BLANK
	83D3	51		0919	LR CHRS, A
	83D4	288610	8610	0920	PI TTYO
	83D7	49		0921	LR A, FLG
	83D8	2443		0922	RI H'43'
	83DA	51		0923	LR CHRS, A
	83DB	288610	8610	0924	PI TTYO
	83DE	43		0925	LR A, 3 START LO ADDRESS
	83DF	07		0926	LR QL, A
	83E0	66		0927	LISU 6
	83E1	68		0928	LISL 0
	83E2	14		0929	SR 4
	83E3	5D		0930	LR I, A
	83E4	03		0931	LR A, QL
	83E5	5D		0932	LR I, A
	83E6	28812C	812C	0933	PI FOP2
	83E9	44		0934	LR A, 4
	83EA	07		0935	LR QL, A
	83EB	14		0936	SR 4
	83EC	5D		0937	LR I, A
	83ED	03		0938	LR A, QL
	83EE	5D		0939	LR I, A
	83EF	28812C	812C	0940	PI FOP2
	83F2	203D		0941	R23 LI C'=
	83F4	51		0942	LR CHRS, A
	83F5	288610	8610	0943	PI TTYO
	83F8	903D	8436	0944	BR BLNK
				0945	*NOW PRINT DATA
	83FA	43		0946	R4 LR A, 3
	83FB	5A		0947	LR 10, A
	83FC	44		0948	LR A, 4
	83FD	5B		0949	LR 11, A
	83FE	49		0950	LR A, FLG
	83FF	250C		0951	CI 00 PORT?
	8401	846A	846C	0952	BZ P10 GO TO PORT ROUTINE
	8403	250F		0953	CI RR TEST FOR REGISTER CHANGE
	8405	9407	840D	0954	BNZ R5 MEMORY LOC
				0955	*
				0956	*REGISTER FETCH IS IT IN SCRATCH OR TEMP MEMORY
	8407	44		0957	LR A, 4
				0958	*UPDATE REG 0-15 ADDRESS TO MEM LOC
	8408	2480		0959	RI H'80' LO RAM-START OF REG 0
	840A	5B		0960	LR 11, A
	840B	02		0961	LR A, QU HI ADDRESS
	840C	5A		0962	LR 10, A HI RAM ADDRESS
				0963	*FETCH DATA FROM MEMORY
	840D	10		0964	R5 LR DC, H
				0965	*DATA NOW IN NOW PRINT
	840E	28812A	812A	0966	PI FOP1 TYPE 2 CHAR FROM MEM
	8411	66		0967	R77 LISU 6
	8412	6C		0968	LISL 4
	8413	03		0969	LR A, QL
	8414	14		0970	SR 4
	8415	5D		0971	LR I, A
	8416	03		0972	LR A, QL
	8417	5D		0973	LR I, A
	8418	2020		0974	LI H'20' BLANK
	841A	51		0975	LR CHRS, A
	841B	288610	8610	0976	PI TTYO BLANK

## APPENDIX H (continued)

ERRS	LOC	OBJECT	ADDR	LINE	SOURCE STATEMENT
				0977	*CHECK FOR END
	841E	44		0978	LR R, 4
	841F	58		0979	LR 11, A SAVE FOR REENTRY
	8420	E6		0980	XS 6
	8421	43		0981	LR A, 3
	8422	5A		0982	LR 10, A SAVE FOR REENTRY
	8423	9404	8428	0983	BNZ R7
	8425	E5		0984	XS 5
	8426	8494	83BB	0985	BZ P3
	8428	44		0986	R7 LR R, 4
				0987	*UPDATE FOR NEXT WORD, LINE HAS MAX OF 8
	8429	1F		0988	INC
	842A	54		0989	LR 4, A INCREMENT ADDRESS
	842B	43		0990	LR A, 3
	842C	19		0991	LNK
	842D	53		0992	LR 3, A INCR HI ADDRESS
	842E	7F		0993	LIS 15
	842F	F4		0994	NS 4
	8430	849D	83CE	0995	BZ R2 NEW LINE
				0996	*ADD 2 BLANKS BETWEEN EACH 4 PRINTS
	8432	2103		0997	NI 3
	8434	94C5	83FA	0998	BNZ R4
	8436	2020		0999	BLNK LI H'20' BLANK
	8438	51		1000	LR CHR5, A
	8439	288610	8610	1001	PI TTYO
	843C	2020		1002	LI H'20'
	843E	51		1003	LR CHR5, A
	843F	288610	8610	1004	PI TTYO
	8442	90B7	83FA	1005	BR R4 CONTINUE THIS LINE
				1006	*PUNCH HEADER, THEN 256 X 8.
				1007	* THEN HEADER 256 X 8.
				1008	* CONTINUE AS ABOVE UNTIL LAST ADDRESS IS COMPLETE
				1009	* HFLG INITIALLY=SIZE, WHEN=0 PUNCH LEADER, WHEN=-SIZE DONE
	8444	70		1010	PPUN LIS 0
	8445	58		1011	LR 11, A LOW DC ADDRESS
	8446	2092		1012	LI H'92'
	8448	51		1013	LR CHR5, A
	8449	288610	8610	1014	PI TTYO
	844C	73		1015	PPA LIS 3
	844D	58		1016	LR HFLG, A NEG SIZE MINUS 1
	844E	F7		1017	NS SIZE MASK HI BITS
	844F	57		1018	LR SIZE, A
	8450	43		1019	LR A, SA
	8451	5A		1020	LR 10, A
	8452	10		1021	LR DC, H
	8453	47		1022	LR A, SIZE LENGTH OF BLOCK X 256
	8454	54		1023	LR 4, A
	8455	2030		1024	PBLK LI H'30' 48 BLANKS
	8457	56		1025	LR XFLG, A
	8458	70		1026	PLP LIS 0
	8459	51		1027	LR CHR5, A
	845A	288610	8610	1028	PI TTYO WRITE BLANK
	845D	36		1029	DS XFLG
	845E	94F9	8458	1030	BNZ PLP CONTINUE BLANKS
				1031	*CHECK FOR DONE
	8460	38		1032	DS HFLG
	8461	8134	8496	1033	BP PPC
				1034	*****DONE*****
	8463	2094		1035	LI H'94'
	8465	51		1036	LR CHR5, A
	8466	288610	8610	1037	PLXX PI TTYO PUNCH OFF

## APPENDIX H (continued)

ERRS	LOC	OBJECT	ADDR	LINE	SOURCE	STATEMENT
	8469	298076	8076	1038	JMP	TZE
				1039	*POR T DISPLAY	
	846C	20E0		1040	PIO	LI H'E0' SCRATCH MEMORY
	846E	07		1041	LR	QL, A
	846F	0F		1042	LR	DC, Q SCRATCH MEMORY
	8470	44		1043	LR	A, 4 PORT ADDRESS
	8471	250F		1044	CI	15 SHORT PORT ADDRESS
	8473	821E	8492	1045	BC	INS BUILD INS INSTRUCT
	8475	2026		1046	LI	H'26' LONG INPUT
	8477	17		1047	ST	
	8478	44		1048	LR	A, 4 PORT ADDRESS
	8479	17		1049	ST	
	847A	77		1050	PXX	LIS 7
	847B	17		1051	ST	
	847C	2029		1052	LI	H'29' JMP INST
	847E	17		1053	ST	
	847F	2C		1054	XDC	
	8480	2A848B	848B	1055	DCI	PXA
	8483	16		1056	LM	
	8484	2C		1057	XDC	
	8485	17		1058	ST	
	8486	2C		1059	XDC	
	8487	16		1060	LM	
	8488	2C		1061	XDC	
	8489	17		1062	ST	
	848A	00		1063	LR	P0, Q GO EXECUTE
				1064	*	
	848B	848D		1065	PXA	DC AL2(PXXY) ADDRESS OF RETURN
				1066	*	
				1067	*RETURN AFTER READ	
	848D	28812C	812C	1068	PXXY	PI FOP2 TO PRINT
	8490	9080	8411	1069	BR	R77
				1070	*SHORT INS	
	8492	24A0		1071	INS	AI H'A0'
	8494	90E4	8479	1072	BR	(PXX-1)
				1073	*OUTPUT 2 RUBOUTS, REGISTER CHRS	HAS -1 AFTER
				1074	* RETURN FROM TTYO OF BLANKS	
				1075	* PUNCH BLOCK OF 256	
	8496	70		1076	PPC	LIS 0
	8497	56		1077	LR	XFLG, A
				1078	*8 BIT FORMAT	
	8498	16		1079	EGHT	LM
	8499	51		1080	LR	CHRS, A
	849A	288610	8610	1081	PI	TTYO OUTPUT BYTE
	849D	36		1082	DS	XFLG
	849E	94F9	8498	1083	BN2	EGHT NEXT BYTE
	84A0	34		1084	DS	4 PAGES PER BLOCK
	84A1	81F4	8496	1085	BP	PPC START NEXT BLOCK
	84A3	58		1086	LR	HFLG, A
				1087	*UPDATE TO NEXT PAGE	
	84A4	43		1088	LR	A, SA
	84A5	1F		1089	INC	
	84A6	C7		1090	AS	SIZE
	84A7	53		1091	LR	SA, A
	84A8	E5		1092	XS	EA END PAGE
	84A9	84AB	8455	1093	BZ	PBLK DO TRAILER
	84AB	90A0	844C	1094	BR	PPH START NEXT HI BLOCK
				1095	* START OF BOOT LOAD	
	84AD	2091		1096	LOAD	LI H'91' READER ON COMMAND
	84AF	51		1097	LR	CHRS, A PASS IT
	84B0	288610	8610	1098	PI	TTYO AND TYPE IT

## APPENDIX H (continued)

ERRS	LOC	OBJECT	ADDR	LINE	SOURCE	STATEMENT
	84B3	2A85BE	85BE	1099	DCI	TTYI SERIAL INPUT PROCESS
	84B6	900F	84C6	1100	BR	BOT1 SKIP OVER PARALLEL CODE
	84B8	2A85A3	85A3	1101	HIGH DCI	PINP
	84BB	70		1102	LIS	0
	84BC	2729		1103	SLF1	OUT PPRT INITIALIZE PARALLEL IN PORT
	84BE	2728		1104	OUT	PSTS INIT CONTROL PORT FOR READER OFF
	84C0	2628		1105	IN	PSTS
	84C2	12		1106	SR	1
	84C3	15		1107	SL	4
	84C4	91F7	84BC	1108	BM	SLF1 LOCK IN LOOP TIL READER READY
	84C6	70		1109	BOT1	LIS 0
	84C7	56		1110	LR	XFLG,A CLEAR FIRST X DETECT FLAG
	84C8	0E		1111	LR	0,DC SET TO TTYI OR PB00 ENTRY
	84C9	57		1112	LR	CKSM,A
	84CA	2885A2	85A2	1113	IDLE	PI CHAR GET HEADER CHARACTER
	84CD	13		1114	SL	1 CLEAR PARITY BIT
	84CE	12		1115	SR	1
	84CF	253A		1116	CI	H'3A' COLON
	84D1	8449	851B	1117	BZ	FORM FORMULATOR FORMAST
	84D3	2553		1118	CI	C'S' IS IT AN LOAD ADDRESS?
	84D5	8431	8507	1119	BZ	SETA
	84D7	252A		1120	CI	C'*' IS IT THE END OF THE TAPE?
	84D9	8438	8512	1121	BZ	ENDX
	84DB	2358		1122	XI	C'X' WELL, IF IT ISN'T AN X, THEN LET'S GO
	84DD	94EC	84CA	1123	BNZ	IDLE
				1124	*****	HAVE THE START OF A DATA LINE *****
	84DF	57		1125	LR	CKSM,A INITIALIZE CHK SUM (ACC=0)
	84E0	78		1126	LIS	H'08'
	84E1	55		1127	LR	CCNT,A INITIALIZE BYTE COUNT TO 8
	84E2	56		1128	LR	XFLG,A SHOW THAT X HAS BEEN DETECTED
	84E3	28857B	857B	1129	CONT	PI BYTE
	84E6	17		1130	ST	STORE THE BYTE
	84E7	35		1131	DS	CCNT AND DECREMENT BYTE COUNT
	84E8	94FA	84E3	1132	BNZ	CONT
	84EA	2885A2	85A2	1133	PI	CHAR GET CHK CHAR FROM TAPE
	84ED	213F		1134	NI	H'3F' MASK TO SIX BITS
	84EF	24D0		1135	AI	H'D0' ASCII CONVERT-- FIRST 0-9
	84F1	8203	84F5	1136	BC	**4 CARRY IF IT WAS TWEN A AND F
	84F3	2439		1137	AI	H'39' FINISH CONVERSION OF A TO F
	84F5	E7		1138	XS	CKSM MAKE THE COMPARE
	84F6	15		1139	SL	4 LO 4 BITS TO HI
	84F7	84D2	84CA	1140	BZ	IDLE IF OK, LET'S GET SOME MORE
				1141	*****	CHK SUM ERROR HALT *****
	84F9	2043		1142	SLF2	LI C'C'
	84FB	51		1143	LR	CHRS,A
	84FC	288610	8610	1144	PI	TTYO
	84FF	204B		1145	LI	C'K'
	8501	51		1146	LR	CHRS,A
	8502	288610	8610	1147	PI	TTYO
	8505	900F	8515	1148	BR	STPP
	8507	28857B	857B	1149	SETA	PI GET NEW LOAD ADDRESS FROM TAPE
	850A	5A		1150	LR	10,A
	850B	28857B	857B	1151	PI	BYTE
	850E	5B		1152	LR	11,A
	850F	10		1153	LR	DC,H SET THE ADDRESS INTO DC
	8510	90B9	84CA	1154	BR	IDLE
	8512	F6		1155	ENDX	NS XFLG HAVE AN *, BUT DOES IT FOLLOW ANY DATA ?
	8513	84B6	84CA	1156	BZ	IDLE MUST BE ONLY AT THE HEAD OF THE TAPE
				1157	*	WAS 2A+0 OR 2A+8
	1158	*****		1158	*****	HALT LOOP FOR WHEN FINISHED *****
	8515	2093		1159	STPP	LI H'93' READER OFF COMMAND

## APPENDIX H (continued)

ERRS	LOC	OBJECT	ADDR	LINE	SOURCE STATEMENT		
	8517	51		1160	LR	CHRS, A	PASS IT IN CHRS
	8518	298466	8466	1161	JMP	PLXX	
	851B	70		1162	FORM	LIS	0
	851C	57		1163	LR	CKSM, A	ZERO CKSM TO START
	851D	18		1164	COM		
	851E	56		1165	LR	XFLG, A	
	851F	28857B	857B	1166	PI	BYTE	LENGTH OF BLOCK
	8522	55		1167	LR	CCNT, A	
	8523	F5		1168	NS	CCNT	
	8524	84F0	8515	1169	BZ	STPP	ZERO RECORD IS END
	8526	28857B	857B	1170	PI	BYTE	ADDRESS HI
	8529	5A		1171	LR	10, A	
	852A	28857B	857B	1172	PI	BYTE	ADDRESS LO
	852D	5B		1173	LR	11, A	
	852E	10		1174	LR	DC, H	
	852F	28857B	857B	1175	PI	BYTE	CODE BYTE
	8532	FB		1176	NS	CHR1	
	8533	9496	84CA	1177	BNZ	IDLE	NOT DATA RECORD
	8535	28857B	857B	1178	FMLP	PI	DATA FETCH
	8538	17		1179	ST		
	8539	35		1180	DS	CCNT	LENGTH
	853A	94FA	8535	1181	BNZ	FMLP	LOOP FOR MORE DATA
	853C	28857B	857B	1182	PI	BYTE	FETCH CKSM
	853F	47		1183	LR	A, CKSM	
	8540	F7		1184	NS	CKSM	SHOULD BE ZERO
	8541	94B7	84F9	1185	BNZ	SLF2	CKSM ERROR
	8543	2984CA	84CA	1186	JMP	IDLE	GET NEXT BLOCK
				1187	*PORT CHANGE ROUTINE		
	8546	20DF		1188	PORT	LI	H'DF'
	8548	07		1189	LR	QL, A	
	8549	0F		1190	LR	DC, Q	
	854A	46		1191	LR	A, 6	
	854B	5B		1192	LR	11, A	
	854C	250F		1193	CI	15	
	854E	821F	856E	1194	BC	OUTS	DO SHORT OUTS
	8550	2027		1195	LI	H'27'	
	8552	17		1196	ST		
	8553	46		1197	LR	A, 6	
	8554	17		1198	ST		
	8555	2044		1199	LI	H'44'	
	8557	17		1200	ST		
	8558	2027		1201	LI	H'27'	
	855A	17		1202	ST		
	855B	46		1203	LR	A, 6	
	855C	17		1204	JMP	ST	
	855D	2029		1205	LI	H'29'	
	855F	17		1206	ST		
	8560	2C		1207	XDC		
	8561	2A8579	8579	1208	DCI	RETX	
	8564	16		1209	LM		
	8565	2C		1210	XDC		
	8566	17		1211	ST		
	8567	2C		1212	XDC		
	8568	16		1213	LM		
	8569	2C		1214	XDC		
	856A	17		1215	ST		
	856B	2000		1216	LI	0	
	856D	0D		1217	LR	P0, Q	NOW EXECUTE
	856E	2480		1218	OUTS	AI	H'B0'
	8570	17		1219	ST		
	8571	2044		1220	LI	H'44'	



## APPENDIX H (continued)

ERRS	LOC	OBJECT	ADDR	LINE	SOURCE	STATEMENT
	8573	17		1221	ST	
	8574	20B0		1222	LI	H'B0'
	8576	C6		1223	AS	6
	8577	90E4	855C	1224	BR	JMP
				1225	*	
	8579	8330		1226	RETX	DC AL2(C55) RETURN ADDRESS
				1227	***GET A BYTE***	
				1228	***	GETS THE BYTE, CONVERTS, AND ADDS TO CHK SUM
	857B	08		1229	BYTE	LR K,P SAVE PC1
	857C	72		1230	LIS	2
	857D	58		1231	LR	HFLG,A SET THE HALF FLAG
	857E	4B		1232	AGAN	LR A,CHR1
	857F	15		1233	SL	4
	8580	5B		1234	LR	CHR1,A
	8581	2885A2	85A2	1235	PI	CHAR
	8584	213F		1236	NI	H'3F' MASK TO 6 BITS
	8586	24D0		1237	AI	H'D0' ASCII CONVERT-- FIRST 0-9
	8588	8203	858C	1238	BC	**+4
	858A	2439		1239	AI	H'39' NEXT CLEAN UP A-F
	858C	CB		1240	AS	CHR1
	858D	5B		1241	LR	CHR1,A TEMP STORE
	858E	46		1242	LR	A,XFLG
	858F	F6		1243	NS	XFLG TEST FOR NEGATIVE
	8590	9104	8595	1244	BM	AND NEGATIVE=FORMULATOR
	8592	4B		1245	LR	A,CHR1
	8593	C7		1246	AS	CKSM ADD NEW CHAR TO CHK SUM
	8594	57		1247	LR	CKSM,A
	8595	38		1248	AND	DS HFLG DECREMENT HALF COUNT
	8596	94E7	857E	1249	BNZ	AGAN GET 2ND HALF
	8598	46		1250	LR	A,XFLG
	8599	F6		1251	NS	XFLG
	859A	8104	859F	1252	BP	ADON IF FORMULATOR
	859C	47		1253	LR	A,CKSM
	859D	CB		1254	AS	CHR1
	859E	57		1255	LR	CKSM,A
	859F	4B		1256	ADON	LR A,CHR1 ONLY HAVE UPPER HALF--GO BACK
	85A0	09		1257	LR	P,K RESTORE PC1
	85A1	1C		1258	POP	
				1259	*****	PARALLEL AND SERIAL INPUT ROUTINES *****
				1260	**	COMMON CALL IS A PUSH TO CHAR
				1261	**	CHAR USES Q TO JUMP TO APPROPRIATE ROUTINE
	85A2	0D		1262	CHAR	LR P0,Q JUMP TO INPUT ROUTINE INDIRECTLY THRU Q REG
				1263	**	PINP: GET A CHARACTER FROM PORT 4
				1264	**	TYPICALLY USED WITH TAPE READER, BUT HAS A HANDSHAKING
				1265	**	DISCIPLINE THAT IS APPLICABLE TO OTHER DEVICES SUCH AS FIFOS
				1266	**	LOOKS FOR A CHARACTER READY INPUT, AND THEN GETS THE
				1267	**	CHARACTER. NEXT CPU GIVES AN ADVANCE PULSE THAT IS REMOVED
				1268	**	AFTER THE DEVICE READY INPUT GOES NOT READY.
	85A3	2628		1269	PINP	IN PSTS GET A CHAR FROM 300CPS READER
	85A5	13		1270	SL	1
	85A6	91FC	85A3	1271	BM	PINP LOOK FOR SPOCKET= HIGH
	85A8	7F		1272	LIS	H'0F' 100 US DELAY AFTER SEEING SPOCKET
	85A9	18		1273	COM	
	85AA	1F		1274	PDLY	INC
	85AB	94FE	85AA	1275	BNZ	PDLY
	85AD	2629		1276	IN	PPRT AND NOW GET DATA BYTE
	85AF	18		1277	COM	
	85B0	51		1278	LR	CHRS,A TEMP STORE CNEW CHAR
	85B1	79		1279	LIS	9 ADVANCE READER TO NEXT CHAR
	85B2	2728		1280	OUT	PSTS
	85B4	2628		1281	IN	PSTS GET READER STATUS

## APPENDIX H (continued)

ERRS	LOC	OBJECT	ADDR	LINE	SOURCE STATEMENT
	85B6	13		1282	SL 1
	85B7	81FC	85B4	1283	BP NOSP AND LOOK FOR MOVING OFF SPOCKET
	85B9	71		1284	LIS 1
	85BA	2728		1285	OUT PSTS REMOVE DRIVE PULSE NOW THAT IT IS MOVING
	85BC	41		1286	LR A,CHRS PICK BACK UP THE NEW CHARACTER
	85BD	1C		1287	POP
				1288	***TTYI: SERIAL I/P CHARACTER RETURNED IN ACC AND REG 1
				1289	** REG BCNT HOLDS BIT COUNT REG CHRS HOLDS CHARACTER
	85BE	2628		1290	TTYI IN IPOR
	85C0	2120		1291	NI H'20'
	85C2	8404	85C7	1292	BZ TTX
	85C4	298636	8636	1293	JMP SCAN
	85C7	2628		1294	TTX IN IPOR
	85C9	91FD	85C7	1295	BM TTX LOOK FOR START BIT
	85CB	40		1296	LR A,BAUD GET DELAY COUNT
	85CC	9001	85CE	1297	DLY3 BR **2 SILLY BRANCH FOR DELAY
	85CE	2401		1298	AI H'01'
	85D0	94FB	85CC	1299	BNZ DLY3 THIS LOOP IS HALF AS MUCH DELAY
	85D2	2628		1300	IN IPOR CHECK START BIT VALIDITY
	85D4	91E9	85BE	1301	BM TTYI
	85D6	79		1302	LIS 9
	85D7	52		1303	LR BCNT,A SET BIT COUNT,9DATA+1 STOP
	85D8	2180		1304	LOOP NI H'80' MASK TO GET INPUT BIT ONLY
	85DA	C1		1305	AS CHRS (LD'G START BIT WILL CLR GARBAGE)
	85DB	32		1306	DS BCNT DROP BIT CNT: 0 IF LAST DATA, NEG IF STOP
	85DC	9114	85F1	1307	BM STOP NEG IF LOOKING FOR STOP BIT
	85DE	8402	85E1	1308	BZ LOP2 IF LAST BIT, DO NOT SHIFT
	85E0	12		1309	SR 1 SHIFT ASSEMBLED CHARACTER TO MAKE ROOM
	85E1	51		1310	LOOP2 LR CHRS,A STORE ASSEMBLED CHARACTER
	85E2	40		1311	LR A,BAUD START OF FULL BIT TIME DELAY
	85E3	27EB		1312	DLY4 OUT NUSE NOP FOR DELAY
	85E5	27EB		1313	OUT NUSE
	85E7	27EB		1314	OUT NUSE
	85E9	2401		1315	AI H'01' INCR WITH A 50S INST
	85EB	94F7	85E3	1316	BNZ DLY4
	85ED	2628		1317	IN IPOR GET NEW BIT
	85EF	90E8	85D8	1318	BR LOOP
	85F1	1C		1319	STOP POP
				1320	***** SERIAL OUTPUT ROUTINE *****
				1321	** HAS 1 START, 8 DATA, 2 STOP. USES PORT 0, BITS 0 AND 7
				1322	** BAUD RATE IS SET BY A DELAY COUNT IN REG BAUD
				1323	** CALL BY PUTTING CHAR IN REG CHRS, AND SETTING DELAY IN BAUD
				1324	** ROUTINE RETURNS WITH ALL 1'S IN REG CHRS, REG BAUD INTACT
	85F2	2628		1325	TTCR IN IPOR
	85F4	2120		1326	NI H'20'
	85F6	8402	85F9	1327	BZ TCRX
	85F8	1C		1328	POP
	85F9	08		1329	TCRX LR K,P
	85FA	7D		1330	LIS CR
	85FB	51		1331	LR CHRS,A
	85FC	288610	8610	1332	PI TTY0
	85FF	7A		1333	LIS LF
	8600	51		1334	LR CHRS,A
	8601	288610	8610	1335	PI TTY0
	8604	2062		1336	LI 98
	8606	51		1337	LR 1,A
	8607	70		1338	LIS 0
	8608	52		1339	LR 2,A
	8609	32		1340	PIX DS 2
	860A	94FE	8609	1341	BNZ PIX
	860C	31		1342	DS 1

## APPENDIX H (continued)

ERRS	LOC	OBJECT	ADDR	LINE	SOURCE	STATEMENT
	8600	94FB	8609	1343	BNZ	PIX
	860F	0C		1344	PK	RETURN
	8610	2628		1345	TTY0	IN IPOR
	8612	2120		1346	NI	H'20' KEYBOARD BIT
	8614	8402	8617	1347	BZ	TTYX
	8616	1C		1348	POP	NO OUTPUT TO TTY
	8617	7B		1349	TTYX	LIS H'0B'
	8618	52		1350	LR	BCNT, A SET BIT COUNT FOR 11 BITS
	8619	70		1351	LIS	0
	861A	2728		1352	OUT	OPOR OUTPUT START BIT
				1353	***	DELAY ROUTINE-- 3.3MS FOR 300 BAUD, 9 MS FOR 110 BAUD
	861C	40		1354	DLY1	LR A, BAUD GET DELAY COUNT
	861D	27EB		1355	DLY2	OUT NUSE NOP FOR DELAY
	861F	27EB		1356	OUT	NUSE
	8621	27EB		1357	OUT	NUSE
	8623	2401		1358	RI	H'01' INCR WITH A 5 US INST
	8625	94F7	861D	1359	BNZ	DLY2
	8627	32		1360	DS	BCNT
	8628	9402	862B	1361	BNZ	DLY5
	862A	1C		1362	POP	
	862B	71		1363	DLY5	LIS 1
	862C	F1		1364	NS	CHRS
	862D	2728		1365	OUT	OPOR
	862F	41		1366	LR	A, CHRS
	8630	12		1367	SR	1
	8631	2400		1368	RI	H'00'
	8633	51		1369	LR	CHRS, A
	8634	90E7	861C	1370	BR	DLY1 NOW DELAY, THEN NEXT BIT?
				1371	*	SCAN KEYBOARD SUBROUTINE
	8636	0A		1372	SCAN	LR A, IS
	8637	65		1373	LISU	5
	8638	6F		1374	LISL	7
	8639	5E		1375	LR	D, A
				1376	*	LOAD KEY CODE
	863A	65		1377	A39	LISU 5
	863B	6E		1378	LISL	6
	863C	2017		1379	LI	H'17'
	863E	52		1380	LR	2, A
				1381	*	LOAD FIRST STROBE
	863F	20FE		1382	LI	H'FE'
	8641	5C		1383	LR	S, A
				1384	*	OUTPUT STROBE
	8642	7F		1385	A40	LIS 15 BLANK DISPLAY
	8643	2721		1386	OUT	P9
	8645	20C0		1387	LI	H'C0'
	8647	2720		1388	OUT	P8
	8649	4E		1389	LR	A, D
	864A	2720		1390	OUT	P8
				1391	*	READ KEYS
	864C	74		1392	LIS	4
	864D	5C		1393	LR	S, A
	864E	2621		1394	IN	P9
	8650	14		1395	SR	4
	8651	8417	8669	1396	BZ	A43 BRANCH IF NO KEY IN THIS COLUMN
	8653	15		1397	SL	4
	8654	9100	865D	1398	A41	BM A42 BRANCH IF KEY IS IN THIS ROW
				1399	*	GO TO NEXT ROW IF NOT LAST ROW
	8656	32		1400	DS	2
	8657	3C		1401	DS	5
	8658	8414	866D	1402	BZ	A44 BRANCH IF LAST ROW IN THIS COLUMN
	865A	13		1403	SL	1

## APPENDIX H (continued)

ERRS	LOC	OBJECT	ADDR	LINE	SOURCE STATEMENT
	865B	90F8	8654	1404	BR A41
				1405	* IS THIS FIRST SCAN
	865D	6C		1406	A42 LISL 4
	865E	71		1407	LIS 1
	865F	EC		1408	XS 5
	8660	941D	867E	1409	BNZ A48 BRANCH IF KBD FLAG NOT SET
				1410	* RESET KBD FLAG
	8662	5E		1411	LR D,A
				1412	* COMPARE TWO SCANS
	8663	4C		1413	LR A,S
	8664	E2		1414	XS 2
	8665	8431	8697	1415	BZ A46 BRANCH IF SCANS ARE THE SAME
	8667	900E	8676	1416	BR A45
				1417	* CHANGE KEY CODE FOR NEXT COLUMN
	8669	20FC		1418	A43 LI H'FC'
	866B	C2		1419	AS 2
	866C	52		1420	LR 2,A
				1421	* SHIFT STROBE ONE POSITION
	866D	6E		1422	A44 LISL 6
	866E	4C		1423	LR A,S
	866F	13		1424	SL 1
	8670	1F		1425	INC
	8671	5C		1426	LR S,A
				1427	* HAS LAST COLUMN BEEN SCANNED ALREADY
	8672	2140		1428	NI H'40'
	8674	94CD	8642	1429	BNZ A40 BRANCH IF NOT LAST STROBE
				1430	* STORE 'FF' IN KEY CODE REGISTER
	8676	20FF		1431	A45 LI H'FF'
	8678	52		1432	LR 2,A
				1433	* RESET KBD FLAG
	8679	6C		1434	LISL 4
	867A	70		1435	LIS 0
	867B	5C		1436	LR S,A
	867C	9004	8681	1437	BR A51
				1438	* GO TO NEXT ROW
				1439	* SET KBD FLAG
	867E	5E		1440	A48 LR D,A
				1441	* STORE KEY CODE OF FIRST SCAN
	867F	42		1442	LR A,2
	8680	5E		1443	LR D,A
				1444	* LOAD COUNTERS FOR BOUNCE TIME OF 28MS
	8681	298724	8724	1445	A51 JMP DISP
	8684	65		1446	A53 LISU 5
	8685	6A		1447	LISL 2
	8686	7B		1448	LIS 11
	8687	5C		1449	LR S,A
	8688	70		1450	A49 LIS 0
	8689	51		1451	LR 1,A
	868A	31		1452	A50 DS 1
	868B	94FE	868A	1453	BNZ A50 BRANCH IF COUNTER 1 NOT ZERO
	868D	3C		1454	DS 5
	868E	94F9	8688	1455	BNZ A49 BRANCH IF COUNTER 2 NOT ZERO
	8690	6C		1456	LISL 4
	8691	71		1457	LIS 1
	8692	FC		1458	NS 5
	8693	8403	8697	1459	BZ A46 BRANCH IF NOT KBD FLAG
	8695	90A4	863A	1460	BR A39
				1461	*STILL SAME KEY?
	8697	62		1462	A46 LISU 2
	8698	68		1463	LISL 0
	8699	4C		1464	LR A,S

## APPENDIX H (continued)

ERRS	LOC	OBJECT	ADDR	LINE	SOURCE STATEMENT		
	869A	E2		1465	X5	2	
	869B	849E	863A	1466	BZ	A39	NOT DIFFERENT
	869D	42		1467	LR	A, 2	
	869E	5C		1468	LR	S, A	
	869F	25FF		1469	CI	H'FF'	
	86A1	8498	863A	1470	BZ	A39	
	86A3	66		1471	LISU	6	
	86A4	6F		1472	LISL	7	
	86A5	14		1473	SR	4	
	86A6	841D	86C4	1474	BZ	A1A	MULTI FUNCTION KEY
	86A8	42		1475	LR	A, 2	
	86A9	2515		1476	CI	H'15'	KILL
	86AB	8440	86EC	1477	BZ	B1A	
	86AD	2514		1478	CI	H'14'	C/R
	86AF	8435	86E5	1479	BZ	C1A	
	86B1	2513		1480	CI	H'13'	MINUS
	86B3	8425	86D9	1481	BZ	E1A	
	86B5	2517		1482	CI	H'17'	PLUS
	86B7	8421	86D9	1483	BZ	E1A	
	86B9	2516		1484	CI	H'16'	CHANGE
	86BB	9418	86D4	1485	BNZ	F1A	
	86BD	72		1486	LIS	2	
	86BE	5E		1487	LR	D, A	CHANGE CODE=2
	86BF	2040		1488	LI	H'40'	
	86C1	5C		1489	LR	S, A	STORE MODE ON
	86C2	9016	86D9	1490	BR	E1A	
	86C4	6E		1491	A1A	LISL	6
	86C5	CC		1492	AS	5	
	86C6	812C	86F3	1493	BP	G1A	NOT FUNCTION
	86C8	71		1494	LIS	1	
	86C9	E2		1495	X5	2	
	86CA	847A	8745	1496	BZ	E70	
	86CC	72		1497	LIS	2	
	86CD	E2		1498	X5	2	
	86CE	8472	8741	1499	BZ	F2716	
	86D0	42		1500	LR	A, 2	
	86D1	2418		1501	AI	24	SHIFT IN TABLE LOOKUP
	86D3	52		1502	LR	2, A	
	86D4	6E		1503	F1A	LISL	6
	86D5	4C		1504	LR	A, S	
	86D6	13		1505	SL	1	
	86D7	12		1506	SR	1	
	86D8	5C		1507	LR	S, A	FUNCTION OFF
	86D9	65		1508	E1A	LISU	5
	86DA	6F		1509		LISL	7
	86DB	4C		1510	LR	A, S	
	86DC	0B		1511	LR	IS, A	ISAR RESTORED
	86DD	2A86FC	86FC	1512	DCI	KTAB	
	86E0	42		1513	LR	A, 2	
	86E1	8E		1514	ADC		
	86E2	16		1515	LM		
	86E3	51		1516	LR	1, A	ASC CODE FOR KEY IN
	86E4	1C		1517	POP		RETURN
				1518	*ENTRY FOR C/R		
	86E5	70		1519	C1A	LIS	0
	86E6	CC		1520	AS	S	
	86E7	8404	86EC	1521	BZ	B1A	CHANGE CODE=0
	86E9	3C		1522	DS	S	CHANGE CODE -1
	86EA	94EE	86D9	1523	BNZ	E1A	STILL EXPECT C/R
				1524	* ENTRY FOR KILL		
	86EC	70		1525	B1A	LIS	0

## APPENDIX H (continued)

ERRS	LOC	OBJECT	ADDR	LINE	SOURCE STATEMENT
	86E	5E		1526	LR D,A R67=0
	86EE	2000		1527	LI H'00'
	86F0	5C		1528	LR S,A FUNCTION NEXT
	86F1	90E7	86D9	1529	BR E1A
				1530	*NUMERIC KEY IN
	86F3	6F		1531	G1A LISL 7
	86F4	70		1532	LIS 0
	86F5	CC		1533	AS 5
	86F6	8402	86F9	1534	BZ G2A CHANGE CODE=0
	86F8	72		1535	LIS 2 CHANGE CODE =2
	86F9	5E		1536	G2A LR D,A
	86FA	90D9	86D4	1537	BR F1A
				1538	*TRANSLATE KEY PAD TO ASC CHARACTER
	86FC	30		1539	KTAB DC C'0'
	86FD	34		1540	DC C'4'
	86FE	38		1541	DC C'8'
	86FF	43		1542	DC C'C'
	8700	31		1543	DC C'1'
	8701	35		1544	DC C'5'
	8702	39		1545	DC C'9'
	8703	44		1546	DC C'D'
	8704	32		1547	DC C'2'
	8705	36		1548	DC C'6'
	8706	41		1549	DC C'A'
	8707	45		1550	DC C'E'
	8708	33		1551	DC C'3'
	8709	37		1552	DC C'7'
	870A	42		1553	DC C'B'
	870B	46		1554	DC C'F'
				1555	*SINGLE FUNCTION KEYS
	870C	4D		1556	DC C'M' MEMORY DISPLAY
	870D	52		1557	DC C'R' REGISTER
	870E	4F		1558	DC C'O' PORT
	870F	2D		1559	DC C-' DELIMITER MINUS
	8710	00		1560	DC H'00' CARRIAGE RETURN
	8711	5B		1561	DC H'5B' KILL
	8712	43		1562	DC C'C' CHANGE
	8713	2B		1563	DC C+' DELIMITER PLUS
				1564	*SHIFT FUNCTION KEYS
	8714	50		1565	DC C'P' PC
	8715	00		1566	DC 0 GO TO 2716
	8716	00		1567	DC 0 GO TO E70
	8717	55		1568	DC C'U' BREAKPOINT
	8718	44		1569	DC C'D' DC
	8719	49		1570	DC C'I' ISAR
	871A	58		1571	DC C'X' HEX
	871B	54		1572	DC C'T' CLEAR BREAK
	871C	42		1573	DC C'B' BACK OR PREVIOUS
	871D	57		1574	DC C'W' STATUS
	871E	56		1575	DC C'V' MOVE
	871F	4C		1576	DC C'L' LOAD
	8720	4E		1577	DC C'N' NEXT
	8721	41		1578	DC C'A' ACCUMULATOR
	8722	4B		1579	DC C'K' FIND
	8723	47		1580	DC C'G' GO TO
				1581	* DISPLAY SUBROUTINE
				1582	*
	8724	66		1583	DISP LISU 6
	8725	6E		1584	LISL 6
	8726	71		1585	LIS 1
	8727	51		1586	LR 1,A

## APPENDIX H (continued)

ERRS	LOC	OBJECT	ADDR	LINE	SOURCE STATEMENT
	8728	4E		1587	LR A,D
	8729	57		1588	LR 7,A
	872A	47		1589	A52 LR A,7
	872B	18		1590	COM
	872C	2720		1591	OUT P8 LED'S ON, SELECT DIGIT OFF
	872E	4E		1592	LR A,D
	872F	18		1593	COM
	8730	15		1594	SL 4
	8731	14		1595	SR 4
	8732	2721		1596	OUT P9
	8734	2620		1597	IN P8
	8736	E1		1598	XS 1
	8737	2720		1599	OUT P8
	8739	41		1600	LR A,1
	873A	13		1601	SL 1
	873B	51		1602	LR 1,A
	873C	8FED	872A	1603	BR7 A52
	873E	298684	8684	1604	JMP A53 RETURN
				1605	*
				1606	*
				1607	* PEP BOARD 2716 AND 38E70 PROGRAMMING
				1608	*
				1609	*
				1610	* USER MUST INITIALIZE PROGRAM BY STORING IN
				1611	* RAM H'2B08' THROUGH H'2BDE':
				1612	* MEMORY START ADDRESS HI BYTE
				1613	* MEMORY START ADDRESS LOW BYTE
				1614	* PROM START ADDRESS HI BYTE
				1615	* PROM START ADDRESS LOW BYTE
				1616	* PROM END ADDRESS HI BYTE
				1617	* PROM END ADDRESS LOW BYTE
				1618	* BLANK CHECK FLAG
				1619	* 0=>OMIT BLANK CHECK
				1620	* H'FF'=>PERFORM BLANK CHECK
				1621	*
				1622	*
				1623	* REGISTER USAGE
				1624	*
				1625	* R0=ERROR CODE
				1626	* 0=>GOOD
				1627	* 1=>BURN VERIFY ERROR
				1628	* 2=>BLANK CHECK ERROR
				1629	* R1=DATA OUTPUT
				1630	* R3=BLANK CHECK FLAG
				1631	* 0=>OMIT BLANK CHECK
				1632	* H'FF'=>PERFORM BLANK CHECK
				1633	* R4=PROM END ADDRESS HI BYTE
				1634	* R5=PROM END ADDRESS LO
				1635	* R7=MASK TO COMPLEMENT DATA OUT
				1636	* 0=>38E70
				1637	* H'FF'=>2716
				1638	* R6=ADDRESS SCRAMBLE (SUB ADDR)
				1639	* =DELAY LOOP COUNTER
				1640	* R9=DELAY LOOP COUNTER
				1641	* H=PROM START ADDRESS
				1642	* Q=MEMORY START ADDRESS
				1643	*
				1644	*
				1645	*
				1646	*
	8741	20FF		1647	F2716 LI H'FF'

## APPENDIX H (continued)

ERRS	LOC	OBJECT	ADDR	LINE		SOURCE	STATEMENT
	8743	9002	8746	1648		BR	F2111
	8745	70		1649	E70	LIS	0
	8746	2A0058	0058	1650	F2111	DCI	BRAM
	8749	57		1651		LR	7, A =>R7
	874A	16		1652		LM	MEMORY START ADDRESS
	874B	06		1653		LR	00, A =>Q
	874C	16		1654		LM	
	874D	07		1655		LR	0L, A
	874E	16		1656		LM	PROM START ADDRESS
	874F	5A		1657		LR	10, A =>H
	8750	16		1658		LM	
	8751	5B		1659		LR	11, A
	8752	16		1660		LM	PROM END ADDRESS
	8753	54		1661		LR	4, A =>4, 5
	8754	16		1662		LM	
	8755	55		1663		LR	5, A
				1664	*		
				1665	*		BLANK CHECK
				1666	*		
	8756	72		1667		LIS	2 SET ERROR FLAG
	8757	50		1668		LR	0, A
	8758	70		1669		CLR	OMIT BLANK CHECK IF FLAG SET
	8759	88		1670		AM	
	875A	53		1671		LR	3, A
	875B	10		1672		LR	DC, H
	875C	2C		1673		XDC	
	875D	10		1674		LR	DC, H
	875E	840B	876A	1675		BZ	CONTU
	8760	51		1676		LR	1, A
				1677	*		
	8761	2887B8	87B8	1678	AGN	PI	ADDR OUTPUT ADDRESS
	8764	2887A2	87A2	1679		PI	REED VERIFY BYTE IS BLANK
	8767	16		1680		LM	INCREMENT DATA COUNTER
	8768	94F8	8761	1681		BMZ	AGN LOOP IF NOT FINISHED
				1682	*		
				1683	*		BURN PROM
				1684	*		
	876A	30		1685	CONTU	DS	0 SET ERROR FLAG
	876B	0F		1686		LR	DC, Q MEMORY START ADDRESS
	876C	2C		1687		XDC	RESTORE PROM START ADDRESS
	876D	11		1688		LR	H, DC
				1689	*		
	876E	2887B8	87B8	1690	BURN	PI	ADDR OUTPUT ADDRESS
	8771	2C		1691		XDC	SWITCH TO MEM DC
	8772	16		1692		LM	GET BYTE OF DATA
	8773	2C		1693		XDC	SWITCH BACK TO PROM DC
	8774	51		1694		LR	1, A SAVE COPY OF DATA BYTE
	8775	E7		1695		XS	7 CPLEMENT IF 2716
	8776	B5		1696		OUTS	5 OUTPUT DATA
	8777	A0		1697		INS	0 TURN ON 2716 WRITE
	8778	13		1698		SL	1
	8779	12		1699		SR	1
	877A	B0		1700		OUTS	0
	877B	21F7		1701		NI	H'F7' TURN ON 2716 BURN
	877D	B0		1702		OUTS	0
	877E	A1		1703		INS	1 TURN ON 38E70 BURN
	877F	2440		1704		AI	H'40'
	8781	B1		1705		OUTS	1
				1706	*		
	8782	7B		1707		LIS	11 BEGIN DELAY
	8783	59		1708		LR	9, A



## APPENDIX H (continued)

ERRS	LOC	OBJECT	ADDR	LINE	SOURCE	STATEMENT
	8784	70		1709	LIS	0
	8785	56		1710	LR	6, A
	8786	A5		1711	WAIT	INS 5
	8787	36		1712	DS	6
	8788	94FD	8786	1713	BNZ	WAIT
	878A	39		1714	DS	9
	878B	94FA	8786	1715	BNZ	WAIT     END DELAY
	878D	A1		1716	INS	1     TURN OFF 38E70 BURN
	878E	21BF		1717	NI	H'BF'
	8790	B1		1718	OUTS	1
	8791	A0		1719	INS	0     TURN OFF 2716 BURN
	8792	2408		1720	AI	8
	8794	B0		1721	OUTS	0
	8795	2480		1722	AI	H'80'     TURN OFF 2716 WRITE
	8797	B0		1723	OUTS	0
	8798	16		1724	LM	INCREMENT FROM DC
	8799	2887A2	87A2	1725	PI	REED     VERIFY DATA
	879C	94D1	876E	1726	BNZ	BURN
	879E	30		1727	DS	0     SUCCESSFUL BURN
	879F	288080	8080	1728	ERR	PI     SAVER     RETURN TO MONITOR
				1729	*	
				1730	*****	
				1731	*	
				1732	*	ROUTINE TO READ DATA AND COMPARE W/ OUTPUT
				1733	*	
				1734	*	R1=ORIGINAL DATA
				1735	*	R7=DEVICE FLAG
				1736	*	
	87A2	70		1737	REED	CLR     CLEAR PORTS
	87A3	B4		1738	OUTS	4
	87A4	B5		1739	OUTS	5
	87A5	C7		1740	AS	7     IDENTIFY DEVICE
	87A6	9404	87AB	1741	BNZ	VER1
	87A8	A4		1742	INS	4     READ PORT 4 FOR 38E70
	87A9	9002	87AC	1743	BR	VER2
	87AB	A5		1744	VER1	INS 5     READ PORT 5 FOR 2716
	87AC	C1		1745	VER2	AS 1     COMPARE W/ OUTPUT
	87AD	1F		1746	INC	
	87AE	94F0	879F	1747	BNZ	ERR
				1748	*	
	87B0	45		1749	LR	A, 5     TEST FOR FINISH
	87B1	EB		1750	XS	11
	87B2	9403	87B6	1751	BNZ	RET
	87B4	44		1752	LR	A, 4
	87B5	EA		1753	XS	10
	87B6	11		1754	RET	LR     H, DC     UPDATE H
	87B7	1C		1755	POP	
				1756	*	
				1757	*****	
				1758	*	
				1759	*	ROUTINE TO TRANSLATE ADDRESS BITS AND OUTPUT THEM
				1760	*	
				1761	*	ADDRESS IN H MUST BE OUTPUT TO PORTS 0 AND 1
				1762	*	P0-0     ADDR BIT     6
				1763	*	P0-1     5
				1764	*	P0-4     2
				1765	*	P0-5     3
				1766	*	P0-6     4
				1767	*	P1-0     7
				1768	*	P1-1     8
				1769	*	P1-2     9

## APPENDIX H (continued)

ERRS	LOC	OBJECT	ADDR	LINE	SOURCE STATEMENT
				1770 *	P1-3 10
				1771 *	P1-4 1
				1772 *	P1-5 0
				1773 *	
				1774 *	P0-3 2716 PROGRAM STROBE--0=>BURN
				1775 *	P0-7 2716 READ/WRITE--0=>WRITE
				1776 *	P1-6 38E70 PROGRAM STROBE--1=>BURN
				1777 *	
				1778 *	PORT 4--DATA FROM 38E70
				1779 *	PORT 5--DATA TO 38E70
				1780 *	DATA TO AND FROM 2716
				1781 *	
				1782 *	TRUE DATA--OUTPUT TO 38E70
				1783 *	FALSE DATA--OUTPUT TO 2716
				1784 *	ALL ADDRESSES
				1785 *	ALL INPUT
				1786 *	
				1787 *	BURN AND R/W BITS ARE FORCED TO NOBURN AND
				1788 *	TO READ; THEY WILL BE TOGGLED WITHIN MAIN
				1789 *	PROGRAM AT APPROPRIATE TIMES.
				1790 *	
87B8	77		1791	ADDR LIS 7	
87B9	FA		1792	NS H11	
87BA	13		1793	SL 1	ADDR BITS 8-10
87BB	1F		1794	INC	SET BIT 7
87BC	56		1795	LR 6,A	SAVE
87BD	48		1796	LR A,L0	
87BE	FB		1797	NS L0	
87BF	9102	87C2	1798	BM **3	BIT 7 SET?
87C1	36		1799	DS 6	NO BIT 7
87C2	73		1800	LIS 3	
87C3	FB		1801	NS L0	
87C4	8406	87CB	1802	BZ OK	ADDR BITS 0-1 ARE 0
87C6	2303		1803	XI 3	INVERT
87C8	9402	87CB	1804	BNZ OK	WAS 10 OR 01
87CA	73		1805	LIS 3	BOTH 1
87CB	15		1806	SL 4	
87CC	C6		1807	AS 6	
87CD	2440		1808	AI H'40'	FORCE E70 PGM STROBE OFF
87CF	18		1809	COM	
87D0	B1		1810	OUTS 1	BITS 7-10 AND 0-1
			1811 *		
			1812 *	NOW DO PORT 0	
			1813 *		
87D1	48		1814	LR A,L0	
87D2	13		1815	SL 1	
87D3	1E		1816	LR J,W	SAVE BIT 6
87D4	13		1817	SL 1	
87D5	8103	87D9	1818	BP **4	BIT 5 WAS 0
87D7	1F		1819	INC	
87D8	1F		1820	INC	
87D9	1D		1821	LR W,J	
87DA	8102	87DD	1822	BP **3	BIT 6 WAS 0
87DC	1F		1823	INC	SET BIT 0 TO 1
87DD	2173		1824	NI H'73'	FORCE BITS 3,7 TO 1
87DF	18		1825	COM	(2716 READ MODE)
87E0	B0		1826	OUTS 0	ADDR BITS 2-6
87E1	1C		1827	POP	
			1828 *		
			1829 *		
			1830 *		

## APPENDIX H (continued)

ERRS	LOC	OBJECT	ADDR	LINE	SOURCE STATEMENT
				1831	*
				1832	*
				1833	*
				1834	*
				1835	*SUBROUTINE TO DISPLAY WITHOUT KEYBOARD READ
				1836	*
87E2	66			1837	DISPX LISU 6
87E3	6E			1838	LISL 6
87E4	71			1839	LIS 1
87E5	51			1840	LR 1, A
87E6	4E			1841	LR A, D
87E7	57			1842	LR 7, A
87E8	47			1843	A52X LR A, 7
87E9	18			1844	COM
87EA	2720			1845	OUT P8
87EC	4E			1846	LR A, D
87ED	18			1847	COM
87EE	15			1848	SL 4
87EF	14			1849	SR 4
87F0	2721			1850	OUT P9
87F2	2620			1851	IN P8
87F4	E1			1852	XS 1
87F5	2720			1853	OUT P8
87F7	41			1854	LR A, 1
87F8	13			1855	SL 1
87F9	51			1856	LR 1, A
87FA	8FED	87E8		1857	BR? A52X
87FC	1C			1858	POP
				1859	*
				1860	*
				1861	*
				1862	END

00 ERRS

## APPENDIX H (continued)

SYMBOL	TYP	VAL	REFERENCES
A	L	82CB	0775 0537 0535 0377 0376 0375 0374 0373
			0372 0371 0369 0368 0367 0366 0365 0364 0363
			0362 0361 0360 0359 0358 0357 0356 0355 0354
A1	L	82F6 0354	
A1A	L	86C4 1474	
A39	L	863A 1470 1466 1460	
A40	L	8642 1429	
A41	L	8654 1404	
A42	L	865D 1398	
A43	L	8669 1396	
A44	L	866D 1402	
A45	L	8676 1416	
A46	L	8697 1459 1415	
A48	L	867E 1409	
A49	L	8688 1455	
A50	L	868A 1453	
A51	L	8681 1437	
A52	L	872A 1603	
A52X	L	87E8 1857	
A53	L	8684 1604	
ADDR	L	87B8 1690 1678	
ADON	L	859F 1252	
AEND	L	8595 1244	
AGAN	L	857E 1249	
AGN	L	8761 1681	
A1A	L	86EC 1521 1477	
BAUD	A	0000 1354 1311 1296	
BAUDT	L	80DD 0229	
BCNT	A	0002 1360 1350 1306 1303	
BLNK	L	8436 0944	
BLP	L	82A2 0666	
BOT1	L	84C6 1100	
BRBK	L	8295 0743	
BRAM	L	0058 1650	
BURN	L	876E 1726	
BXT	L	82F4 0746	
BXX	L	8292 0736 0696 0678	
B	L	837D 0355	
BYTE	L	857B 1182 1178 1175 1172 1170 1166 1151 1149 1129	
C0	L	830A 0750	
C1	L	8312 0754	
C1A	L	86E5 1479	
C2	L	8318 0753	
C3	L	831F 0761	
C5	L	8324 0780	
C55	L	8330 1226	
CCNT	A	0005 1180 1168 1167 1131 1127 0021	
CC	A	000B 0505 0492 0490 0476 0449 0409	
CHAR	L	85A2 1235 1133 1113	
CHR1	A	000B 1256 1254 1245 1241 1240 1234 1232 1176	
CHRS	A	0001 1369 1366 1364 1334 1331 1310 1305 1286 1278 1160 1146 1143 1097 1080 1036 1027 1013 1003 1000 0975 0942 0923 0919 0877 0773 0382 0343 0128 0085 0082 0076 0067	
CKSM	A	0007 1255 1253 1247 1246 1184 1183 1163 1138 1125 1112 0350 0349 0108 0069 0066 0017	
CONT	L	84E3 1132	
CONTU	L	876A 1675	
COR1	L	8202 0494	
COR2	L	8209 0506	

## APPENDIX H (continued)

SYMBOL	TYP	VAL	REFERENCES
COR3	L	820B	0509
COR4	L	820D	0504
COR5	L	820F	0493
COR6	L	8210	0491
COR7	L	821D	0481
COR8	L	8219	0519
CORN	L	81ED	0433 0431
CORT	L	81EE	0435 0427
CR	A	000D	1330 0518 0426
C	L	82FE	0775 0356
D0	L	0044	0255
D	L	8334	0357
DISP	L	8724	1445
DISPX	L	87E2	
DLY1	L	861C	1370
DLY2	L	861D	1359
DLY3	L	85CC	1299
DLY4	L	85E3	1316
DLY5	L	862B	1361
E1A	L	86D9	1529 1523 1490 1483 1481
E70	L	8745	1496
EA	A	0005	1092
EAHI	A	0005	0121 0064 0062 0057 0055
EALO	A	0006	0118 0061 0058
EGHT	L	8498	1083
ENDX	L	8512	1121
ERR	L	879F	1747
EX	L	004A	0316
E	L	833A	0358
F1	L	836F	0044 0042
F1A	L	86D4	1537 1485
F2111	L	8746	1648
F2716	L	8741	1499
FA	L	818B	0482
FRA	L	8192	0418
FRX	L	81EB	0521
FB	L	8197	0479 0477 0450 0447 0442 0440
FCNT	A	0008	0908 0795 0745 0532 0525 0522 0516 0406
FCUN	L	8024	0122 0073
FF	L	8236	0538
FHI	L	8134	0335
FINI	L	813D	0341
FLG	A	0009	0950 0921 0907 0874 0815 0789 0747
FLOP	L	804A	0106
FLP1	L	812D	0346
FMLP	L	8535	1181
FOK	L	81C9	0445
FOP1	L	812A	0966 0888 0879 0103
FOP2	L	812C	1068 0940 0933 0735 0728 0132 0114 0099
			0096 0093 0090
FORM	L	851B	1117
FPN1	L	8010	0071
FPUN	L	8003	0793 0136
F	L	8340	0537 0359
G	L	8343	0360
G1	L	834F	0796
G1A	L	86F3	1493
G2A	L	86F9	1534
HERE	L	806D	0134
HEX	L	82CC	0562 0007
HFLG	A	0008	1248 1231 1086 1032 1016 0345 0333 0331

## APPENDIX H (continued)

SYMBOL	TYP	VAL	REFERENCES
			0135 0133 0131 0113 0105 0102 0088 0072
HIGH	L	84B8	0822
HII	A	000A	1792
HXX	L	015F	
H	L	0361	0361
I1	L	0354	0904 0739
IDLE	L	84CA	1186 1177 1156 1154 1140 1123
INS	L	0492	1045
IPOR	A	0028	1345 1325 1317 1300 1294 1290 0223 0222
I	L	0352	0362
J	L	035E	0363
JMP	L	055C	1224
K	L	036D	0364
KTAB	L	06FC	1512
LF	A	000A	1333
LO	A	000B	1814 1801 1797 1796
L	L	0367	0365
LOAD	L	04AD	0827
LOOP	L	05D8	1318
LOP2	L	05E1	1308
MBGN	L	0252	0592
MBY2	L	025A	0579
MCOM	L	027F	0587
MEND	L	0260	0580
MLOP	L	0282	0643 0640
MM	A	000A	0829
MOVE	L	024A	0536
MXX	L	026B	0602
M	L	036A	0366
N1	L	0386	0845 0838 0791
N2	L	0384	0851
NOSP	L	05B4	1283
NUSE	A	00EB	1357 1356 1355 1314 1313 1312
N	L	0382	0367
OK	L	07CB	1804 1802
OO	A	000C	0951 0824 0749
OPOR	A	0028	1365 1352 0160
O	L	0364	0368
OUTS	L	056E	1194
P1	L	0395	0869 0786
P2	L	039A	0790
P3	L	03BB	0985 0909
P8	A	0020	1853 1851 1845 1599 1597 1591 1390 1388
P9	A	0021	1850 1596 1394 1386
PBLK	L	0455	1093
POLY	L	05AA	1275
PINP	L	05A3	1271 1101
P10	L	046C	0952
PIX	L	0609	1343 1341
PLP	L	0458	1030
PLUS	L	02DB	0701
PLXX	L	0466	1161
PORTA	A	0020	0313 0305 0190 0158
PORTB	A	0020	0311 0307 0213 0162
PORTC	A	0029	0215 0164
PORT	L	0546	0751
PP	A	000D	0673 0788 0752
PPA	L	044C	1094
PPC	L	0496	1085 1033
PPRT	A	0029	1276 1103
PPUN	L	0444	0818

## APPENDIX H (continued)

SYMBOL	TYP	VAL	REFERENCES
PSTS	A	0028	1285 1281 1280 1269 1105 1104
P	L	838E	0369
PXR	L	848B	1055
PXX	L	847A	1072
PXXY	L	848D	1065
R1	L	83C4	0830 0825
R2	L	83CE	0995 0910
R22	L	83D1	0864
R23	L	83F2	0816
R4	L	83FA	1005 0998
R5	L	840D	0954
R77	L	8411	1069
R7	L	8428	0983
RA9	L	0009	0303 0210
RAM	L	0000	0269 0232 0167 0045 0040 0036 0035
REED	L	87A2	1725 1679
REST	L	80E3	0806
RETX	L	8579	1208
RET	L	87B6	1751
RPTC	L	8184	0777
RR	A	000F	0953 0906 0814
RTNL	L	823B	0540 0533
R	L	83C2	0371
SA	A	0003	1091 1088 1019
SAVER	L	8080	1728 0247 0048
SCAN	L	8636	1293
SETR	L	8507	1119
SIZE	A	0007	1090 1022 1018 1017
SLF1	L	84BC	1108
SLF2	L	84F9	1185
STB	L	8248	0551
STOP	L	85F1	1307
STPP	L	8515	1169 1148
STR1	L	8162	0437 0400 0393 0391
STRT	L	8160	0901 0644 0146
S	L	838E	0376 0372
TBLS	L	8148	0395
TCRX	L	85F9	1327
TT	L	82B8	0741
TTCR	L	85F2	0917 0380 0115
TTLP	L	82C3	0695
TTX	L	85C7	1295 1292
TTYI	L	858E	1301 1099 0423 0385
TTYO	L	8610	1335 1332 1147 1144 1098 1081 1037 1028 1014 1004 1001 0976 0943 0924 0920 0878 0774 0428 0387 0383 0344 0129 0086 0083 0077 0068
TTYX	L	8617	1347
TZE	L	8076	1038 0235 0226 0078
T	L	82FA	0373
U	L	82FC	0374
V	L	8361	0535 0375
VER1	L	87AB	1741
VER2	L	87AC	1743
WRIT	L	8786	1715 1713
XFLG	A	0006	1251 1250 1243 1242 1165 1155 1128 1110 1082 1077 1029 1025 0020
XX	L	8000	
X	L	82CC	0377
ZR	L	80E8	0260
ZB	L	80F5	0276
ZC	L	809B	0188
ZD	L	80B2	0207
ZE	L	80DB	













**FAIRCHILD**

Manufactured under one of the following U.S. Patents: 2981877, 3015048, 3064167, 3108359, 3117260, other patents pending.  
Fairchild cannot assume responsibility for use of any circuitry described other than circuitry embodied in a Fairchild product.  
Fairchild reserves the right to make changes in the circuitry or specifications at any time without notice.

Printed in U.S.A. /332-12-0006-049/5M/May, 1979