

SOFTWARE REFERENCE MANUAL

H8 COMPUTER

FRONT PANEL MONITOR PAM-8

Copyright © 1979
Heath Company
All Rights Reserved

HEATH COMPANY
BENTON HARBOR, MICHIGAN 49022

595-2348

Printed in the United
States of America

TABLE OF CONTENTS

INTRODUCTION	1-4
THEORY OF OPERATION	
Power Up and Master Clear	1-5
Clock Interrupts	1-5
PAM-8 Modes/Using RST and RTM	1-6
H8 Displays	1-7
H8 Keypad	1-9
DISPLAYING AND ALTERING MEMORY LOCATIONS	
Specifying a Memory Address	1-10
Altering a Memory Location	1-12
Stepping Through Memory	1-13
DISPLAYING AND ALTERING REGISTERS	
Specifying a Register for Display	1-14
Altering the Contents of a Selected Register	1-15
Stepping Through the Registers	1-15
PROGRAM EXECUTION CONTROL	
Initiating Program Execution	1-16
Breakpointing	1-16
Single Instruction Operation	1-17
Interrupting a Program During Execution	1-17

LOAD/DUMP ROUTINES	1-18
Loading From Tape	1-18
Dumping to Tape	1-18
Copying a Tape	1-20
Tape Errors	1-20
I/O FACILITIES	
Inputting From a Port	1-21
Outputting to a Port	1-21
Addressing Port Pairs	1-21
ADVANCED CONTROL	
16-Bit Tick Counter (TICCNT)	1-22
Using the Keypad	1-22
Display Usage	1-23
Using Interrupts	1-24
APPENDIX A	
Panel Monitor Listing	1-25
APPENDIX B	
DEMO: PAM-8	1-64
INDEX	1-69

INTRODUCTION

This Manual describes the functions and operations of the Heath H8 Panel Monitor Program, PAM-8, which resides permanently in a ROM on the H8 CPU board. PAM-8 provides a sophisticated front panel display and keyboard emulation as well as handling master clear and interrupt operations. Some of the major features of PAM-8 are:

- Memory contents display and alteration.
- Register contents display and alteration.
- Program execution control (both breakpoint and single instruction operation).
- Self-contained bootstraps for program loading and dumping.
- Port input and output routines.

In addition to the above features, PAM-8 can be instructed (by means of a flag byte contained in H8 RAM) to bypass some or all of its normal functions so the sophisticated user can augment or totally replace them.

Communication with the Panel Monitor is accomplished through three devices: the keypad, the 7-segment displays, and the audio alert. The user enters commands and values through the 16-key keypad, and PAM-8 responds visually through the front panel displays. In addition to the front panel displays, PAM-8 provides the keypad entry and function feedback to the built-in speaker. Appropriate signals (short, medium, and long beeps) indicate that commands and data are accepted or rejected.

THEORY OF OPERATION

This section will supplement the information contained in the "Operation" and "Circuit Description" sections of your H8 Operation Manual. In order to fully understand how PAM-8 operates, you must be familiar with the H8 front panel and CPU. A thorough knowledge of the 8080 instruction set and its architecture is also essential.

Power Up and Master Clear

PAM-8 initializes the H8 whenever you power-up or master clear (RST). You initiate the power-up operation by turning on the rear panel Power switch. You can master clear by simultaneously depressing both the lower right-hand (RST \emptyset) and lower left-hand (\emptyset) keys of the H8 front panel keypad. Both power-up and RST cause a level zero (highest priority) interrupt and result in a long beep from the audio alert.

During initialization, PAM-8 enters a routine which determines the high limit of continuous RAM. Once the high limit of available RAM is determined, the H8 stack pointer (SP) is set to this value and control is passed to the front panel command loop. Using this feature, you can immediately determine the total amount of continuous memory above 8K by displaying stack pointer value.

Clock Interrupts

The Clock Interrupt is a crucial element in the operation of the H8 front panel system. This level one interrupt is generated by the front panel hardware every 2,000 μ S. PAM-8 uses this interrupt to check for some keyboard commands, to check for user program breakpoints, and to refresh the front panel displays.

PAM-8 performs these functions using a series of subroutines which are executed as necessary when indicated by the interrupts. For this reason, all user programs must maintain a valid stack (at high memory) containing at least 80 free bytes at all times. If this stack space is not available and PAM-8 is running (it can be disabled; see the Advanced Control Section), unpredictable software damage can occur in your program. In the same manner, if your program should execute a DI (Disable Interrupt) instruction, no front panel services including the RTM (Return To Monitor) function are available until an EI (Enable Interrupt) instruction is executed or until a master clear (RST/ \emptyset) is performed.

PAM-8 Modes/Using RST and RTM

PAM-8 is always in either the monitor mode or the user mode. In the monitor mode no user program is executing, PAM-8 loops reading the keypad and refreshing the displays. All commands entered via the keypad are valid; however, the RTM command is meaningless.

When your program is being executed, PAM-8 is in the user mode and the MON LED on the front panel is extinguished. Only two keyboard commands are valid in this mode: RST (master clear) and RTM (Return To Monitor). NOTE: Both of these commands are dual key commands. No single key command is recognized, so a user program may have free use of the entire keypad.

You can return PAM-8 to the monitor mode by using the RTM command (simultaneously press the \emptyset and the # keys). This command stops program execution at the end of the current instruction, stores the current value of each register, and returns PAM-8 to the monitor mode. You can then continue your program by pressing the GO key. The RST command (simultaneously press the 0 and the / keys) performs the master clear operation described earlier and does not save any register values.

Normally, when a user program is running, PAM-8 is also running. Thus, if PAM-8 is displaying the contents of the HL register pair and the user program is started, it continues to display the contents of this register pair as the program is run. If the user program changes the contents of the HL pair, the change is immediately reflected in the front panel displays. In a similar manner, if a memory location is displayed when a user program is started, it is displayed during the time the user program is run. If the user program changes the contents of the displayed memory location, the front panel display changes.

Since PAM-8 does not recognize keypad commands in the user mode, the RTM command must be used before the memory location or register being displayed is changed to a new location or a different register. Once you select the new location or different register, you can resume program execution by pressing GO.

NOTE: PAM-8 requires about 10% of the H8 CPU's resources to process the display interrupts. Programs which are compute-bound may be slowed down by simultaneous operation of PAM-8. In this situation, you may wish to turn off the clock interrupts to improve execution time. See "Using Interrupts" on Page 1-24.

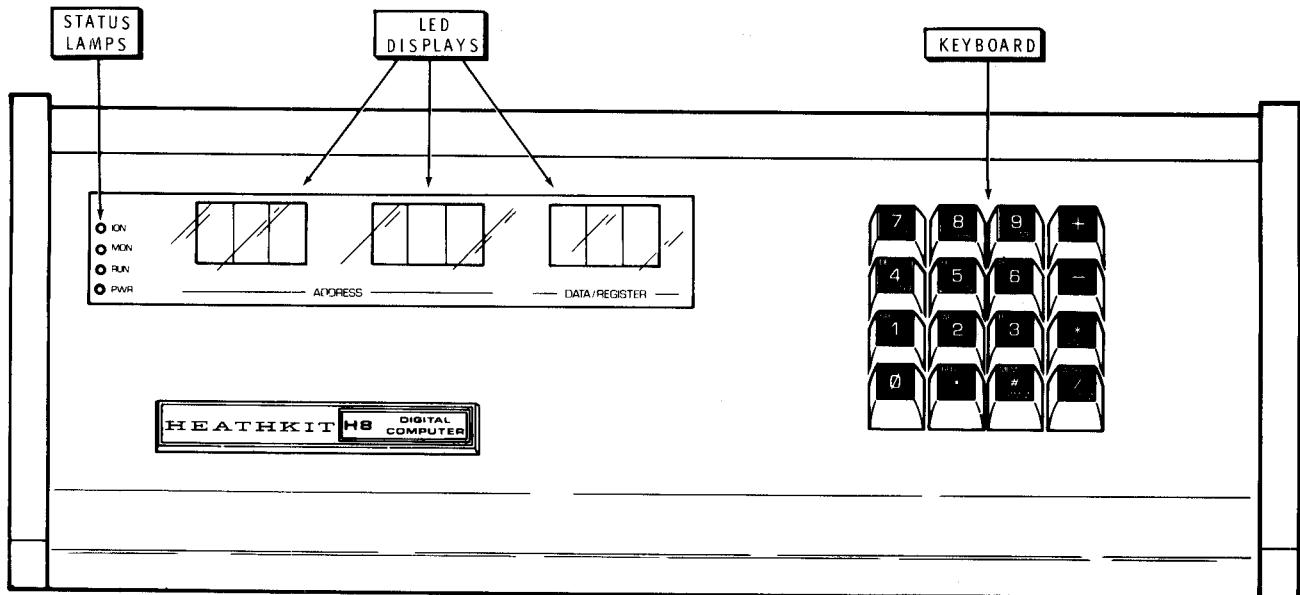


Figure 1-1

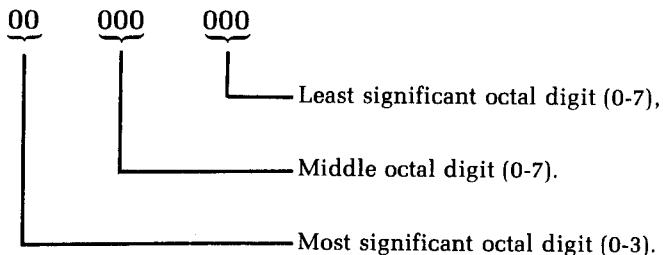
H8 Displays

You must understand the H8 front panel presentation in order to use PAM-8. The display is made up of 9 digits, in three groups of three digits each. See Figure 1-1. Each group of three digits displays one byte (eight bits) of information. This information may be the contents of a designated register or memory location, or it may be the address of a memory location itself. The register names are also displayed.

All binary numbers are converted to octal format for display on the H8 front panel. The following table shows binary to octal conversion.

<u>BINARY NUMBER</u>	<u>OCTAL NUMBER</u>
000	0
001	1
010	2
011	3
100	4
101	5
110	6
111	7

Each byte is displayed as two-and-one-half octal digits. The octal numbers lie in the range of 000 to 377 for binary numbers in the range 00000000 to 11111111, as shown below.



NOTE: As there are only eight bits in a byte, the most significant octal digit only represents two bits and is therefore displayed as 0 to 3. If the user should inadvertently enter the octal digits 4 to 7 into the most significant digit, the most significant bit is lost. Losing this bit converts 4 through 7 into the digits 0 through 3 respectively.

Also note that 16-bit numbers, such as memory addresses and certain register contents, are still displayed as two eight-bit numbers. Therefore, the H8 front panel representation of the number is made up of **two** groups of three octal numbers in the range of 000 to 377. This representation of 16-bit binary numbers is known as **offset octal**, and is used consistently throughout all H8 displays of 16-bit numbers. Offset octal must not be confused with octal. For example:

$\begin{array}{ccccccccc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ | & | & | & | & | & | & | & | \\ 3 & 7 & 7 & 3 & 7 & 7 & 7 & 7 \end{array}$
A 16-bit binary number
Offset octal representation (377 377)

$\begin{array}{ccccccccc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ | & | & | & | & | & | & | & | & | \\ 1 & 7 & 7 & 7 & 7 & 7 & 7 & 7 & 7 \end{array}$
A 16-bit binary number
True Octal representation (177777)

The lower example shows true octal representation of a 16-bit binary number. This is **not** used by the H8 front panel displays or any H8 software. Occasionally you will see offset octal numbers printed with a decimal point separating the upper and lower bytes. For example:

377.377

Hi Byte Lo Byte



H8 Keypad

The H8 Keypad consists of 16 keys, as shown in Figure 1-1. When the keypad is operating under the control of PAM-8, it exhibits a number of unique properties.

- Each keystroke is verified by a short beep from the audio alert.
- Octal digits are entered using the keys 0 through 7.
- Holding a key down continuously repeats the key's function.
- The + key increments memory port or register locations.
- The - key decrements memory port or register locations.
- The * key cancels previous keypad entries.
- The ALTER key causes PAM-8 to enter the alter mode.
- The MEM key causes PAM-8 to enter the display memory mode.
- The REG key causes PAM-8 to enter the register mode.

Many of the keys on the keypad have multiple functions, depending on the PAM-8 mode being used. In the register mode, for example, the numeric keys (1-6) call the register indicated in the upper left-hand corner of the key. When the PAM-8 is in neither the register nor the memory mode, the keys perform the functions indicated in the lower right-hand corner of the key.



The # and / keys have additional special functions, as indicated earlier. When the / key is pressed simultaneously with the 0 key, the RST (master clear) sequence is initiated. When the # sign key is depressed simultaneously with the 0 key, the RTM (Return To Monitor) function is initiated, the user program is stopped, and PAM-8 regains control.

Each key is covered in greater detail as the various function are discussed.



DISPLAYING AND ALTERING MEMORY LOCATIONS

One of the major features of PAM-8 is its ability to examine the contents of any H8 memory location and to modify the contents of that memory location if it is RAM.

When the H8 is first powered up, PAM-8 is in the display memory mode. This mode is indicated by all digits displaying octal numbers and no decimal points being on.

Specifying a Memory Address

If you wish to display or alter the contents of a memory location. You must first place PAM-8 in the memory address mode and then enter the desired memory address. Place PAM-8 in the memory address mode (if not already there) by pressing the MEM (Memory) key. Specify the address to be displayed or altered by entering the 6-digit address (offset octal).

When you press the MEM key, all the decimal points will light. This indicates that the address may now be entered. Once the full 6-digit address is entered, the decimal points turn off, indicating that address entry is completed. After all 6 digits are entered, the address is displayed in the left-most six displays, and the contents of the addressed memory location are displayed in the right-hand 3 digits.

NOTE: As you press each key, including the MEM key, a short beep indicates successful entry. As each group of three octal digits is successfully entered, a medium beep is sounded. The sequence by which you specify a memory address is shown in Figure 1-2.

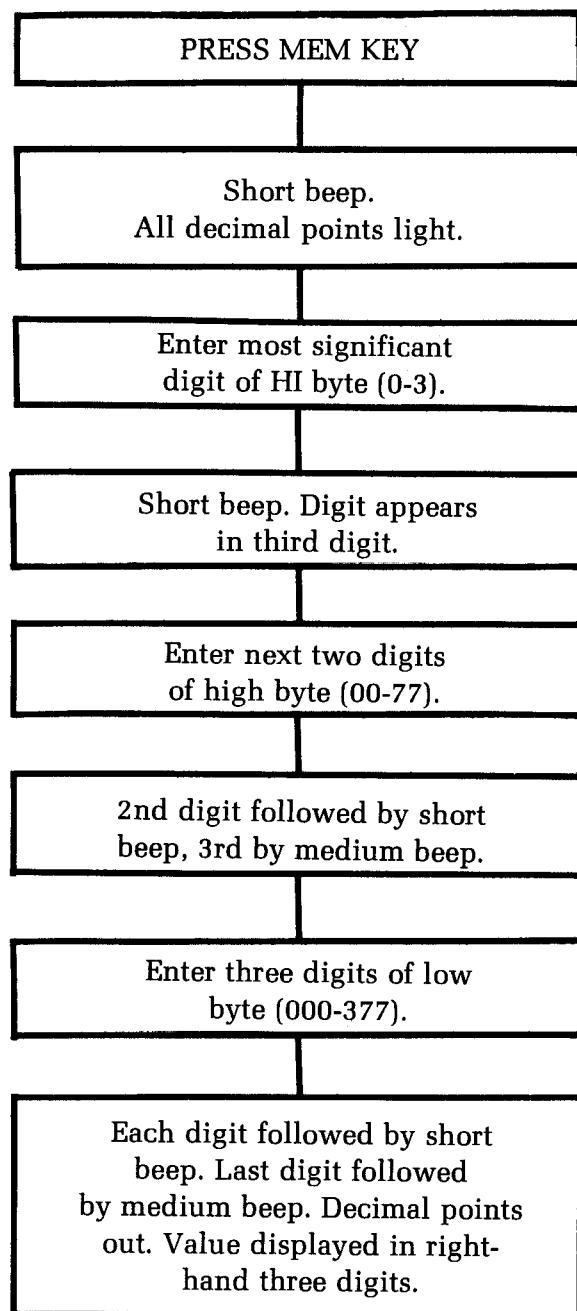


Figure 1-2
Entering a memory address through PAM-8.

NOTE: If you press a non-octal digit key as one of the six address digits, an error is flagged (a long beep). Once this error is flagged, the PAM-8 considers the address complete and extinguishes the decimal points. The entire sequence must be repeated.

Altering a Memory Location

Before you can alter a memory location, you must first display the contents of the memory location by specifying the memory address as described in the preceding paragraphs. After you specify the memory address, press the ALTER key. This will cause PAM-8 to enter the memory alter mode.

When PAM-8 enters the memory alter mode, a single decimal point rotates from right to left through all 9 digits. You can now alter the contents of the displayed location by entering the new octal value (three digits on the keypad). When the three digits have been entered, acoustical verification (a short beep) is given **and the memory address is incremented**. You can then alter this new location by entering three more digits or pressing one of the following keys, causing the monitor to perform the indicated function:

<u>KEY</u>	<u>FUNCTION</u>
+	Increment the address.
-	Decrement the address.
MEM	Specify a new memory address (leave memory alter mode).
REG	Specify a register for display (leave memory alter mode).
ALTER	Exit from the alter mode (into the display mode).

NOTE: PAM-8 automatically increments the memory address as each entry (3 octal digits) is complete. Therefore, you may load a program in sequential locations very rapidly. Each location is modified by simply entering the three octal digits.

The following example reviews each step as the H8 is turned on; the memory address mode is entered; and the location 040 123 is addressed, altered to 345, checked, and closed.

<u>DISPLAY</u>	<u>COMMENTS</u>
X X X X X X X X X	Random memory display at power up (X=random number.)
X.X.X. X.X.X. X.X.X.	MEM key pressed. (In memory address mode, a short beep.)
X.X.0. X.X.X. X.X.X.	0 key pressed. (Short beep.)
X.0.4. X.X.X. X.X.X.	4 key pressed. (Short beep.)
0.4.0. X.X.X. X.X.X.	0 key pressed. (Medium beep.) Contents of location 040 XXX displayed.)
0.4.0. X.X.1. X.X.X.	1 key pressed. (Short beep. Contents of 040 XX1 displayed.)
0.4.0. X.1.2. X.X.X.	2 key pressed. (Short beep. Contents of 040 X12 displayed.)
0 4 0 1 2 3 X X X	3 key pressed. (Medium beep. Contents of desired location 040 123 displayed, decimal points out.)
0.4.0 1.2.3 X.X.X	ALTER key pressed. (Short beep. Decimal points rotate.)
0.4.0. 1.2.3. X.X.3.	3 key pressed. (Short beep. Decimal points rotate.)
0.4.0. 1.2.3. X.3.4.	4 key pressed. (Short beep. Decimal points rotate.)
0.4.0. 1.2.4. X.X.X.	5 key pressed. (Medium beep. Address increments one location. Decimal points rotate.)
0.4.0 1.2.3 3.4.5	-key pressed. (Short beep. Address decrements one location. Decimal points rotate.)
0 4 0 1 2 3 3 4 5	ALTER key pressed. (Short beep. Decimal points go out.)

Stepping Through Memory

When PAM-8 is either in the display memory or alter memory modes, the + and - keys increment and decrement the memory address. Each time you press the key, PAM-8 increments (or decrements) the memory address one location. If you hold the key down, the auto-repeat function of PAM-8 causes the memory address to increment or decrement repeatedly (approximately one location every second).

DISPLAYING AND ALTERING REGISTERS

PAM-8 can display and alter the contents of the 8080 CPU registers, just as it displays and alters the contents of H8 memory locations. Although the process is quite similar, a few special features should be noted.

Specifying a Register for Display

Press the REG key to specify that a register is to be displayed. After you press the REG key, press a second key (SP through PC, see the Table below) to specify the desired register or register pair.

When the REG key is pressed, six decimal points light, indicating that you must now select a register. NOTE: Simply pressing the REG key causes a register name to appear in the right-hand digits. However, you must select a register using the Register Select key before a register is definitely selected and its true contents are displayed. Once a register is selected, the decimal points are extinguished.

The contents of the selected register pair are displayed in the six left-most displays. The register name (or names) are displayed in the two right-most digits of the right-hand three displays. The registers are selected and displayed in accordance with the following table:

<u>KEY</u>	<u>LEFT 3 DIGITS</u>	<u>MIDDLE 3 DIGITS</u>	<u>RIGHT PAIR</u>	<u>COMMENTS</u>
SP (1)	000 to 377	000 to 377	SP	Stack pointer
AF (2)	000 to 377	000 to 377	AF	AF Register pair
BC (3)	000 to 377	000 to 377	BC	BC Register pair
DE (4)	000 to 377	000 to 377	DE	DE Register pair
HL (5)	000 to 377	000 to 377	HL	HL Register pair
PC (6)	000 to 377	000 to 377	PC	Program counter

NOTE: The contents of any single eight-bit register may lie in the range of 000 to 377 octal. The stack pointer (SP) and the program counter (PC) are 16-bit registers and are displayed as two sets of three octal numbers. Each 3-digit grouping corresponds to one byte (8 bit number). When a register pair is displayed, the left three digits correspond to the left register and the middle three digits correspond to the right register. For example:

256 312 AF

Register A contains 256 and F contains 312.

Altering the Contents of a Selected Register

To alter the contents of a register (or register pair), you must first specify it as described in the preceding paragraphs. After you select the register or register pair, press the ALTER key. This will cause the six left-hand decimal points to rotate right to left, indicating that you may enter 6 digits to alter the contents of the indicated register or register pair.

Alternatively, you may press one of the following command keys:

<u>KEY</u>	<u>FUNCTION</u>
+	Changes the register pair being displayed.
-	Changes the register pair being displayed.
MEM	Specify a new memory address (leave the alter register mode).
REG	Specify a new register for display (leave alter register mode).
ALTER	Exit the register alter mode.

NOTE: Stack pointer register (SP) is not a direct display of the real stack pointer register, but simply a copy of the real stack pointer register and is used for display purposes only. The stack pointer cannot be altered from the front panel. To alter the stack pointer register, an SPHL (SPHL = 371) instruction must be written into memory. The desired new stack pointer value is then placed in the HL register pair. PAM-8's single instruction mode is used to execute the SPHL swap instructions, loading the stack pointer with the contents loaded in the HL register pair.

Stepping Through the Registers

Use + and - keys to change the register pair being displayed. For example, if the DE register pair is being displayed, press the + key causes the next sequential register pair to be displayed (the HL pair). In the same manner, pressing the - key causes the register to decrement to the preceding pair. For example, if the DE pair is being displayed, pressing the - key displays the BC register pair. NOTE: Holding down either the + key or the - key causes the display to continuously increment or decrement through all the six registers/register pairs.

PROGRAM EXECUTION CONTROL

PAM-8 supports three basic program execution control facilities:

- Beginning or starting execution.
- Breakpointing.
- Single instruction.

Each of these execution controls permits the programmer to execute the desired portions of a program and examine its effects. He may execute the entire program, or a small group of instructions, or a single program instruction.

Initiating Program Execution

To begin the execution of a program residing in H8 memory, place the address of the first instruction to be executed in the PC (program counter). Use the methods described in “Displaying and Altering Registers” (Page 1-14). Once the address of this first instruction is placed in the program counter, press the GO key and program execution will begin. NOTE: Unless the program disables the front panel, the display continues to be actively updated, although the front panel commands are no longer active (except for RST and RTM). If the program counter is displayed when you press the GO key, PAM-8 continuously monitors the program counter.

Breakpointing

Breakpointing permits the programmer to execute small portions of a program and then return to PAM-8. Breakpointing is especially useful when a program is being “debugged.” Small portions of the program may be executed and their results observed. If there is an error, it may be corrected before an entire program is involved.

When the H8 executes a program and encounters a halt instruction, it re-enters PAM-8 and sounds the alarm. All of the registers are preserved and the program counter points to the address **following** the address of the halt instruction. Thus, you can breakpoint a program from the front panel by inserting halt instructions (HLT = 166) at the desired points throughout the program. When a particular

section of the program is tested and the breakpoint feature is no longer required, you can change the halt to a NOP (NOP = 000). Once the halts are changed to NOPs, execution of the NOP simply passes control to the next successive instruction. Program execution for breakpointing uses the GO key as described above.

NOTE: If you temporarily replace an existing instruction with a halt, you must restore the instruction before resuming program execution. The contents of the program counter point to the address **following** the halt. Therefore, if the instruction which replaced the halt is to be executed, when the program continues, the contents of the program counter must be decremented one location before execution is resumed.

Single Instruction Operation

Any user program may be operated in the single instruction mode. This procedure is identical to the GO command, except that the SI key is pressed rather than the GO key. When the SI key is pressed, a single **instruction** (not a single machine cycle) is executed and then control is returned to PAM-8. Single instruction operation is available for careful inspection of program results and for executing special programs, such as swapping the HL register pair with the stack pointer as discussed in "Altering the Contents of a Selected Register" (Page 1-15).

Interrupting a Program During Execution

You can interrupt a running program (with all registers preserved at the point of interruption) by pressing RTM & 0. You can then examine and/or alter the contents of various memory locations and all the registers as required. Resume execution of the program at the next sequential instruction by simply pressing the GO key. NOTE: Although all registers and memory locations are preserved when RTM & 0 are pressed, it is very difficult to stop a program at an exact location. Therefore, use the breakpoint feature if you want to stop the program at an exact location.

LOAD/DUMP ROUTINES

PAM-8 contains a routine that lets you load and dump memory contents from or to a tape. This feature is especially important, as most computers require one or two successive "boot strap" routines to be hand-loaded before a desired program can be loaded into the main memory. All these "boot strap" routines are contained within the PAM-8 ROM, and use sophisticated error checking techniques. Thus, a program can be loaded or dumped by simply pressing a single key.

Loading From Tape

To load from a tape, ready the reader device with the tape to be loaded prior to executing the load command. Place PAM-8 in the display memory mode and press the LOAD key. Once the LOAD key is pressed, PAM-8 starts the tape transport and scans the tape for the first file record.

No change will be seen on the front panel displays until PAM-8 finds the first file. When the first file record is located, PAM-8 checks it to see if it is the first (or only) record in a sequence, and the record is a memory dump record. If it is not a memory dump record, a number two error is flagged (see "Tape Errors" on Page 1-20).

Once a correct record is found, loading proceeds. The loading procedure places the entry point address of the program being loaded in the H8 program counter. The H8 memory is then loaded. The displays continuously show the address being loaded and the data being loaded at these addresses. When the load is complete, PAM-8 sounds a long beep and displays the final memory address. If the load is faulty, a number one error is displayed and the audio alert continuously beeps. (See "Tape Errors," Page 1-20.)

NOTE: You may abort a partial load by using the CANCEL key. Naturally, the load image resulting from this action is incorrect, and should not be executed.

Dumping to Tape

Before dumping a memory image onto tape, the following three dump parameters are required:

- The entry point address (the program starting address).
- The dump starting address.
- The dump ending address.

Set the desired entry point address by placing this value in the program counter (PC). This value will be placed in the program counter whenever you load the program so execution will begin at this address when you press the GO key.

Place the dump starting address into the first two H8 RAM cells. These are: 040 000 (offset octal) and 040 001 (offset octal). NOTE: The low order byte of the address should be placed into location 040 000 and the high order byte of the starting address should be placed into location 040 001.

Enter the dump ending address as a memory address using the # (MEM) key. Then ready the tape transport and press the DUMP key. As the tape dump takes place, the number of bytes left to be dumped and the contents of the memory location being dumped are displayed on the front panel. You can abort a dump by using the CANCEL key. If the CANCEL key is used, an incomplete dump image is left on the tape. This cannot be loaded at a future date. NOTE: A successful load automatically sets up the following three dump parameters:

- A. The program starting locations are stored in locations 040 000 and 040 001.
- B. The program ending location is displayed.
- C. The program counter contains the program entry point.

Figure 1-3A shows the steps of a typical dump sequence and Figure 1-3B shows the steps of a typical load sequence.

1. Set PC to 040 100; (040 100 = entry address).
2. Set 040 000 to 100 (100 = low byte of dump start).
3. Set 040 001 to 040 (040 = high byte of dump start).
4. Enter memory address 052 340 (052 340 = end address of dump).
5. Be sure tape is ready.
6. Press DUMP.

Figure 1-3A
The H8 memory image dump.

1. Be sure tape is ready.
2. Press LOAD.

Figure 1-3B
The H8 memory image load.

Copying a Tape

The beginning and final address of the load image are placed at the appropriate points. Thus, to copy a tape, simply load the tape as described in "Loading From Tape" (Page 1-18). Then ready the dump tape drive and press the DUMP key. A dump then takes place, including entry point, initial address, and final address.

In a similar manner, to load, alter, and then dump, enter only the ending address. The other parameters are unchanged from the load if locations 040 000, 040 001 or the program counter have not been modified during the altering procedure.

Tape Errors

PAM-8 detects two types of tape errors: record errors and checksum errors. In either case, when an error is detected, the tape transport is halted. The error number is then displayed in the center three digits (001 for a checksum error, 002 for a record error) and the alarm is repeatedly sounded. To halt the alarm and return to the command mode, press the CANCEL key.

RECORD ERRORS

The following are typical causes of record errors.

- Attempting to load a file which is not a memory image. For example, loading an editor text file or a BASIC program file.
- Attempting to start a load in the middle of a load image. Therefore missing the initialization information at the start of the file.
- A tape error which causes a portion of the load image to be missed so the next record read is not in the proper sequence.

CHECKSUM ERRORS

A checksum error is flagged when the CRC (Cyclical Redundancy Check) checksum following a record does not match the CRC calculated by PAM-8. This error means that the record is either incorrectly recorded or the load is faulty. In either case, the load should be attempted again. If successive loads result in repeated failures, the original tape must be suspected as faulty.

I/O FACILITIES

PAM-8 supports two commands that allow you to perform input and output functions on H8 I/O ports. These front panel instructions permit simple manipulation of the H8 I/O ports without your having to write extensive routines to perform these functions.

Inputting From a Port

To input from a port, press the # key. Then enter three zero digits and the three-digit address (octal) of the desired port. NOTE: The front panel should now display 000 AAA, where AAA is the port address and 000 is meaningless. Press the IN key to read the port, the value is displayed in the three left-most digits of the front panel display.

Outputting to a Port

To output to a specified port, press the # key. Then enter the value to be supplied to the port in the three left-most displays. The port address is entered into the middle three displays. The display is of the form VVV AAA, where V stands for value, and A for address. Pressing the OUT key causes the value to be outputted to the indicated port.

Addressing Port Pairs

Frequently, ports are assigned in pairs, where one of the two port addresses is the control and status register and the other port is the data port. Address port pairs by using the + and - key to change ports. Once the initial port has been defined, the + key increments the port address to a new higher numbered port, and the - key is used to decrement to a lower numbered port.

ADVANCED CONTROL

One of the advanced features of PAM-8 is its provisions allowing sophisticated users to augment or replace PAM-8's functions. Augmenting or replacing PAM-8 functions is usually done in conjunction with assembly language programs. Sometimes it is possible to implement these features by using the POKE and PEEK commands in BASIC. The sample exercise in "Appendix B" (Page 1-64) uses several PAM-8 functions, including the clock, I/O, and the audio alarm.

The following discussion refers to symbols and locations defined in the PAM-8 program listing, given in its complete form as "Appendix A." It is recommended that you review the PAM-8 listing in order to become familiar with its various features. This can be done in conjunction with reading the following section, or independently. In either case, a first overview followed by a detailed analysis of the listing is probably necessary for a complete understanding.

16-Bit Tick Counter (TICCNT)

PAM-8 maintains a 16-bit (2 byte) tick counter known as TICCNT. The value of this counter is incremented each time a clock interrupt is processed. As an interrupt occurs once every 2 mS, the counter is incremented once every 2 mS. As long as clock interrupts are not disabled, this value can be used by any program to compute elapsed time. The tick counter may be set to any desired value, but it should not be frequently reset, as this interferes with the front panel refresh cycle. The contents of the tick counter are contained in memory locations 040 033 (the least significant byte) and 040 034 (the most significant byte).

Using the Keypad

When your program is running, PAM-8 does not recognize any single key command. Thus, all single key patterns are available for your program. To read keypad patterns, you can use one of two routines. First, you may take an input from port IP. PAD; or second, your program may use PAM-8's RCK routine. The input port IP. PAD is permanently assigned to port location 360. Inputting a binary number from this port detects which of the 16 keys are depressed. These results are shown in the table on Page 1-57 of "Appendix A."

A far more sophisticated keypad routine is available to you in the RCK (read Console Keypad) routine. This is also described in "Appendix A" (see Page 1-57). RCK provides keypad decoding, keypad debounce routines, auto-repeat routines, and acoustical feedback.

NOTE: If you use two key combinations, each key must reside in a separate bank. The first bank includes keys 0-7 and the second bank includes keys 8-#. RCK cannot decode two key combinations.

Display Usage

When a user program is running, PAM-8 normally displays the contents of the selected register or memory location. However, you may disable this process and display any arbitrary segment pattern, or completely disable the display to provide greater computational through-put. The display usage is primarily controlled by setting various bits in the .MFLAG memory cell. This memory cell is found at location 040 010.

MANUAL UPDATING

By setting the UO.DDU (see "Appendix A," Page 1-25, for an explanation of the user option bits, UO.XXX) bit in the .MFLAG memory location, you can instruct PAM-8 to continue refreshing the front panel displays and to disable updating. When this is done, PAM-8 continues to refresh the LED's from a 9-byte block of RAM cells found at locations 040 013 through 040 023. A description of these front panel LED's (FPLEDS) is found in "Appendix A" (see Page 1-60). When the UO.DDU bit is set in .MFLAG, the contents of these bytes are not altered in any manner by PAM-8.

You can use this technique to display numbers, letters, or arbitrary bar patterns (see Page 1-58) on the front panel displays. For instance, your program may alter the display by inserting any value into FPLEDS. The front panel LED segments will display a decimal integer if you use the octal to 7-segment pattern (DODA) display.

MANUAL DISPLAY REFRESHING

By setting the UO.NFR (User Option.No Front Panel Refresh) bit in the .MFLAG memory cell, you can instruct PAM-8 to stop refreshing the front panel displays. Setting the UO.NFR bit does not disable the clock interrupts; therefore, the tick counter (TICCNT) is still incremented. But PAM-8 does not refresh the displays from the information contained in the FPLEDS bytes.

NOTE: If you desire, you may write a program to refresh the front panel LED displays. Usually this is done using the clock interrupts. If you undertake an independent front panel refresh program, take extreme care to avoid burning the displays due to excessive refreshing. **The total power dissipated in the LEDs is determined by the refresh cycle, and too frequent refreshing will result in excessive display heating.**

Using Interrupts

All H-8 interrupts cause control to be transferred into the low 64 bytes of memory. PAM-8 occupies this memory space so all interrupts are first processed by PAM-8. Except for level zero interrupts, which are used as master clears, you can supply an interrupt processing routine for each of the seven additional interrupts. The following sections explain the use of each of these interrupts.

I/O INTERRUPTS

Interrupts numbered 3 through 7 are I/O interrupts. PAM-8 does not process these interrupts in any way. When a level 3 through level 7 interrupt is received, PAM-8 immediately transfers to the user interrupt vectors contained in memory locations 040 037 through 040 064. These locations are listed in "Appendix A" (see Page 1-60). Each location must contain a jump instruction pointing to the appropriate program location which processes these interrupts.

NOTE: If any of these interrupts occur, you must supply a processing routine for them. This routine must be complete including both entry and exit processing. When you use H8 interrupts, you must use only the available vector which is 6 to insure compatibility with future H8 products. You may also use 2 if you will not be using BUG-8.

CLOCK INTERRUPTS

The level one interrupts are generated by the front panel hardware every 2 mS. PAM-8 normally processes these interrupts. However, by setting a processing vector in UIVEC and setting the UO.INT bit in the MFLAG cell, PAM-8 enters the users routine each time a clock interrupt is generated. "Appendix A" (see Page 1-31) gives the required entry and exit conditions for processing clock interrupts.

SINGLE INSTRUCTION AND BREAKPOINT INTERRUPTS

Level two interrupts are generated by the single instruction hardware contained on the CPU card. When a single instruction is requested, the result of the interrupt is processed by PAM-8. If the single instruction interrupt was generated by PAM-8 in response to a Monitor Mode Single Instruction register condition, PAM-8 processes it. Otherwise, PAM-8 jumps to the user level two interrupt vector (UIVEC). Since the level two interrupt does not affect PAM-8, a level two restart instruction can be used as a breakpoint instruction by the user programs.

APPENDIX A

Panel Monitor Listing

This appendix contains a complete listing of the PAM-8 front panel monitor program. PAM-8 resides in the low 1,024 bytes of the H8 computer. It provides all the control for front panel operation, and cassette or paper tape load and dump facilities. It also provides for master clear and front panel interrupt processing. PAM-8 presumes RAM cells are available for its use in locations 040 000 through 040 077 and 80 bytes are available in high memory for a stack. The use of these RAM cells is described on Page 1-60 of this Appendix and in the memory map on Page 0-36.

Pages 1-61, 1-62, and 1-63 of this Appendix are a symbolic reference table. Use this table to find the program locations where each symbolic address is used. Symbolic addresses are listed in alphabetical sequence.

FAM/8 - HB FRONT PANEL MONITOR. #01.00.00,
INTRODUCTION.

HEATH X86ASM V1.1 06/21/77
15:43:50 01-APR-77 PAGE 1

4 *** FAM/8 - HB FRONT PANEL MONITOR.

5 * JGL, 05/01/76.

6 * FOR *WINTER* INC.

7 * COPYRIGHT 05/1976, WINTER CORPORATION,

902 N. 9TH ST.

LAFAYETTE, IND.

14 *** FAM/8 - HB FRONT PANEL MONITOR.

15 * THIS PROGRAM RESIDES IN ROM IN THE LOW 1024 BYTES OF THE HEATH
16 * HB COMPUTER. IT ACTUALLY CONSISTS OF TWO VIRTUALLY INDEPENDENT
17 * ROUTINES; A TASK-TIME PROGRAM WHICH PROVIDES SOPHISTICATED
18 * FRONT PANEL MONITOR SERVICE, AND AN INTERRUPT-TIME PROGRAM WHICH
19 * PROVIDES BOTH A REAL-TIME CLOCK AND EMULATES AN EFFECTIVE
20 * HARDWARE FRONT PANEL.

21 *

22 * ****
23 * INTERRUPTS.
24 *

25 * FAM/8 IS THE PRIMARY PROCESSOR FOR ALL INTERRUPTS.
26 * THEY ARE PROCESSED AS FOLLOWS:
27 *

28 * RST

29 * USE
30 * 0
MASTER CLEAR. (NEVER USED FOR I/O OR RST)

31 * 1
CLOCK INTERRUPT. NORMALLY TAKEN BY FAM/8,
32 * SETTING BIT #0, CLK* IN BYTE #MFLAG* ALLOWS
33 * USER PROCESSING (VIA A JUMP THROUGH *UIVEC*).
34 * UPON ENTRY OF THE USER ROUTINE, THE STACK
35 * CONTAINS:

36 * (STACK+0) = RETURN ADDRESS (TO FAM/8),
37 * (STACK+1) = (STACKPTR+14)
38 * (STACK+2) = (AF),
39 * (STACK+3) = (DE),
40 * (STACK+4) = (BC),
41 * (STACK+5) = (HL),
42 * (STACK+6) = (PC),
43 * (STACK+7) = (FC),
44 * (STACK+8) = (PC).

45 * THE USER'S ROUTINE SHOULD RETURN TO FAM/8 VIA
46 * A *RETI* WITHOUT ENABLING INTERRUPTS.
47 * 2
SINGLE STEP. SINGLE STEP INTERRUPTS GENERATED
48 * BY FAM/8 ARE PROCESSED BY FAM/8.
49 * ANY SINGLE STEP INTERRUPT RECEIVED WHEN IN
50 * USER MODE CAUSES A JUMP THROUGH *UIVEC*+3.
51 * STACK, UPON USER ROUTINE ENTRY;
52 * (STACK+0) = (STACKPTR+12)
53 * (STACK+2) = (AF),
54 * (STACK+4) = (BC)

FAM/B - HB FRONT PANEL MONITOR #01.00.00.
INTRODUCTION.

HEATH X8ASM V1.1 06/21/77
15.43:51 01-AFR-77 PAGE 2

```
55 *      (STACK+6) = (DE)
56 *      (STACK+8) = (HL)
57 *      (STACK+10) = (PC)
58 *      THE USER'S ROUTINE SHOULD HANDLE ITS OWN RETURN
59 *      FROM THE INTERRUPT.
60 *
61 *
62 *      THE FOLLOWING INTERRUPTS ARE VECTORED DIRECTLY THROUGH *UIVEC*.
63 *      THE USER ROUTINE MUST HAVE SETUP A JUMP IN *UIVEC* BEFORE ANY
64 *      OF THESE INTERRUPTS MAY OCCUR.
65 *
66 *      3      I/O 3. CAUSES A DIRECT JUMP THROUGH *UIVEC*+8
67 *      4      I/O 4. CAUSES A DIRECT JUMP THROUGH *UIVEC*+9
68 *
69 *      5      I/O 5. CAUSES A DIRECT JUMP THROUGH *UIVEC*+12
70 *
71 *      6      I/O 6. CAUSES A DIRECT JUMP THROUGH *UIVEC*+15
72 *
73 *      7      I/O 7. CAUSES A DIRECT JUMP THROUGH *UIVEC*+18
```

PAN8 - H8 FRONT PANEL MONITOR #01.00.00
ASSEMBLY CONSTANTS.

HEATH X8ASM V1.1 06/21/77
15:43:52...01-APR-77...PAGE 3

77 ** ASSEMBLY CONSTANTS

```

79 ** I/O PORTS
 80 IP.PAD EQU 360Q          PAD INPUT PORT
 81 IP.PAU EQU 360Q          PAD INPUT PORT
 82 OP.CTL EQU 360Q          CONTROL OUTPUT PORT
 83 OP.DIG EQU 360Q          DIGIT SELECT OUTPUT PORT
 84 OP.SEG EQU 361Q          SEGMENT SELECT OUTPUT PORT
 85 IP.TPC EQU 371Q          TAPE CONTROL IN
 86 OP.TPC EQU 371Q          TAPE CONTROL OUT
 87 IP.TPD EQU 370Q          TAPE DATA IN
 88 OP.TPD EQU 370Q          TAPE DATA OUT

90 ** ASCII CHARACTERS.
 91 A.SYN EQU 026Q          SYNC CHARACTER
 92 A.STX EQU 002Q          STX CHARACTER

95 ** FRONT PANEL HARDWARE CONTROL BITS.
 96 CB.SSI EQU 00010000B     SINGLE STEP INTERRUPT
 97 CB.MTL EQU 00100000B     MONITOR LIGHT
 98 CB.CLI EQU 01000000B     CLOCK INTERRUPT ENABLE
 99 CB.SPK EQU 10000000B     SPEAKER ENABLE

102 ** DISPLAY MODE FLAGS (IN *DSMOD*)
 103 DM.MR EQU 0             MEMORY READ
 104 DM.MW EQU 1             MEMORY WRITE
 105 DM.RW EQU 2             REGISTER READ
 106 DM.RR EQU 3             REGISTER WRITE
 107 DM.RM EQU 3             TAPE DEFINITIONS
 108 XTEXT EQU 1             TAPE EQUIVALENCES

110X ** TAPE EQUIVALENCES.
 111X RT.MI EQU 1             RECORD TYPE - MEMORY IMAGE
 112X RT.BF EQU 2             RECORD TYPE - BASIC PROGRAM
 000.001 EQU 1             RECORD TYPE - COMPRESSED TEXT
 000.002 EQU 2
 000.003 EQU 3

115X ** BLOCK SIZE FOR INTER-PRODUCT COMMUNICATION.
 116X
 117X BLKSIZE EQU 512
 002.000 EQU 118X
 119X

```

PAM78 - H8 FRONT PANEL MONITOR \$01.00.00.
ASSEMBLY CONSTANTS.

HEATH X8ASM V1.1 06/21/77
15:43:56 Q1-AFR-77 PAGE 4

121 ** MACHINE INSTRUCTIONS.

000.166	122 MI.HLT EQU 01110110B HALT
000.311	123 MI.RET EQU 11001001B RETURN
000.333	124 MI.IN EQU 11011011B INPUT
000.323	125 MI.OUT EQU 11010011B OUTPUT
000.072	126 MI.LDA EQU 0011010B LDA
000.349	127 MI.ANI EQU 111000110B ANI
000.021	128 MI.LXID EQU 00010001B LXI D

131 ** USER OPTION BITS.

132 *	THESE BITS ARE SET IN CELL .MFLAG.
133 *	
000.200	134 UD.HLT EQU 10000000B DISABLE HALT PROCESSING
000.100	135 UD.NFR EQU 00000001B NO REFRESH OF FRONT PANEL
000.002	136 UD.CLI EQU 000000010B DISABLE DISPLAY UPDATE
000.001	137 UD.DCU EQU 000000010B ALLOW CLOCK INTERRUPT PROCESSING
000.000	138 UD.CLK EQU 00000001B

140 XTEXT U8251 DEFINE 8251 USART BITS

PAM/8 - H8 FRONT PANEL MONITOR #01.00.00.
8251 USART BIT DEFINITIONS.

HEATH X8ASM V1.0 02/18/77
13:23:23 01-APR-77 PAGE 5

143X ** 8251 USART BIT DEFINITIONS.

144X *

145X **** MODE INSTRUCTION CONTROL BITS.

147X

148X UMI.1B EQU 0100000B 1 STOP BIT

149X UMI.HB EQU 1000000B 1 1/2 STOP BITS

150X UMI.2B EQU 1100000B 2 STOP BITS

000.040 UMI.FE EQU 0010000B EVEN PARITY

000.020 UMI.PA EQU 0001000B USE PARITY

000.000 UMI.L5 EQU 0000000B 5 BIT CHARACTERS

000.004 UMI.L6 EQU 00000100B 6 BIT CHARACTERS

000.010 UMI.L7 EQU 00001000B 7 BIT CHARACTERS

000.014 UMI.L8 EQU 00001100B 8 BIT CHARACTERS

000.001 UMI.1X EQU 00000001B CLOCK X 1

000.002 UMI.16X EQU 00000010B CLOCK X 16

000.003 UMI.64X EQU 00000011B CLOCK X 64

161X *** COMMAND INSTRUCTION BITS.

162X

000.100 UCI.IR EQU 0100000B INTERNAL RESET

000.040 UCI.R0 EQU 0010000B READER-ON CONTROL FLAG

000.020 UCI.ER EQU 0001000B ERROR RESET

000.004 UCI.RE EQU 00000100B RECEIVE ENABLE

000.002 UCI.IE EQU 00000010B ENABLE INTERRUPTS FLAG

000.001 UCI.TE EQU 00000001B TRANSMIT ENABLE

170X *** STATUS READ COMMAND BITS.

171X

172X USR.FE EQU 0010000B FRAMING ERROR

173X USR.DE EQU 0001000B OVERRUN ERROR

000.040

000.020 USR.PE EQU 00001000B PARITY ERROR

000.010

000.004 USR.TXE EQU 00000100B TRANSMITTER EMPTY

000.002

000.001 USR.RXR EQU 00000010B RECEIVER READY

000.001

177X USR.TXR EQU 00000001B TRANSMITTER READY

FAM/B - HB FRONT PANEL MONITOR #01.00.00, HEATH X8ASM V1.0 02/18/77
 13:23:25 01-APR-77 PAGE 6

180 *** INTERRUPT VECTORS.

181 *

182

184 ** LEVEL 0 - RESET

185 * THIS 'INTERRUPT' MAY NOT BE PROCESSED BY A USER PROGRAM.

186 *

187

000,000 188 ORG 00A

000,000 021 371 003 190 INIT0 LXI D,FRSR0M (DE) = ROM COPY OF PRS CODE
 000,003 041 012 040 191 LXI H,PRSRN+FRSL-1 (HL) = RAM DESTINATION FOR CODE
 000,006 303 073 000 192 IMP INIT
 377,073 193 ERPL INIT-1000A INITIZE
 BYTE IN WORD 10A MUST BE 0

195 ** LEVEL 1 - CLOCK

000,010 196 EQU 10A INTERRUPT ENTRY POINT

000,000 197 INT1 EQU 10B

000,011 198 ERNZ *-10B INTO TAKES UP ONE BYTE
 315 132 000 199 CALL SAVALL SAVE USER REGISTERS
 000,014 028 000 200 HLT D,0
 000,016 303 201 000 201 JMF CLOCK
 377,201 203 ERPL CLOCK-1000A PROCESS CLOCK INTERRUPT
 EXTRA BYTE MUST BE 0

205 ** LEVEL 2 - SINGLE STEP

206 * IF THIS INTERRUPT IS RECEIVED WHEN NOT IN MONITOR MODE,
 207 * THEN IT IS ASSUMED TO BE GENERATED BY A USER PROGRAM
 208 * (SINGLE STEPPING OR BREAKPOINT). IN SUCH CASE, THE
 209 * USER PROGRAM IS ENTERED THROUGH (UIVEC+3)

000,020 210 EQU 20A LEVEL 2 ENTRY

212 INT2 EQU 20A

000,000 213 ERNZ *-21A INT1 TAKES EXTRA BYTE
 000,021 315 132 000 214 CALL SAVALL SAVE REGISTERS
 000,024 032 215 LDAX D (A) = (CTLFLG)
 040,011 216 SET CTLFLG
 000,025 303 244 001 217 JMPL SPRTN
 218 * STEF RETURN

220 *** I/O INTERRUPT VECTORS.

221 *

222 * INTERRUPTS 3 THROUGH 7 ARE AVAILABLE FOR GENERAL I/O USE.

223 *

224 * THESE INTERRUPTS ARE NOT SUPPORTED BY FAM/B; AND SHOULD
 225 * NEVER OCCUR UNLESS THE USER HAS SUPPLIED HANDLER ROUTINES
 226 * (THROUGH UIVEC).

227

FAM/B - H8 FRONT PANEL MONITOR #01.00.00.

HARDWARE INTERRUPT VECTORS

HEATH XBASM V1.0 02/18/77
13:23:26 01-APR-77 PAGE 7

```

000.030      228      ORG    30A
000.030      229      INT3   JMP    UIVEC+6   JUMP TO USER ROUTINE
000.030      230      DB     24413    HEATH PART NUMBER 444-13

000.040      233      ORG    40A
000.040      234      INT4   JMP    UIVEC+9   JUMP TO USER ROUTINE
000.043      235      DB     100Q,112Q,107Q,114Q,100Q SUPPORT CODE
000.043      236      DB     100Q,112Q,107Q,114Q,100Q SUPPORT CODE

000.050      238      ORG    50A
000.050      239      INT5   JMP    UIVEC+12  JUMP TO USER ROUTINE
000.050      240      DB     24413    HEATH PART NUMBER 444-13

241      **      DLY - DELAY TIME INTERVAL.
242      *      ENTRY (A) = MILLISECOND DELAY COUNT/2
243      *      EXIT   NONE
244      *      USES   A,F
245      *      USES   A,F
246      *      USES   A,F

000.053      365      ORG    F5H
000.054      248      DLY   FLUSH  F5H      SAVE COUNT
000.055      249      XRA   A      DONT SOUND HORN
000.055      250      JMP   HRNO  PROCESS AS HORN

000.060      252      ORG    60A
000.060      253      INT6   JMP    UIVEC+15  JUMP TO USER ROUTINE
000.060      254      DB     24413    HEATH PART NUMBER 444-13

000.063      076,320  MUL   ACB,SS1CB,CL1+CB,SPK OFF MONITOR MODE LIGHT
000.065      001      256,60,  JMP   SS1      RETURN TO USER PROGRAM

000.070      259      ORG    70A
000.070      260,1NT7  JMP    UIVEC+1B   JUMP TO USER ROUTINE

```

```

FAM/B - HB FRONT PANEL MONITOR #01,00,00,..... HEATH XBASM V1.0 02/18/77
MASTER CLEAR PROCESSING ..... 13:23:28 01-AFR-77 PAGE 8

263 ** INIT - INITIALIZE SYSTEM
264 * INIT IS CALLED WHENEVER A HARDWARE MASTER-CLEAR IS INITIATED.
265 *
266 *
267 * SETUP FAM/B CONTROL CELLS IN RAM.
268 * DECODE HOW MUCH MEMORY EXISTS, SETUP STACKFOINTER, AND
269 * ENTER THE MONITOR LOOP.
270 *
271 * ENTRY FROM MASTER CLEAR
272 * EXIT INTO FAM/B MAIN LOOP
273
274 000,073 032 INIT LDAX D COPY *FRSRDNX INTO RAM
000,074 16 MOV H,A MOVE BYTE
000,075 051 DCX H DECREMENT DESTINATION
000,076 034 INR E INCREMENT SOURCE
000,077 302,073,000 JNZ INIT IF NOT DONE
004,000
281 SINCR EQU 4000A SEARCH INCREMENT
282 MVI D,SINCR/256 (DE) = SEARCH INCREMENT
000,102 026,004 LXI H,START-SINCR (HL) = FIRST RAM - SEARCH INCREMENT
000,104 041 000 034
285
286 * DETERMINE MEMORY LIMIT.
287
000,107 167 INITI MOV H,A RESTORE VALUE READ
000,110 031 DAD D INCREMENT TRIAL ADDRESS
000,111 176 MOV A,H (A) = CURRENT MEMORY VALUE
000,112 065 DCR M TRY TO CHANGE IT
000,113 276 CMP M
000,114 302,107,000 JNE INITI IF MEMORY CHANGED
000,117 053 INIT2 DCX H SET STACKFOINTER = MEMORY LIMIT -1
000,120 371 SFHL H SET *FC* VALUE ON STACK
000,121 345 PUSH H
000,122 041,322,000 297 LXI H,ERROR
000,125 345 298 PUSH H SET 'RETURN' ADDRESS
300
301 * CONFIGURE LOAD/DUMP UART
302
000,126 076,116 MVI A,UMI;1BTUMI,L8+UMI;16X
000,130 323,371 OUT 0F,TFC SET 8 BIT, NO PARITY, 1 STOP, X16

```

PAM/8 - H8 FRONT PANEL MONITOR #01.00.00.
INTERRUPT TIME SUBROUTINES

HEATH X8ASM V1.0 02/18/77

13:23:29 01-AFR-77 PAGE 9

```

307 ** SAVALL - SAVE ALL REGISTERS ON STACK.
308 * SAVALL IS CALLED WHEN AN INTERRUPT IS ACCEPTED, IN ORDER TO
309 * SAVE THE CONTENTS OF THE REGISTERS ON THE STACK.
310 * ENTRY CALLED DIRECTLY FROM INTERRUPT ROUTINE.
311 * EXIT ALL REGISTERS PUSHED ON STACK,
312 * IF NOT YET IN MONITOR MODE, REGPTR = ADDRESS OF REGISTERS
313 * ON STACK,
314 * (IE) = ADDRESS OF CTLFLG.
315 *
316 *
317
318 000.132 343 SAVALL XTHL SET H,L ON STACK TOP.
000.133 325 FUSH D
000.134 305 PUSH B
000.135 365 PUSH FSW
000.136 353 XCHG (D,E) = RETURN ADDRESS.
000.137 041 012 000 323 LXI H,10 (H,L) = ADDRESS OF USERS SF
000.142 071 DAD SF SET ON STACK AS 'REGISTER'.
000.143 345 324 PUSH H SET RETURN ADDRESS.
000.144 325 325 LXI D,CTLFLG
000.145 021 011 046 326 PUSH D
000.150 032 328 LXI D
000.151 057 329 LDAX D (A) = CTLFLG.
000.152 346 060 330 CMA
000.154 310 331 ANI CR,MTL+CR,SSI SAVE REGISTER ADDRS IF USER OR SINGLE STEP.
000.155 041 002 000 332 RZ RETURN IF WAS INTERRUPT OF MONITOR LOOP.
000.160 071 333 LXI H,2
000.161 042 035 040 334 DAD SF (H,L) = ADDRESS OF 'STACKPTR' ON STACK
000.164 311 SHLD REGPTR
RET
336 **
337 CUI - CHECK FOR USER INTERRUPT PROCESSING.
338 * CUI IS CALLED TO SEE IF THE USER HAS SPECIFIED PROCESSING
339 * FOR THE CLOCK INTERRUPT.
340 *
341 *
342 *
343 *
344 040.010 344 CUI SET 'MFLAG' REFERENCE TO MFLAG.
000.165 012 345 CUI INTXIT POP PSW REMOVE FAKE 'STACK REGISTER'
000.000 . 346 ERRNZ B (A) = +MFLAG
000.166 017 . 347 RRC CODE ASSUMED = 01.
000.167 334.037.040 348 CC UIVEC IF SPECIFIED, TRANSFER TO USER.
349
350 *
351 RETURN TO PROGRAM FROM INTERRUPT.
352 INTXIT POP PSW
353 POP PSW
354 POP B
355 POP D
356 POP H
357 EI
358 RET

```

PAM/8 - HB FRONT PANEL MONITOR * 01,00,00,
PROCESS CLOCK INTERRUPTSHEATH X8ASM V1.0 02/18/77
13:23:31 01-APR-77 PAGE 10

```

361 *** CLOCK = PROCESS CLOCK INTERRUPT.
362 *      CLOCK IS ENTERED WHENEVER A MILLISECOND CLOCK INTERRUPT IS
363 *      PROCESSED.
364 *
365 *
366 *      TICCNT IS INCREMENTED EVERY INTERRUPT.
367
368
369 CLOCK LHDN TICCNT
370 INX H
371 SHLD TICCNT
372
373 ** REFRESH FRONT PANEL.
374 *
375 * THIS CODE DISPLAYS THE APPROPRIATE PATTERN ON THE
376 * FRONT PANEL LEDS. THE LEDs ARE PAINTED IN REVERSE ORDER,
377 * ONE PER INTERRUPT. FIRST, NUMBER 9 IS LIT, THEN NUMBER 8,
378 * ETC.
379
380
381 LXI H, MFLAG
382 MOV A,M
383 MOV B,A
384 ANI U,NFR
385 INX H
386 ERNZ CTLFLG-,NFLAG-1
387 MOV A,M
388 MOV C,B
389 JNZ CLN3
390 INX H
391 ERNZ REFIND-CTLFLG-1
392 DCR H DECREMENT DIGIT INDEX
393 JNZ CLN2
394 MOV H,I
395 CLK2 MOU E,M
396 DAD D
397 MOU C,E
398 CLK3 ORA *
399 OUT C,A
400 OUT OF,DIG
401 MOU A,M
402 OUT OF,SEG
403
404 * SEE IF TIME TO RECODE DISPLAY VALUES.
405
406 HUI L,*TICCNT
407 MOV A,M
408 ANI 37Q
409 CZ UFD
410
411 * EXIT CLOCK INTERRUPT.
412
413 LXI B,CTLFLG
414 LDAX B
415 ANI CB,ML
416 JNZ INTXT

```

(A) = CTLFLG
 (B) = CURRENT FLAG
 (C) = O IN CASE NO PANEL DISPLAY
 (D) = NOT WRAP-AROUND
 (E) = INDEX + FIXED BITS
 (F) = SELECT DIGIT
 (H,L) = ADDRESS OF PATTERN
 (I) = REFIND
 (J) = CLN1
 (K) = CLN2
 (L) = CLN3
 (M) = NOT WRAP AROUND
 (N) = DISPLAY AROUND
 (O) = ADDRESS OF PATTERN
 (P) = CTLNLG
 (Q) = INDEX + FIXED BITS
 (R) = SELECT SEGMENT
 (S) = UPDATE FRONT PANEL DISPLAYS
 (T) = RECODE DISPLAY
 (U) = FRONT PANEL MODE
 (V) = MONITOR MODE

FAM/B - H8 FRONT PANEL MONITOR #01.00.00.
PROCESS CLOCK INTERRUPTS

HEATH X86ASM V1.0 02/18/77
13:23:34 01-APR-77 PAGE 11

```

        000,266 013      417      DCX     B
        000,000      418      ERRNZ   CTLFLG-, MFLAG-1
        000,267 012      419      LDAX    B      (A) = MFLAG
        000,000      420      ERRNZ   UD, HALT-2000Q ASSUME HIGH-ORDER
        000,270 027      421      RAL
        000,271 332 313 000 422      JOC     CLK4      SKIP IT
        423      *      NOT IN MONITOR MODE. CHECK FOR HALT
        424      *      NOT IN MONITOR MODE. CHECK FOR HALT
        425      *      NOT IN MONITOR MODE. CHECK FOR HALT
        000,274 076 012      426      MOVI   A, Y10      (A) = INDEX OF YPK REG
        000,276 315 052,003 427      CALL   LRA, LOCATE REGISTER ADDRESS
        000,301 136      428      MOV    E,M
        000,302 043      429      INX    H
        000,303 126      430      MOV    D,M      (D,E) = FC CONTENTS
        000,304 033      431      DCX    D
        000,305 032      432      LDAX   D      CHECK FOR HALT
        000,306 376,166 433      CPI    M,HLT
        000,310 312 322 000 434      JE     ERROR      IF HALT, BE IN MONITOR MODE
        000,310 312 322 000 435      *      CHECK FOR RETURN TO MONITOR KEY ENTRY.
        436      *      CHECK FOR RETURN TO MONITOR KEY ENTRY.
        437      *      CHECK FOR RETURN TO MONITOR KEY ENTRY.
        000,313      438      CLK4      EQU   *
        000,313 333 360 439      IN     IP, PAD
        000,315 376 056 440      CPI    560      SEE IF '56' AND '57'
        000,317 302 165 000 441      JNE   CUI1      IF NOT, ALLOW USER PROCESSING OF CLOCK

```

```

PAM/8 - HB FRONT PANEL MONITOR #01.00.00. HEATH X8ASM V1.1 96/24/77.
..... MTR - MAIN EXECUTIVE LOOP. 15:44:09 01-AFR-77 PAGE 12

        445 *** ERROR - COMMAND ERROR.
        446 *      ERROR IS CALLED AS A 'FAIL-OUT' ROUTINE.
        447 *
        448 *      I.T. RESETS THE OPERATIONAL MODE, AND RESTORES THE STACK POINTER.
        449 *
        450 *      ENTRY NONE
        451 *      EXIT TO MTR LOOP
        452 *      CTLFLG SET
        453 *      MFLAG CLEARED
        454 *      USES ALL.
        455 *
        456

        457
        000.322    041.010.040.    458   ERROR EQU * H,MFLAG.
        000.322    041.010.040.    459   LXI A,M
        000.325    176             460   MOV (A) = MFLAG
        000.325    176             461   ANI 3770-U0;DU-U0,NFR
        000.326    346.275.       462   MOV M,A
        000.330    176             463   REPLACE
        000.331    043             464   INX H
        000.332    066.360.       465   MOVI M,CB,SSI+CB.MTL+CB.CLI+CB.SPK
        000.000    000             466   RESTORE *CTLFLG*
        000.334    373             467   ERNRZ CTLFLG,MFLAG-1.
        000.335    052.035.040.   468   LHLD REGTR
        000.340    371             469   SFHL RESTORE STACK POINTER TO EMPTY STATE
        000.341    315.136.002.   470   CALL ALARM ALARM FOR 200 MS
        000.344    373             471   *** MTR - MONITOR LOOP.
        000.345    041.345.006.   472   *
        000.350    345             473   * THIS IS THE MAIN EXECUTIVE LOOP FOR THE FRONT PANEL EMULATOR.
        000.351    001.007.040.   474
        000.354    012             475   EQU *
        000.355    346.001.       476   MTR EQU *
        000.357    057             477   EI
        000.360    062.006.040.   478   MTRI
        000.363    315.260.003.   479   MTRI
        000.366    052.024.040.   480   FUSH H
        000.371    376.012.       481   LXI B,DISPMOD SET 'MTRI' AS RETURN ADDRESS
        000.373    322.005.001.   482   LDAX B (BC) = #DISPMOD
        000.376    137             483   ANI 1 (A) = 1 IF ALTER
        000.007    002             484   CMA
        001.000    017             485   STA DISPROT ROTATE LED PERIODS IF ALTER
        001.001    332.051.001.   486
        001.001    332.051.001.   487   * READ KEY
        000.363    315.260.003.   488
        000.366    052.024.040.   489   CALL RICK READ CONSOLE KEYPAD
        000.371    376.012.       490   LHLD ABUS
        000.373    322.005.001.   491   CPI 10
        000.376    137             492   INC MTR4
        000.007    002             493   MOV E,A IF IN 'ALWAYS VALID' GROUP
        000.377    012             494   SET DISPMOD SAVE VALUE
        001.000    017             495   LDAX B (A) = DISPMOD
        001.001    332.051.001.   496   FRC
        001.001    332.051.001.   497   JC MTR5 IF IN ALTER MODE

```

FAM/8 - H8 FRONT PANEL MONITOR #01.00.00,
MTR - MAIN EXECUTIVE LOOP.

HEATH X8ASM V1.0 02/18/77
13:23:37 01-AFR-77 PAGE 13

```

001.004 17.3    498    MOV     A,E      (A) = CODE
          499    *      HAVE A COMMAND (NOT A VALUE)
          500    *      (A) = COMMAND
          501    SUI     4      (A) = COMMAND
          502    MTR4   JC     ERROR
          503    MOV     E,A      IF BAD COMMAND
          504    PUSH    H      SAVE ABUSIS VALUE
          505    LXI    H,MTRA
          506    HUI    D,0      (H,L) = ADDRESS OF TABLE ENTRY
          507    DAD    D
          508    MOV     E,H
          509    DAD    D      (H,L) = ADDRESS OF PROCESSOR
          510    XTHL   D      SET ADDRESS, (H,L) = (ABUS)
          511    LXI    D,REGI
          512    D,REGI
          513    SET    DSFMOD
          514    LDAX   B      (A) = DSFMOD
          515    ANI    B      SET 'Z' IF MEMORY
          516    LDAX   B      (A) = DSFMOD
          517    RET
          518
          519    EQU     *      JUMP TABLE
          520    MTRA   DB     GO-*      4 = GO
          521    DB     IN-*      5 = INPUT
          522    DB     OUT-*      6 = OUTPUT
          523    DB     STEP-*      7 = SINGLE STEP
          524    DB     RMEM-*      8 = CASSETTE LOAD
          525    DB     WMEM-*      9 = CASSETTE DUMP
          526    DB     NEXT-*      + = NEXT
          527    DB     LAST-*      - = LAST
          528    DB     ABORT-*      * = ABORT
          529    DB     R#W-*      / = DISPLAY/ALTER
          530    DB     MEMM-*      $ = MEMORY MODE
          531    DB     REGM-*      % = REGISTER MODE
          532    DB
          533    *      PROCESS MEMORY/REGISTER ALTERATIONS.
          534    *      THIS CODE IS ENTERED IF
          535    *
          536    *      1) A# IN ALTER MODE, AND
          537    *      2) A KEY FROM Q-7 WAS ENTERED.
          538    *
          539    *      1) A# IN ALTER MODE, AND
          540    *      2) A KEY FROM Q-7 WAS ENTERED.
          541    MTRS   RRC
          542    MOV     A,E      (A) = VALUE
          543    JC     MTR6      IS REGISTER
          544    STC
          545    CALL   IOR
          546    INX   H      INPUT OCTAL BYTE
          547    *      DISPLAY NEXT LOCATION

```

PAM/8 - HB FRONT PANEL MONITOR *01,00,00,
MTR - MAIN EXECUTIVE LOOP.

HFATH X8BASM.V1,0..02/18/77
13:23:39 01-AFR-77 PAGE 14

```
548 ** SAE = STORE ABUS, AND EXIT.  
549 * ENTRY (HL) = ABUS$ VALUE.  
550 * EXIT TO (RET)  
551 * USES NONE  
552 *  
553 *  
001,063 042,024,040 554 SAE SHLD ABUS  
001,066 311 555 RET  
556  
557 * ALTER REGISTER  
558  
001,067 365 559 MTR6 PUSH FSW SAVE CORE  
001,070 315,047,003 560 CALL LRA LOCATE REGISTER ADDRESS  
001,073 247 561 ANA A  
001,074 342,322,000 562 JZ ERROR  
001,077 043 563 INX H NOT ALLOWED TO ALTER STACKFOINTER  
001,100 361 564 FOF FSW RESTORE VALUE AND CARRY FLAG  
001,101 303 062 003 565 JMP IOA INPUT OCTAL ADDRESS
```

FAM/B - HB FRONT PANEL MONITOR #01.00.00.
MONITOR TASK SUBROUTINES.

HEATH XBASM V1.1 06/21/77
15:44:14 01-APR-77 PAGE 15

```

569 ** REGM = ENTER REGISTER DISPLAY MODE.
570 * ENTRY (A) = DSFMOD
571 * (BC) = *DSFMOD
572 * SET DSFMOD
573 * SET DISPLAY REGISTER MODE
      001.104 076 002    574 REGM MVI A,2 SET DSFMOD
      040.007          575 SET STAX B SET DISPLAY REGISTER MODE
      001.106 002          576 ERNZ DSFMOD-DSFR0T-1
      000.000          577 DSFMOD
      001.107 013          578 DCX (BC) = *DSFR0T
      001.110 252          579 XRA A SET ALL FERIODS ON
      001.111 002          580 STAX B CALL RCK
      001.112 260 003          581 ICR A READ KEY ENTRY
      001.115 075          582 CPI 6 DISPLACE
      001.116 378 006          583 JNC EROR NOT I-6
      001.120 322 322 006          584 RLC D SET NEW REG IND
      001.123 007          585 STAX B SET REGI
      001.124 022          586 SET RETY
      040.005          587
      001.125 311          588
      590 ** RSW = TOGGLE DISPLAY/ALTER MODE.
      591 * ENTRY (A) = DSFMOD
      592 * (BC) = ADDRESS OF DSFMOD
      593 *
      594 SET DSFMOD
      040.007          595 XRI 1
      001.126 356 001          596 R$4 SET DSFMOD
      001.130 002          597 STAX B
      001.131 311          598 RET
      600 ** NEXT = INCREMENT DISPLAY ELEMENT.
      601 * ENTRY (HL) = (ABUSS)
      602 * (DE) = ADDRESS OF REGIND
      603 *
      604 NEXT INX H SAE IF MEMORY, STORE ABUSS AND EXIT
      001.132 043          605 JZ SAE
      001.133 312 063 001          606 IS REGISTER MODE.
      607
      040.005          608 * SET REGI
      610 * SET REGI
      001.136 032          611 LDAX D (A) = REGI
      001.137 306 002          612 ADI 2 INCREMENT REG INNEX
      001.141 022          613 STAX D WRAP TO *SF*X
      001.142 376 014          614 CPI 12
      001.144 330          615 RC 12 IF NOT TOO LARGE, EXIT
      001.145 257          616 XRA A OVERFLOW
      001.146 022          617 STAX D
      001.147 311          618 ABORT RET

```

PAM/B - HB FRONT PANEL MONITOR #01.00.00,
MONITOR TASK SUBROUTINES.

HEATH X8ASM V1.1 06/21/77
15:44:16 01-APR-77 PAGE 16

620 ** LAST - DECREMENT DISPLAY ELEMENT.

621 * ENTRY (HL) = (ABUSS)

622 * (DE) = ADDRESS OF REGND
(DE) = ADDRESS OF REGND

623 *

624 LAST LDX H
625 JZ SAE IF MEMORY, STORE AND EXIT

626 IS REGISTER MODE.

627 JZ SAE

628 *

629 SET REGI
LDA D
SUI 2
STAX D

630 SET REGI
LDA D
SUI 2

631 LST2 STAX D
SUI 2

632 STAX D

633 RNC A,10
IF OK

634 HLT UNDERFLOW TO *FCX

635 STAX D

636 RET

637 RET

638 RET

640 ** MEMM - ENTER DISPLAY MEMORY MODE.

641 * ENTRY (BC) = ADDRESS OF INSFMOD

642 *

643 MEMM XRA A
SET DSFMOD (A) = 0

644 XRA A
SET DSFMOD

645 . SET DSFMOD

646 STAX B SET DISPLAY MEMORY MODE

647 DCX B

648 STAX B

649 LXI H,ABUSS+1

650 JMP IOA INPUT OCTAL ADDRESS

653 ** IN - INPUT DATA BYTE.

654 *

655 ** OUT - OUTPUT DATA BYTE.

656 * ENTRY (HL) = (ABUSS)

657 *

658 *

659 IN HLT IN

660 HLT IN

661 DB MI,LXID

662 OUT B,MI,OUT SKIP NEXT INSTRUCTION

663 MOV AH (A) = VALUE

664 MOV HL (H) = PORT

665 MOV LB (L) = IN/OUT INSTRUCTION

666 SHLD IOWRK

667 CALL IOWRK

668 MOV LH (L) = PORT

669 MOV HA (H) = VALUE

670 JMP SAE STORE ABUSS AND EXIT

PAM/8 - H8 FRONT PANEL MONITOR #01.00.00,
GO AND *STEP* FUNCTIONS

HEATH X8ASM V1.0 02/18/77

13:23:43 01-AFR-77 PAGE 18

```

675 ** GO - RETURN TO USER MODE
676 *
677 * ENTRY...NONE.
678 ROUTINE IS IN WASTE SPACE
001.222..303.063.000 679. 60...JMP...GO.

```

```

681 ** SSTEP - SINGLE STEP INSTRUCTION.
682 * ENTRY...NONE.
683 * ENTRY...NONE.
684 * ENTRY...NONE.
001.225..363.000 685 SSTEP EQU *
001.225..072..011.040 686 DI SINGLE STEP
001.226..356.020 687 LDA CTLFLG DISABLE INTERRUPTS UNTIL THE RIGHT TIME
001.231..356.020 688 XRI CB.SSI
001.233..323.360 689 OUT OF.CTL
001.235..062.011.040 690 STI CTLFLG CLEAR SINGLE STEP INHIBIT
001.240..341.000 691 FOF H PRIME SINGLE STEP INTERRUPT
001.241..303.172.000 692 JMP INTIXT SET NEW FLAG VALUES
001.241..303.172.000 693 H CLEAN STACK
001.241..303.172.000 694 RETURN TO USER ROUTINE FOR STEP

```

```

694 ** STPRTN - SINGLE STEP RETURN.
695 * ENTRY...NONE.
001.244..366.020 696 STPRTN EQU *
001.244..697 ORI CB.SSI DISABLE SINGLE STEP INTERRUPTION
001.246..323.360 698 OUT OP.CTL
040.011..699 SET CTLFLG TURN OFF SINGLE STEP ENABLE
001.250..022..700 STAX D
001.251..346.040..701 ANI CB.MTL SEE IF IN MONITOR MODE
001.253..302.344.000..702 JNZ MTR
001.256..303.042.040..703 JMP DIVVEC+3 TRANSFER TO USER'S ROUTINE

```

```

705 ** RMEM - LOAD MEMORY FROM TAPE.
706 *
707 RMEM LXI H,TABT
001.261..041.244.002..708 RMEM SHLD TERRY
001.264..042.031.040..709 * JMP LOAD SETUP ERROR EXIT ADDRESS

```

PAN/8 - H8 FRONT PANEL MONITOR \$01.00.00,
LOAD = LOAD MEMORY FROM TAPE

HÉATH XBASH V1.1 06/21/77
15:44:12 01-AFR-77 PAGE 12

```

712 *** LOAD - LOAD MEMORY FROM TAPE.
713 * READ THE NEXT RECORD FROM THE CASSETTE TAPE.
714 *
715 * USE THE LOAD ADDRESS IN THE TAPE RECORD.
716 *
717 * ENTRY (HL) = ERROR EXIT ADDRESS.
718 * EXIT (HL) = ERROR EXIT (IN STACK) SET TO ENTRY ADDRESS.
719 * USER F-REG (IN STACK) SET TO ENTRY ADDRESS.
720 * TO CALLER IF ALL OK.
721 * TO ERROR EXIT IF TAPE ERRORS DETECTED.
722

001.267
001.267 001.000.376 725 LXI R,1000A-RT,M,*256-256 (BC) = REQUIRED TYPE AND #
001.272 315.265 002 726 LDAO SRS SCAN FOR COUNT START
001.275 157 727 MOV L,A (HL) = COUNT
001.276 353 728 XCHG (DE) = COUNT, (HL) = TYPE AND #
001.277 015 729 DCR C (C) = - NEXT #
001.300 011 730 DAD B
001.301 174 731 MOV A,H
001.302 305 732 PUSH B SAVE TYPE AND #
001.303 365 733 PUSH FSW SAVE TYPE CODE
001.304 346 177 734 ANI 1770 CLEAR END FLAG BIT
001.306 265 735 ORA L SEQUENCE ERROR
001.307 076 002 736 MOVI A,2 IF NOT RIGHT TYPE OR SEQUENCE
001.311 302.205.002 737 JNE TERR READ ADDR
001.314 315.325 002 738 CALL RNF
001.317 104 739 MOVI B,H (BC) = P-REG ADDRESS
001.320 117 740 MOVI C,A
001.321 076.012 741 MOVI A,10
001.323 326 742 PUSH D SAVE (DE)
001.324 315.052.003 743 CALL LRA
001.327 321 744 POP D LOCATE REG ADDRESS
001.330 161 745 MOV H,C RESTORE (DE)
001.331 043 746 INX H SET P-REG IN MEM
001.332 160 747 MOV M,B
001.333 315.325 002 748 CALL RNF READ ADDRESS
001.336 157 749 MOVI L,A (HL) = ADDRESS, (DE) = COUNT
001.337 042 000 040 750 SHLD START
001.342 315.331 002 751 CALL RN8 READ BYTE
001.343 162 752 LDAI MOV M,A
001.346 042 024 040 753 SHLD ABUS SET ABUS FOR DISPLAY
001.351 043 754 INX H
001.352 033 755 MOVI D
001.353 172 756 INC H
001.354 263 758 MOVI A,D
001.355 302.342.001 759 ORA E,D
001.360 315.172.002 760 JNZ LOA1 IF MORE TO GO
001.363 361 761 CALL STC CHECK TAPE CHECKSUM
001.364 301 762 READ NEXT BLOCK
001.365 007 763 * READ NEXT BLOCK
001.363 361 764 POP PSW (A) = FILE TYPE BYTE
001.364 301 765 POP B (BC) = -(LAST TYPE, LAST #)
001.365 007 766 POP B RLC

```

PAM/8 - H8 FRONT PANEL MONITOR #01.00.00.
LOAD = LOAD MEMORY FROM TAPE.

HEATH X686H VI.1 06/21/77
15:44:21 01-APR-77 PAGE... 20.

001.366 332 133 002 768 JC TFT
001.371..393.272.001..769 JMP LOAD
..... ALL DONE - TURN OFF TAPE
..... READ ANOTHER RECORD.

PAM/B - H8 FRONT PANEL MONITOR #01.00.00,
DUMP = DUMP MEMORY TO MAG/PAPER TAPE

HEATH X8ASM V1.0 02/18/77
13:23:47 01-APR-77 PAGE 21

```

772 *** DUMP .. DUMP..MEMORY..T9..MAG..TAPE,
773 * DUMP..SPECIFIED..MEMORY..RANGE..TQ..MAG..TAPE,
774 * DUMP..SPECIFIED..MEMORY..RANGE..TQ..MAG..TAPE,
775 *
776 * ENTRY (START) .. START ADDRESS,
777 * (AUSS) .. END ADDRESS
778 * USER FC = ENTRY POINT ADDRESS
779 * EXIT TO CALLER,
780
781
001.374 041 244 002 782 WMEM EQU * H,1PABT
001.374 041 244 002 783 LXI H,A,TERR
001.377 042 031 040 784 SHLD TERRX
001.377 042 031 040 785 SHLD TERRX
SEJUE..ERROR..EXIT

002.002 076 001 786 DUMP MVI * H,1UCL,IE
002.004 323 371 787 OUT OF.TPC
002.006 076 026 788 MVI A,A,SYN
002.010 046 040 789 MVI H,32
002.012 046 040 790 WME1 CALL WNB
002.015 315 024 003 790 WME1 CALL DCR H
002.015 045 045 791 DCR H
002.016 302 012 002 792 JNZ WME1, WRITE..SYN..HEADER
002.021 076 002 793 MVI A,A,STX
002.023 315 024 003 794 CALL WNB
002.026 154 054 795 MOV L,H
002.027 042 027 040 796 SHLD CRCSUM
002.032 041 001 201 797 LXI H,R,T,MI+80H*256+1 FIRST AND LAST MI RECORD
002.035 315 017 003 798 CALL WNF
002.040 052 000 040 799 LHLD START
002.043 353 000 800 XCHG (DE,E),= START..ADDRESS,
002.044 052 024 040 801 LHLD ABUS (HL)= STOP..ADDR
002.047 043 000 802 INX (HL)= COMPUTE..WITH..STOP+1.
002.051 223 000 803 SUB E,L
002.052 157 005 805 MOV L,A
002.053 174 006 MOV A,H
002.054 232 807 SBB (HL)= COUNT
002.055 147 808 MOV H,A
002.056 315 017 003 809 CALL WNF
002.061 345 810 PUSH H
002.062 076 012 811 PWVI A,10
002.064 325 812 PUSH D
002.065 315 052 003 813 CALL LR,A
002.070 176 814 MOV A,H LOCATE P-REG ADDRESS
002.071 043 815 INX
002.072 146 816 MOV H,M
002.073 157 817 MOV L,A (HL)= CONTENTS OF PC
002.074 315 017 003 818 CALL WNF
002.077 341 819 POP H
002.100 321 820 POP D
002.101 315 017 003 821 CALL WNF
002.104 176 822 MOV A,H
002.105 315 024 003 823 WME2 CALL WNB
002.110 042 024 040 824 SHLD ABUS WRITE..BYTE
002.113 043 825 POP H SET ADDRESS FOR DISPLAY
002.114 033 826 INX H
002.114 033 827 DEX D

```

PAM/8 - HB FRONT PANEL MONITOR \$01.00.00,
 DUMP - DUMP MEMORY TO MAG/PAPER TAPE

```

HEATH XBASH V1.0 02/18/77
13:23:49 01-AFR-77 PAGE 22

002.115 172      828      MOV     A,D
002.116 263      829      ORA     E
002.117 302 104.002. 830      JNZ     WME2
002.117 302 104.002. 831      *      IF MORE TO GO

002.122 052 027.040 833      *      WRITE CHECKSUM
002.125 315 017.003 834      LHD     CRCSUM
002.130 315 017.003 835      CALL    WNF     WRITE IT
002.130 315 017.003 836      CALL    WNF     FLUSH CHECKSUM
002.130 315 017.003 837      *      JMP    TFT

839  **      TFT - TURN OFF TAPE.
840  *      STOP THE TAPE TRANSPORT.
841  *      *
842  *      *

002.133 257      843      XRA     A
002.134 323 371    844  TFT    OUT    OF,TFC   TURN OFF TAPE
002.134 323 371    845      *      *

847  **      HORN - MAKE NOISE.
848  *      ENTRY (A) = (MILLISECOND COUNT)/2
849  *      EXIT  NONE
850  *      USES   A,F
851  *      *
852      853      XTHL   XTHL
002.136 076 144    854  ALARM  MUL   A,200/2   200 MS BEEP
002.140 365      855  HORN   PUSH  FSW
002.141 076.200    856  *      MUL   A,CB,SPK
002.143 343      857  *      TURN ON SPEAKER
002.144 325      858  HRNO   XTHL
002.145 353      859  *      PUSH  D
002.146 041 011 040 860  XCHG
002.151 256      861  *      LXI   H,CTLFLG
002.152 136      862  XRA   H
002.153 167      863  MOU   M
002.154 056 033    864  MOV   M,A
002.156 172      865  HVI   L,*TICOUNT
002.157 206      866  *      (E) = OLD CTLFLG VALUE
002.160 276      867  MOV   A,D
002.161 302 160 002 868  ADD   N
002.164 056 011    869  HRN2  CMF   N
002.166 163      870  JNE   HRN2
002.167 321      871  MVI   L,*CTLFLG
002.170 341      872  MOV   H,E
002.171 311      873  POP   D
002.171 311      874  POF   H
002.171 311      875  RET

```

PAN/B - HB FRONT PANEL MONITOR #01.00.00,
TAPE PROCESSING SUBROUTINES

HEATH X8ASM V1.1 06/21/77
15:44:25 01-APR-77 PAGE 23

```

880 ** CTC - VERIFY CHECKSUM.
881 * ENTRY TAPE JUST BEFORE CRC.
882 * EXIT TO CALLER IF OK.
883 * USES TO *TPERR* IF BAD.
884 * A,F,H,L

885 * USES READ NEXT FAIR.
886 * CALL RNF.
887 * CALL RNF.
888 * CALL RNF.
889 * CALL RNF.
890 * MOV A,H.
891 * ORA L.
892 * RJZ RETURN OF OK.
893 * MVI A,1.
894 * JNE .TPERR. (B) = CODE.
895 * TERR - PROCESS TAPE ERROR.
896 * DISPLAY ERR NUMBER IN LOW BYTE OF ARUSS.
897 * IF ERROR NUMBER EVEN, DONT ALLOW *.
898 * IF ERROR NUMBER ODD, ALLOW *.
899 * ENTRY (A) = NUMBER.
900 * IS *, RETURN (IF PARITY ERROR).
901 * CALL TFF.
902 * CALL TFF.
903 * ENTRY (A) = NUMBER.
904 * CALL TFF.
905 * MOV B,A. (B) = CODE.
906 * STAA ARUSS.
907 * CALL TFF.
908 * CALL TFF.
909 * TURN OFF TAPE.
910 * IS *, RETURN (IF PARITY ERROR).
911 * DB MIANI FALL THROUGH WITH CARRY CLEAR.
912 * MOV A,B.
913 * TFR.
914 * CALL TFF.
915 * RRC.
916 * RC.
917 * CALL TFF.
918 * BEEP AND FLASH ERROR NUMBER.
919 * ALARM ALARM IF PROPER TIME.
920 * CALL TXIT SEE IF *.
921 * IN IP,PA0.
922 * CPI 00101111B.
923 * JE TER3.
924 * LDAA TICCNT+1.
925 * RAR IF *.
926 * SET IF 1/2 SECOND.
927 * JMP TER1.
928 * CALL TFF.
929 * IN IP,PA0.
930 * CPI 00101111B.
931 * JE TER3.
932 * LDAA TICCNT+1.
933 * RAR IF *.
934 * SET IF 1/2 SECOND.
935 * JMP TER1.

```

PAM/B - HB FRONT PANEL MONITOR #01:00.00:
TAPE PROCESSING SUBROUTINES

HEATH X8ASM V1.0 02/18/77
13:23:52 01-APR-77 PAGE 24

929 ** TPART - ABORT TAPE LOAD OR DUMP.
930 * ENTERED WHEN LOADING OR DUMPING, AND THE '*' KEY
931 * IS STRUCK.

002,244 257 934 TPABT XRA A
002,245 323 371 935 TPABT OUT OP,TPC OFF TAPE
002,247 303 322 000 936 JMF ERROR

939 ** TXIT - CHECK FOR USER FORCED EXIT.
940 * TXIT CHECKS FOR AN '*' KEYED ENTRY. IF SO, TAKE
941 * THE TAPE DRIVER ABNORMAL EXIT.
942 *
943 *
944 * ENTRY NONE
945 * EXIT TO *RET* IF NOT '*',
946 * (A) = FORT STATUS
947 * USES A,F
948 * USES A,F
949 *

002,252 333 360 950 IF,FAU
002,254 376 157 951 TXIT IN IF,FAU
002,256 333 371 952 CPI 0110111B
002,260 300 953 IN * REAN TAPE STATUS
002,261 052 031 040 954 RNE NOT '*', RETURN WITH STATUS
002,264 351 955 LHLD TPERRX
002,264 351 956 FCHL ENTER (TPERRX)

958 ** SRS - SCAN RECORD START
959 *
960 * SRIS READS BYTES UNTIL IT RECOGNIZES THE START OF A RECORD.
961 *
962 * THIS REQUIRES
963 * AT LEAST 10 SYNC CHARACTERS
964 * 1 STX CHARACTER.
965 *
966 * THE CRC-16 IS THEN INITIALIZED.

967 *
968 * ENTRY NONE
969 * EXIT TAPE POSITIONED (AND MOVING). CRCSUM = 0.
970 * (DE) = HEADER BYTES
971 * (HA) = RECORD COUNT
972 * USES A,F,D,E,H,L
973 *
974 *
002,265 026 000 975 SRS EQU *
002,267 142 976 SRS1 MVI D,O
002,270 152 977 MOV H,D
002,270 152 978 MOV L,D
 (HL) = 0

PAM/B - HB FRONT PANEL MONITOR #01.00.00,
TAPE PROCESSING SUBROUTINES

HEATH X86ASM V1.0 02/18/77
13:23:54 01-AFR-77 PAGE 25

```

002,271 315,331,002 979 SRS2 CALL RNB ..... READ.NEXT.BYTE
002,274 024 980 INR D ..... READ.NEXT.BYTE
002,275 376,024 981 CPI A,SYN ..... HAVE.SYN
002,277 312,271,002 982 JE SRS2 ..... NOT.STX
002,302 376,002 983 CPI A,STX ..... NOT.STX - START.OVER
002,304 302,265,002 984 JNE SRS1 ..... NOT.STX
002,307 076,012 985 HUI A,10 ..... SEE.IF.ENOUGH.SYN.CHARACTERS...
002,311 272,265,002 986 CMF D ..... NOT.ENOUGH
002,312 322,027,040 987 SHLD SRS1 ..... CLEAR.CRC-16
002,315 042,027,040 988 CRCSUM ..... READ.LEADER
002,320 315,325,002 990 CALL RNF ..... CALL.RNF
002,323 124 991 MOV D,H ..... MOV.D,H
002,324 137 992 MOV E,A ..... MOV.E,A
002,325 147 993 * ..... JME RNF ..... READ.COUNT
002,330 147 994 * ..... JME RNF ..... READ.COUNT
002,331 076,064 1018 RNB HUI A,UCI,RU+UCI,ER+UCI,RE TURN.ON.PEADER.FOR.NEXT.BYTE
002,333 323,371 1019 OUT OF.TPC
002,335 315,252,002 1020 RNB1 CALL TXIT ..... CHECK.FOR *, READ STATUS
002,340 346,002 1021 ANI USR,RXR ..... RSB1
002,342 312,335,002 1022 JZ IF NOT READY
002,345 333,370 1023 TN INPUT.DATA ..... TPA,SPD
002,346 333,370 1024 * ..... JMP CRC CHECKSUM

```

PAM/8 - H8 FRONT PANEL MONITOR #01.00.00,
TAPE PROCESSING SUBROUTINES

HEATH X86ASM V1.0 02/18/77
13:23:56 01-APR-77 PAGE 26

```

1026 ** CRC = COMPUTE_CRC-16.
1027 * CRC COMPUTES A CRC-16 CHECKSUM FROM THE POLYNOMIAL.
1028 * (X + 1) * (X^15 + X + 1).
1029 *
1030 * SINCE THE CHECKSUM GENERATED IS A DIVISION REMAINDER,
1031 * A CHECKSUMMED DATA SEQUENCE CAN BE VERIFIED BY RUNNING
1032 * THE DATA THROUGH CRC, AND THEN RUNNING THE PREVIOUSLY OBTAINED
1033 * CHECKSUM THROUGH CRC. THE RESULTANT CHECKSUM SHOULD BE 0.
1034 *
1035 *
1036 * ENTRY (CRCSUM) = CURRENT CHECKSUM
1037 *
1038 * (A) = BYTE
1039 * EXIT (CRCSUM) UPDATED
1040 * (A), UNCHANGED.
1041 * USES F
1042 *
1043
002.347 305
002.350 006 010
002.352 345
002.353 052 027 046
002.356 007
002.357 117
002.360 175
002.361 207
002.362 157
002.363 174
002.364 027
002.365 147
002.366 027
002.367 251
002.370 012
002.371 322 004 003
002.374 174
002.375 356 260
002.377 147
003.000 175
003.001 356 005
003.003 157
003.004 171
003.005 005
003.006 302 356 002
003.011 042 027 040
003.014 341
003.015 301
003.016 311
1044 CRC
1045
1046
1047
1048 CRC1
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066 CRC2
1067
1068
1069
1070
1071
1072

PUSH B
MOV B,B
PUSH H
MOV H,H
LHD CRCSUM
MOV C,A
MOV A,L
MOV A,L
ADD A,A
MOV L,A
MOV A,H
MOV H,H
RAL H,A
RAL H,A
XRA C
XRA C
RCR CRC2
JNC CRC2
MOV A,H
XRI 200Q
MOV H,A
MOV A,L
XRI 50
MOV L,A
MOV A,C
DCR B
JNZ CRC1
SHLD CRCSUM
POP H
POP B
RESTORE HL
RESTORE BC
RET
EXIT.

```

FAM/8 - H8 FRONT PANEL MONITOR #01.00.00
TAPE PROCESSING SUBROUTINES

HEATH X8ASM V1.0 02/18/77
13:23:58 01-AFR-77 PAGE 27

```

1074 ** WNF - WRITE NEXT FAIR.
1075 * WFI. WRITES THE NEXT TWO BYTES TO THE CASSETTE DRIVE.
1076 * ENTRY (H,L) = BYTES.
1077 * EXIT   WRITTEN.
1078 * USES A,F.
1079 * A,F.
1080 * A,F.
1081

003.017 174 1082 WNF    MOV    A,H
003.020 315 024.003 1084 CALL   WNB   A,L
003.023 175 1085 MOV    WNB   A,L
1086 * JMP    WNB   WRITE NEXT BYTE.

```

```

1088 ** WNB - WRITE BYTE
1089 * WNB - WRITE BYTE
1090 * WNB WRITES THE NEXT BYTE TO THE CASSETTE TAPE.
1091 * ENTRY (A) = BYTE
1092 * EXIT   NONE.
1093 * USES F
1094 * F
1095

1096
003.024 365 1097 WNB   PUSH   FSU   CHECK FOR *, READ STATUS
003.025 315 252 002 1098 WNB1  CALL   TEXIT
003.030 346 001 1099 ANI   USR,TXR
003.032 312 025 003 1100 JZ    WNB1
003.035 076 021 1101 MVI   A,UCL,ERHUCI,IE,ENABLE TRANSMITTER
003.037 323 371 1102 OUT   OF,TFC
003.041 361 1103 POP   FSU
003.042 323 370 1104 OUT   OF,TFO
003.044 303 347 002 1105 CRC   COMPUTE CRC.

```

FAM/8 - H8 FRONT PANEL MONITOR #01.00.00,
SUBROUTINES

HEATH X8ASM V1.0 02/18/77
13:23:59 01-AFR-77 PAGE 28

```

1109 ** LRA - LOCATE REGISTER ADDRESS.
1110 * ENTRY NONE.
1111 * EXIT (A) = REGISTER INDEX
1112 * (H,L) = STORAGE ADDRESS
1113 * (H,E) = (G,A)
1114 * USES A,D,E,H,L,F
1115 *
1116 *
1117 *
1118 *
003.047 072.005 040 1119 LRA LDA REGI
003.052 137 1120 LRA NOV E,A
003.053 026 000 1121 NOV D,O
003.055 052 035 046 1122 LHLD REGPTR
003.060 031 1123 DAD (DE) = (REGPTR)+(REGI)
003.061 311 1124 RET
1125 *
1126 ** IOA - INPUT OCTAL ADDRESS.
1127 * ENTRY (H,L) = ADDRESS OF RECEPTION DOUBLE BYTE.
1128 * EXIT TO *KEY* IF ERROR.
1129 * TO *RET*+1 IF OK. VALUE IN MEMORY.
1130 * USES A,D,E,H,L,F
1131 *
1132 *
1133 003.062 315.066 003 1134 IOA CALL IOB INPUT BYTE
003.065 053 1135 BCX H
1136 *
1137 ** IOB - INPUT OCTAL BYTE.
1138 * READ ONE OCTAL BYTE FROM THE KEYSET.
1139 * ENTRY (H,L) = ADDRESS OF BYTE TO HOLD VALUE
1140 * (C) = SET IF FIRST DIGIT IN '(A)'
1141 * EXIT TO *RET* IF ALL OK
1142 * USES Y0,*ERRK* IF ERROR
1143 *
1144 *
1145 * USES A,D,E,H,L,F
1146 *
1147 *
1148 003.066 026 003 1149 IOB MVI D,3 (D) = DIGIT COUNT
003.070 324 260 003 1150 IOB CNC RCK READ CONSOLE KEYSET
1149 *
1150 *
1151 *
1152 CPI 8
1153 JNC ERROR IF ILLEGAL DIGIT
1154 *
1155 MOV E,A (E) = VALUE
1156 MOV A,H
1157 RLC SHIFT 3
1158 RLC

```

PAM/B - HS FRONT PANEL MONITOR #01.00.00.
SUBROUTINES

HEATH X8ASM V1.0 02/18/77
13:24:01 01-APR-77 PAGE 29

```

003.104 007 1159 RLC
003.105 346 370 1160 ANI 3750
003.107 263 1161 ORA E
003.110 167 1162 MOV H,A RÉFLAGE
003.111 025 1163 FCR 0
003.112 362 070 003 1164 JNZ 0 IF NOT DONE
003.115 076 017 1165 MWI A,30/2 BEEP FOR 30 MS
003.117 303 140 002 1166 JMP HORN

1168 ** POP = DECODE FOR OCTAL DISPLAY.
1169 * ENTRY (H,L) = ADDRESS OF LED REFRESH AREA
1170 * (B) = *UR* PATTERN TO FORCE ON BARS OR PERIODS
1171 * EXIT (A) = OCTAL VALUE
1172 * (H,L) = NEX DIGIT ADDRESS
1173 * USES A,B,C,D,H,L
1174 *
1175

003.122 326 1176 PUSH D
003.123 026 003 1177 POP D,DATA/256
003.125 016 003 1178 MWI C,3 LEFT 3 PLACES
003.127 027 1179 MWI
003.128 027 1180 MWI
003.130 027 1181 MWI
003.131 027 1182 MWI
003.132 365 1183 MWI
003.133 346 907 1184 MWI
003.135 306 356 1185 MWI
003.137 132 1186 MWI
003.140 032 1187 LDAX D
003.141 250 1188 XRA B
003.142 348 177 1189 ANI 7
003.144 250 1190 XRA B
003.145 167 1191 MOV H,A SET IN MEMORY
003.146 043 1192 INX H
003.147 170 1193 MOV A,B
003.150 092 1194 RLC
003.151 107 1195 MOV B,A
003.152 361 1196 POP PSW (A) = VALUE
003.153 015 1197 DCR C
003.154 302 127 003 1198 DDP1 IF MORE TO GO
003.157 321 1199 POP D
003.160 311 1200 RET RETURN

```

PAM/8 - HB FRONT PANEL MONITOR #01.00.00,
UFD - UPDATE FRONT PANEL DISPLAYS.

HEATH XBASM V1.0 02/18/77
13:24:02 01-AFR-77 PAGE 30

```

1203 ** UFD - UPDATE FRONT PANEL DISPLAYS.
1204 *
1205 * UFD IS CALLED BY THE CLOCK INTERRUPT PROCESSOR WHEN IT IS
1206 * TIME TO UPDATE THE DISPLAY CONTENTS. CURRENTLY, THIS IS DONE
1207 * EVERY 32 INTERRUPTS, OR ABOUT 32 TIMES A SECOND.
1208 *
1209 *
1210 * ENTRY (H,L) = ADDRESS OF REFCT
1211 * EXIT NONE
1212 * USES ALL
1213

1214
003.161 076 002 1215 UFD EQU * MVI A,0.EDU
003.163 240 1216 ANA B
003.164 300 1217 RNC
1218 RNZ IF NOT TO HANDLE UPDATE

003.165 056 006 1219 MVI L,*DSFR0T
003.167 176 1220 MOV A,M
003.170 007 1222 RLC
003.171 167 1223 MOV M,A ROTATE PATTERN
003.172 107 1224 MOV B,A
003.173 043 1225 INX H
000.000 1226 ERRNZ DSFMOD-DSFR0T-1 (A) = DSFMOD
003.174 176 1227 MOV A,M
003.175 002 1228 ANI 2
003.177 052 024 040 1229 LHLD ABUS
003.202 312 227 003 1230 JZ UFD IF MEMORY
1231
1232 * AM DISPLAYING REGISTERS.
1233
003.205 315 047 003 1234 CALL LRA LOCATE REGISTER ADDRESS
003.210 345 1235 PUSH H,DSPA
003.211 041 342 003 1236 LXI D,D
003.214 031 1237 MOV A,M <H,L> = ADDRESS OF REG NAME PATTERNS
003.215 176 1238 INX H
003.216 043 1239 MOV H,H
003.217 146 1240 MOV L,A (H,L) = REG NAME PATTERN
003.220 157 1241 XTHL
003.221 343 1242 ORA H,
003.222 264 1243 MOU A,H CLEAR 'Z'
003.223 176 1244 INX H
003.224 043 1245 MOV H,H
003.225 146 1246 MOU L,A (HL) = ADDRESS OF REGISTER FAIR CONTENTS
003.226 157 1247
1248
1249 * SETUP DISPLAY
1250
003.227 365 1251 UFD1 PUSH FSU
003.230 353 1252 XCCH
003.231 041 013 040 1253 LXI H,ALEPS
003.234 172 1254 MOV A,D
003.235 315 142 003 1255 CALL D,D FORMAT.ABANS.HIGH.HALF
003.240 173 1256 MOV A,E
003.241 315 142 003 1257 CALL D,D FORMAT.ABANK.LOW.HALF
003.244 361 1258 FOF

```

PAM/B - HB FRONT PANEL MONITOR \$01.00.00.
UFD - UPDATE FRONT PANEL DISPLAYS.

HEATH XBASM V1.0 02/18/77
13:24:04 01-APR-77 PAGE 31

```
003,245 032 1259 LMAX P
003,246 312 122 003 1260 JZ D00 IF MEMORY, DECODE BYTE VALUE
003,247 1261 * IS REGISTER, SET REGISTER NAME.

003,251 066 377 1263 MVI N,3770 CLEAR DIGIT
003,253 341 1264 POF H
003,254 042 022 040 1265 SHLD DLENS+1
003,257 311 1266 RET
003,258 1267
```

HÁN/8 = H8 FRON'T PÁNEL MÓNITOR #61.00.00.
RCK = READ CONSOLE KEYFÁN.

HEATH X8ASM V1.1 06/21/77
15:44:39.91-APR-77 PAGE 32

```

1271 ** RCK - READ CONSOLE KEYPAD.
1272 * RCK IS CALLED TO READ A KEYSTROKE FROM THE CONSOLE KEYFÁN.
1273 * WHENEVER A KEY IS ACCEPTED.
1274 * RCK PERFORMS DÉBOUNCING, AND AUTO-REPEAT. A *BIP* IS SOUNDED.
1275 * WHEN A VALUE IS ACCEPTED.

1276 *
1277 *
1278 *
1279 * KEY PAD VALUES:
1280 * 1111 1110 = 0
1281 * 1111 1100 = 1
1282 * 1111 1010 = 2
1283 * 1111 1000 = 3
1284 * 1111 0110 = 4
1285 * 1111 0100 = 5
1286 * 1111 0010 = 6
1287 * 1111 0000 = 7
1288 * 1110 1111 = 8
1289 * 1110 1111 = 9
1290 * 1010 1111 = +
1291 * 1000 1111 = -
1292 * 0110 1111 = *
1293 * 0100 1111 = /
1294 * 0010 1111 = #
1295 * 0000 1111 = ,
1296 *
1297 * ENTRY NONE
1298 * EXIT TO CALLER WHEN A KEY IS HIT
1299 * (A) = 0 = 0'
1300 * 1 = 1'
1301 * 2 = 2'
1302 * 3 = 3'
1303 * 4 = 4'
1304 * 5 = 5'
1305 * 6 = 6'
1306 * 7 = 7'
1307 * 8 = 8'
1308 * 9 = 9'
1309 *
1310 * 10 = +
1311 * 11 = -
1312 * 12 = *
1313 * 13 = /
1314 * 14 = #
1315 * 15 = .
1316 * USES A,F
1317 *
1318 *
1319 RCK EQU *
1320 FUSH H
1321 FUSH B
1322 MOI C,400/20
1323 LXI H,RCKA WAIT 400 MS
1324 RCK1 IN IF.PAD INPUT PAD VALUE
1325 MOV B,A (B),F VALUE
1326 MOV B,A (B),F VALUE

```

PAM/S = HB FRONT PANEL MONITOR #01:00:00; HEATH X8ASM.V1:1 96/21/77
RCK - READ CONSOLE KEYPAD, 15:44:41 01-AFR-77 PAGE 33

```

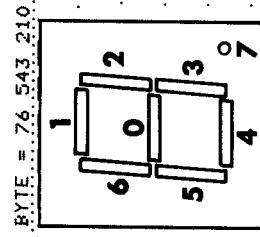
003:272 074 042 1327 MVI A,20/2
003:274 315 053 000 1328 CALL DLY
003:277 170 1329 MOV A,B
003:300 276 1330 CMP H
003:301 302 310 003 1331 JNE RCK2
003:304 015 1332 ICR C
003:305 302 267 003 1333 JNZ RCK1
003:306 1334 WAIT N CYCLES
003:307 * HAVE KEY VALUE
003:308 1336
003:310 167 1337 RCK2
003:311 356 376 1338 MOV M,A
003:313 017 1339 XRI 376Q
003:314 322 326 003 1340 RRC
003:317 017 1341 JNC RCN3
003:320 017 1342 RRC
003:321 042 1343 RRC
003:322 017 1344 RRC
003:323 322 267 003 1345 RRC
003:326 107 1346 RCK3
003:327 076 002 1347 MOV B,A
003:331 315 140 002 1348 MVI A,4/2
003:334 170 1349 CAL HORN
003:335 346 017 1350 MOV A,B
003:337 301 1351 ANI 17Q
003:340 341 1352 POP B
003:341 311 1353 RET H
003:342 1354 RETURN

```

PAM/8 - H8 FRONT PANEL MONITOR. #01.00.00.
SEGMENT PATTERNS AND CONSTANTS.

HEATH X-BASM V4.1 06/24/77
15:44:42 01-AFR-77 PAGE 34

1357 ** DISPLAY SEGMENT CODING:



BYTE = 76 543 210

1358 * DSFA DS 0

1359 * DSFA DS 0

1360 * DSFA DS 0

1361 * DSFA DS 0

1362 * DSFA DS 0

1363 * DSFA DS 0

1364 * DSFA DS 0

1365 * DSFA DS 0

1366 * DSFA DS 0

1370 ** REGISTER INDEX TO 7-SEGMENT PATTERN

003.342	244	230	1373	DSFA	DS 0
003.342	220	234	1374	DW	100100010100100B SF
003.344	206	215	1375	DW	1001110010010000B AF
003.346	302	214	1376	DW	1000110110000110B BC
003.350	222	217	1377	DW	1000110011000010B DE
003.352	230	316	1378	DW	100011110010010B HL
003.354	230	316	1378	DW	110011101001100B PC

1380 ** OCTAL TO 7-SEGMENT PATTERN

003.356	001	1383	DODA	DS 0
003.356	163	1384	DR	00000001B 0
003.357	110	1385	DR	01100011B 1
003.360	140	1386	DR	01001000B 2
003.361	062	1387	DR	01000000B 3
003.362	044	1388	DR	0010010B 4
003.364	004	1389	DR	00100100B 5
003.365	161	1390	DR	0000000B 6
003.366	000	1391	DR	0110001B 7
003.367	040	1392	DR	0000000B 8

1394 ** I/O ROUTINES TO BE COPIED INTO AND USED IN RAM.

1395 * MUST CONTINUE TO 377A FOR FFTER COPY.

1396 * THE TABLE MUST ALSO BE BACKWARDS TO THE FINAL RAM
1397 *

003.371.....1398.....1399.....1400.....ORG...4009A-7

003.371	001	1401	PRSRDM	EQU *	REFIND
003.372	000	1402	DB	1	CFLFLAG
003.373	000	1403	DB	0	,MFLAG

PAM/B - HB FRONT PANEL MONITOR #01.00.00,
CONSTANTS AND TABLES.

HEATH XBASM V1.1 06/21/77
15:44:44 01-APR-77 PAGE 35

003.374	000	1405	DB	0	DSPMOD
003.375	000	1406	DB	0	DSPPROT
003.376	012	1407	DB	10	REGI
003.377	311	1408	DB	MI.RET	
060.000		1409			
		1410			
			ERNZ *	-4000A	

FAM/B - HB FRONT PANEL MONITOR #01.00.00,
RAM CELLS

HEATH X8ASM V1.1 06/21/77
15:44:44 01-APR-77 PAGE 36

1413 ** THE FOLLOWING ARE CONTROL CELLS AND FLAGS USED BY THE KEYPAD
1414 * MONITOR.

040.000..... 1415 ORG DS 40000A 8192
040.000..... 1417 START DS 2 DUMP STARTING ADDRESS
040.002..... 1418 DOWNK DS 2 IN OR OUT INSTRUCTION
040.002..... 1419 FRSRAM EQU * FOLLOWING CELLS INITIALIZED FROM ROM
040.004..... 1420 EQU DS 1
040.004..... 1421 DS 1 RET

040.005..... 1422 REGI DS 1 INDEX OF REGISTER UNDER DISPLAY
040.006..... 1423 DS 1 PERIOD FLAG BYTE
040.007..... 1424 DS 1 DISPLAY MODE

040.910..... 1425 DS 1
040.910..... 1426 DS 1
040.910..... 1427 MELAG DS 1
040.910..... 1428 * SEE #0.***** BITS DESCRIBED AT FRONT

040.011..... 1429 CTIFLG DS 1 FRONT PANEL CONTROL BITS
040.012..... 1430 REFND DS 1
000.007..... 1431 DS 1 REFRESH INDEX (0 TO 7)
040.013..... 1432 PRSL EQU * FRSRAM END OF AREA INITIALIZED FROM ROM

040.013..... 1433 DS 1
040.013..... 1434 FFLEIS EQU * FRONT PANEL LED PATTERNS
040.013..... 1435 ALEIS DS 1
040.014..... 1436 DS 1
040.015..... 1437 DS 1
040.015..... 1438 DS 1

040.016..... 1439 DS 1 ADDR 0
040.017..... 1440 DS 1 ADDR 4
040.020..... 1441 DS 1 ADDR 5
040.021..... 1442 DS 1
040.021..... 1443 DS 1 DATA 0
040.022..... 1444 DS 1 DATA 1
040.023..... 1445 DS 1 DATA 2
040.024..... 1446 DS 1

040.024..... 1447 ABUS DS 2 ADDRESS BUS
040.026..... 1448 RCKA DS 1 RCK SAVE AREA
040.027..... 1449 CRCSUM DS 2 CRC-16 CHECKSUM
040.031..... 1450 TFLRXX DS 2 TAPE ERROR EXIT ADDRESS
040.033..... 1451 TICKCT DS 2 CLOCK TICK COUNTER
040.035..... 1452 DS 2
040.035..... 1453 REGTR DS 2 REGISTS CONTENTS POINTER

040.037..... 1454 DS 0 USER INTERRUPT VECTORS
040.037..... 1455 UVEC DS 0
040.042..... 1456 DS 3 JUMP TO CLOCK PROCESSOR
040.042..... 1457 DS 3 JUMP TO SINGLE STEP PROCESSOR
040.045..... 1458 DS 3 JUMP TO I/O 3
040.050..... 1459 DS 3 JUMP TO I/O 4
040.053..... 1460 DS 3 JUMP TO I/O 5
040.056..... 1461 DS 3 JUMP TO I/O 6
040.061..... 1462 DS 3 JUMP TO I/O 7

040.064..... 1463 DS 1
040.064..... 1464 DS 1 ENH
ASSEMBLY COMPLETE
1464 STATEMENTS
0. ERRORS DETECTED
22310 BYTES FREE

CROSS REFERENCE TABLE.

XREF V1.0 PAGE 37

CROSS REFERENCE TABLE.

	XREF	V1.0	PAGE	38
IOA	003062	565	651	1134L
IOB	003066	545	1134	1149L
IOP1	003070	1150L	1164	
IOWRK	040002	666	667	1419L
IP.FAD	000360	81E	439	922
IP.TPC	000371	85E	953	954 1325
IP.TPD	000370	87E		1023
LAST	001150	528	625L	
LOAD	001272	728L	769	
LOAD	001343	752L	759	
LOAD	001267	724E		
LRA	003047	586	1119L	1234
LRA.	003052	427	743	813 1120L
LST2	001154	631L		
MEMM	001155	531	644L	
MI.ANI	000346	128E	912	
MI.HLT	000166	123E	433	
MI.IN	000333	125E	660	
MI.LDA	000072	127E		
MI.LXD	000021	129E	661	
MI.OUT	000323	126E	662	
MI.RET	000311	124E	1408	
MTR	000344	476E	702	
MTR1	000345	479	479L	
MTR4	001005	492	502L	
MTR5	001051	497	541L	
MTR6	001067	543	559L	
MTRA	001035	566	520E	
NEXT	001132	527	604L	
UP.CYL	000360	83E	689	698
OP.DIG	000360	83E	400	
OP.SEG	000361	84E	402	
OP.TPC	000371	86E	304	787
OP.TPD	000370	88E	1104	
OUT	001202	523	662L	
PRSL	000007	191	1432E	
PRSRAM	040004	191	1420E	1432
PRSRDM	003371	190	1401E	
R\$W	001126	530	595L	
RCK	003260	489	580	1150 1319E
RCK1	003247	1325L	1333	1345
RCK2	003310	1331L	1337L	
RCK3	003326	1340	1346L	
RCKA	040026	1323	1431L	
REFIND	040012	391	1453L	
REGI	04005	512	586	609 630 1119 1423L
REGM	001104	532	573L	
REGPTR	040035	535	467	1122 1453L
RMEM	001261	525	708L	
RNB	002331	752	979	1004 1018L
RNB1	002335	1020L	1022	
RNP	002325	738	748	BBB 990 1004L
RT.BE	000002	113E		
RT.CT	000003	114E		
RT.MI	000001	112E	725	797
SAE	001063	554L	605	626 670
SAVALL	000132	200	319L	
SINCR	004000	281E	283	284

CROSS REFERENCE TABLE.

	XREF V1.0	PAGE	39
SRS.....	002265.....	726.....	975E.....
SRS1.....	002265.....	976L.....	984.....
SRS2.....	002271.....	979L.....	982.....
SST1.....	001235.....	257.....	690L.....
SSTEP.....	001225.....	524.....	685E.....
START.....	040000.....	284.....	750.....
SPRTN.....	001244.....	219.....	696E.....
TER1.....	002220.....	920L.....	927.....
TER3.....	002215.....	913L.....	924.....
TFT.....	002133.....	768.....	844L.....
TICNT.....	040033.....	366.....	371.....
TFBT.....	002244.....	708.....	783.....
TPERR.....	002295.....	737.....	206L.....
TPERRX.....	040031.....	709.....	784.....
TPXT.....	002252.....	321.....	254L.....
UCI_ER.....	000020.....	165E.....	1020.....
UCI_IE.....	000002.....	167E.....	1098.....
UCI_IR.....	000100.....	163E.....	1101.....
UCI_RU.....	00004.....	164E.....	1018.....
UCI_RO.....	000040.....	164E.....	1018.....
UCI_TE.....	000001.....	168E.....	786.....
UF0.....	003161.....	469.....	1215E.....
UF01.....	003227.....	1230.....	1251L.....
UVVEC.....	040037.....	229.....	234.....
UMI_16X.....	000002.....	158E.....	303.....
UMI_1K.....	000100.....	148E.....	303.....
UMI_1X.....	000001.....	157E.....	
UMI_2B.....	000300.....	150E.....	
UMI_64X.....	000003.....	159E.....	
UMI_HB.....	000200.....	149E.....	
UMI_L5.....	000000.....	153E.....	
UMI_L6.....	000004.....	154E.....	
UMI_L7.....	000010.....	155E.....	
UMI_L8.....	000014.....	156E.....	303.....
UMI_PA.....	000020.....	152E.....	
UMI_PE.....	000040.....	151E.....	
UO_CLN.....	000001.....	138E.....	346.....
UO_DOU.....	000002.....	137E.....	461.....
UO_HLT.....	0000200.....	135E.....	420.....
UO_NFR.....	000100.....	136E.....	384.....
USR_FE.....	000040.....	172E.....	461.....
USR_OE.....	000020.....	173E.....	
USR_PE.....	000010.....	174E.....	
USR_RXR.....	000002.....	176E.....	1021.....
USR_TXE.....	000004.....	175E.....	
USR_TXR.....	000001.....	177E.....	1099.....
WME1.....	002012.....	790L.....	792.....
WME2.....	002104.....	823L.....	830.....
WMEM.....	001374.....	524.....	782E.....
WNB.....	003024.....	790.....	794.....
WNB1.....	003025.....	1098L.....	1100.....
WNP.....	003017.....	798.....	809.....
		818.....	821.....
		835.....	836.....
		1083L.....	

25434 BYTES FREE

APPENDIX B

Demo: PAM-8

This program shows the advanced features of PAM-8 and, as such, should not be evaluated as either an efficient or useful routine. The program uses the H8 clock, keyboard, display and interrupt capabilities to create an accurate interval timer that lets you enter an integer value from zero through nine seconds. When the program has counted down to zero, an audio alert is sounded, ending the program and returning control to PAM-8.

Use the H8 keypad to enter the machine code, set the program counter, and execute the program. While the program is being executed, the front panel display will be turned off and the computer will wait for you to enter a digit from the keypad. A single digit corresponding to the integer you selected is displayed and decremented until control is returned to PAM-8.

The timer is typical of a program you might create. An interval timer, a clock, or even a game requires that you communicate with the H8. The keypad lets you communicate with the CPU, and the CPU uses the LED display to communicate with you. The computer understands the selected time interval when you press a decimal key on the front panel. The job status, or decremented time interval, is relayed to you by the front panel displays. This interaction between you and the machine is characteristic of most software applications.

The program uses the PAM-8 firmware. Although it appears simple enough, you must study both the program and the PAM-8 listing ("Appendix A") in order to understand what happens when the program is operating. We suggest that you take a course in assembly language programming, such as the Heath EC-1108, if you have difficulty understanding the program.

The program source listing was prepared on an H8 computer system using the text editor (TED-8) and the assembler (HASL-8). NOTE: Your programs can be handwritten and assembled if you have only an H8.

The Sample Program

This program initially blanks the LED display and waits for you to enter an integer value. The computer verifies that the value you selected is permissible and then increments and stores the integer. The value was incremented because the display routine always decrements the count by one when it is called.

The most subtle part of this program is the interrupt service routine.* The H8 requires that you initialize the interrupt service routine by loading an instruction and address into the user interrupt vector (UIVEC) before executing the interrupt. After UIVEC is initialized, the program will jump to the service routine after the next interrupt signal is generated.

The main body of the program is a “do-nothing” loop that holds the program in a wait status until the interval timer has reached zero. You could replace the loop with another program which would execute simultaneously with the clock counter. When the countdown is complete, the program returns the H8 computer to its original status before halting.

*NOTE: Basically, an interrupt is a CPU response to a control signal. This signal directs the software to automatically save the current CPU status and transfers program control to a specified routine, called an interrupt handler. When the interrupt handler completes the routine, program control returns to its original status and normal program execution continues.

```

HEATH.ASM #104, 01.00.
PAGE 1

*** **** DEMO: PAMB ****
*** SYSTEM DEFINITIONS ***
*** ****

040.100
    ORG     40100A
    ERROR   EQU     322A    RESET PAMB
    HORN    EQU     2140A    MAKE NOISE
    RCK     EQU     32260A    READ CONSOLE KEYPAD
    DODA   EQU     3356A    OCTAL TO 7-SEGMENT PATTERN
    MELAG  EQU     40010A    USER FLAG OPTIONS
    FPLEDS EQU     40013A    FRONT PANEL L.E.D. PATTERNS
    UIVEC  EQU     40037A    USER 'INTERRUPT' VECTOR
    UO:CLK EQU     1A       ALLOW CLOCK INTERRUPT PROCESSING
    UO:DIOU EQU     2A      DISABLE DISPLAY UPDATE
    MI:JMP EQU     303A    MACHINE INSTRUCTION (8080) JUMP
    LEDOFF EQU     377A    BLANK L.E.D. DISPLAY
*** ****

* DISABLE UPDATING OF L.E.D. DISPLAY
* AND TURN OFF L.E.D.'S
* ****

040.100 076 002 PAMB
    HUI     A,0;0;0;0      DISABLE NORMAL UPDATING
    STA     MFLAG      DONE
    LXI     H,FPLEDS   L.E.D. DISPLAY ADDRESS
    MUL    B,9         COUNT L.E.D.'S
    A,LEDOFF   TURN OFF L.E.D.
    040.102 062 010 040
    040.105 041 013 040
    040.110 006 011
    040.112 076 377
    040.114 167
    040.115 043
    040.116 005
    040.117 302 114 040
* ****

* READ A DECIMAL INTEGER FROM HB FRONT PANEL
* IF NOT DECIMAL -- RETURN TO PAM-8.
* INCREMENT THE INTEGER (A PROGRAM REQUIREMENT)
* STORE THE DIGIT
* ****

040.122 315 260 003
    CALL    RCK      READ CONSOLE KEYPAD
    CPI    10D      TEST IF ZERO THRU NINE
    040.125 376 012
    040.127 322 322 000
    040.132 074
    040.133 062 254 040
* ****

* INITIALIZE CLOCK COUNTER.
* PROGRAM REQUIRES ONE INTERRUPT BEFORE DISPLAY
* ****

040.136 041 001 000
    040.141 042 252 040
* ****

* INITIALIZE SERVICE INTERRUPT ROUTINE
* LOAD THE USER INTERRUPT VECTOR (UIVEC) WITH A
* JUMP INSTRUCTION AND THE ADDRESS OF THE SERVICE
* ROUTINE...ENABLE USER CLOCK INTERRUPT!
* ****

```

```

        *      MVI    A,MI. JMP      SET-UP JUMP INSTRUCTION
        *      STA    U16C      STORE JUMP INSTRUCTION
        *      LXI    H,INTRP   USER INTERRUPT ADDRESS
        *      SHLD   U16C+1   POSITIONED
        *      MVI    A,LU,DU+UO,CLK  *ENABLE UPDATE & ENABLE CLOCK INT.
        *      STA    U16C+2   *DISABLE UPDATE & DISABLE CLOCK INT.
        *      ****
        *      * WAIT FOR CLOCK TO REACH ZERO
        *
        *      040,164...072,254,040  LOOP.....LDA....DIGIT.....DO NOTHING LOOP.
        *      040,167...376,000     CPI....0.....DO NOTHING LOOP.
        *      040,171...302,164,040  JNZ....LOOP.....WAIT FOR END.
        *      ****
        *      ****
        *      * RETURN TO NORMAL INTERRUPT STATUS & HALT.
        *
        *      * DISABLE INTERRUPT & TURN ON SPEAKER.
        *
        *      040,174...076,092... MVI....A,UD,DU.....A,UD,DU
        *      040,176...062,010,040  STA    MFLAG.....DISABLE UPDATE & CLOCK INTERRUPT
        *      040,201...076,372...  MVI....A,590/2.....250.MS.BEEP
        *      040,203...315,140,002  CALL   HORN
        *      040,206...166...      HL,T
        *      ****
        *      ****
        *      * INTERRUPT ROUTINE
        *      * CLOCK AND DISPLAY INTERRUPT
        *
        *      040,207...052,252,040  INTRE...LHLD...TICK.....GET COUNT (BETWEEN 0 & 500)
        *      040,212...053...      DCX   H.....TICK.....TICK=TICK-1
        *      040,213...042,252,040  SHLD   TICK.....STORE COUNT
        *      040,216...175...      MOV   A,L.....TEST FOR ZERO
        *      040,217...264...      ORA   H.....COMPARE WITH 'H'
        *      040,220...300...      RNE   RNE.....EXIT IF .NE. 0
        *      ****
        *      ****
        *      * UPDATE L,E,D,DISPLAY FOR .NEW..DIGIT.
        *
        *      040,221...072,254,040  LDA....DIGIT.....GET INTEGER
        *      040,224...075...      DCR   A.....DIGIT=DIGIT-1
        *      040,225...062,254,040  STA    A.....SAVE INTEGER
        *      040,230...041,356,003  LXI   H,100A   DECODE DISPLAY ADDRESS
        *      040,233...205...      ADD   L,A.....POSITION DISPLAY
        *      040,234...157...      MOV   L,A.....ALL SET -- GO
        *      040,235...176...      MOV   A,M.....DISPLAY SET
        *      040,236...366,200...  ORI   200Q.....MASK = TURN OFF D.P.
        *      040,240...062,017,040  STA    EP1E$+4.....TURN-ON-THE-LIGHTS.
        *      040,243...041,364,001  LXI   H,500.....RESTORE COUNT
        *      040,246...042,252,040  SHLD  TICK.....WITH 500
        *      040,251...311...      RET
        *      ****
        *      * STORAGE AREA & END ASSEMBLY
        *
        *      040,252...          TICK DS 2
        *      040,254...001...      DIGIT DB 1
        *      040,255...000...      FMB
        *
        *00130 STATEMENTS ASSEMBLED
        *12275 BYTES FREE
        *NO ERRORS DETECTED

```


INDEX

Addressing Port Pairs, 1-21
Advanced Control, 1-22
Alter Key, 1-9, 1-12
Altering a Memory Location, 1-12 ff
Altering a Selected Register, 1-15
Audio Alert, 1-9

Binary to Octal Conversion, 1-7
Breakpoint Interrupts, 1-24
Breakpointing, 1-16

Cancel, 1-9, 1-19
Checksum Errors, 1-20
Clock, 1-35, 1-6
Clock Interrupts, 1-5, 1-24
Copying a Tape, 1-20

Deccrements Memory, 1-9
Disable Interrupt, 1-5
Displays, 1-7, 1-23
DODA, 1-66
Dump Routines, 1-17 ff
Dumping, 1-18, 1-46

Execution Control, 1-16 ff
Entry Point, 1-19
Ending Address, 1-20
ERROR, 1-66

FPLEDS, 1-23 (1-60, 1-66)
GO Key, 1-16
Halt Instruction, 1-16
HORN, 1-66

I/O, 1-21
I/O Interrupts, 1-24
IP.PAD, 1-22
Increment Memory, 1-9
Initialization, 1-5
Inputting, 1-21
Interrupt Vectors, 1-31
Interrupts, 1-24, 1-26
Interrupting a Program, 1-17

Keypad, 1-9, 1-22

Load Routines, 1-17 ff
Loading, 1-18, 1-44

MEM Key, 1-9
.MFLAG (1-23, 1-66)
Manual DI, 1-23
Manual Updating, 1-23
Master Clear, 1-5
Monitor Mode, 1-6

Offset Octal, 1-8
Outputting, 1-21

Port I/O, 1-21
PAM/8 RAM Cells, 1-60
Power-Up, 1-5

RAM High Limit, 1-5
RCK, 1-22, 1-57, 1-66
REG Key, 1-9
RST, 1-5 ff, 1-9
RTM, 1-5 ff, 1-17, 1-9
Record Errors, 1-20
Refreshing, 1-23
Repeats, 1-9

- Single Instruction, 1-17
Single Instruction Interrupts, 1-24
Specifying a Memory Address, 1-10 ff
Specifying a Register for Display, 1-14
Stack Space, 1-5
Stepping Through Memory, 1-13
Stepping Through the Registers, 1-15

TICCNT, 1-22 Tape Errors, 1-20, 1-48
 Tick Counter, 1-22

 UIVEC, (1-60)
 USART Bit Definitions, 1-30
User Mode, 1-6
User Option Bits, 1-23, 1-29
UO.CLK, (1-66)
UO.DDU, (1-66)

○

○

○

C

C

C