

SOFTWARE REFERENCE MANUAL

H8 COMPUTER

FRONT PANEL MONITOR PAM-8

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/Ø) and lower left-hand (Ø) 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/Ø) 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.

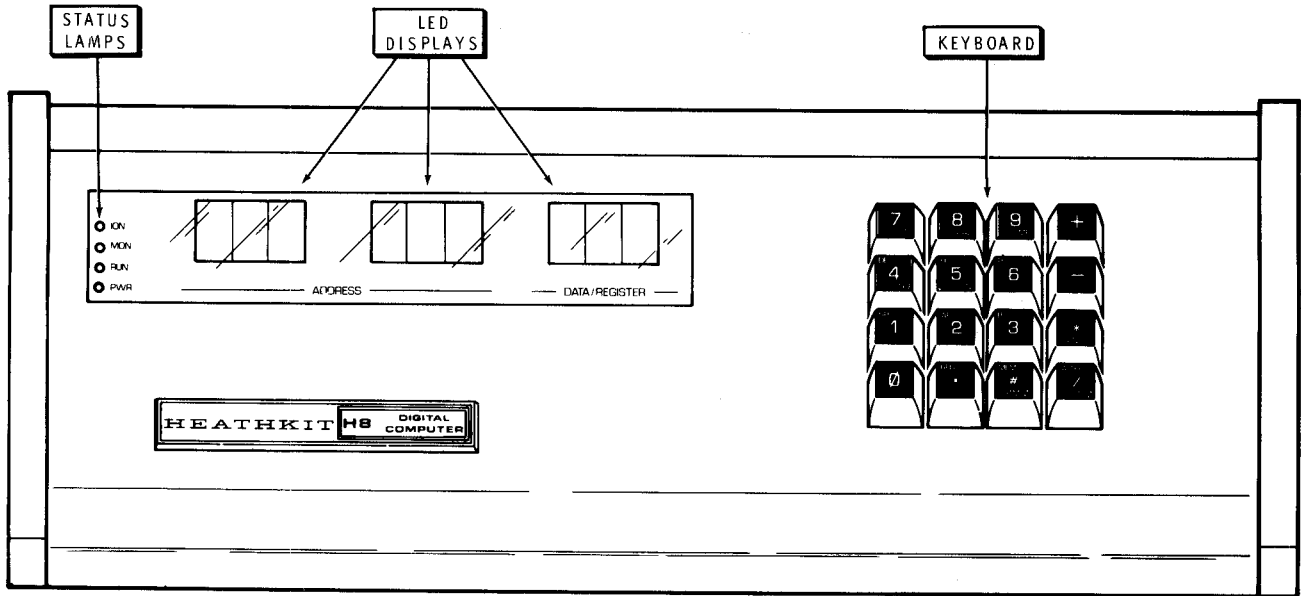


Figure 1-1

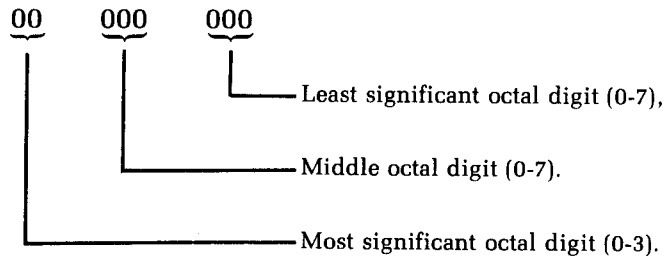
H8 Displays

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:

$\overbrace{11111111} \quad \overbrace{11111111}$	A 16-bit binary number																
<table style="margin: auto; border-collapse: collapse;"> <tr> <td style="text-align: center; padding: 0 5px;"> </td><td style="text-align: center; padding: 0 5px;"> </td><td style="text-align: center; padding: 0 5px;"> </td><td style="text-align: center; padding: 0 5px;"> </td><td style="text-align: center; padding: 0 5px;"> </td><td style="text-align: center; padding: 0 5px;"> </td><td style="text-align: center; padding: 0 5px;"> </td><td style="text-align: center; padding: 0 5px;"> </td> </tr> <tr> <td style="text-align: center; padding: 0 5px;">3</td><td style="text-align: center; padding: 0 5px;">7</td><td style="text-align: center; padding: 0 5px;">7</td><td style="text-align: center; padding: 0 5px;">3</td><td style="text-align: center; padding: 0 5px;">7</td><td style="text-align: center; padding: 0 5px;">7</td><td colspan="2"></td> </tr> </table>									3	7	7	3	7	7			Offset octal representation (377 377)
3	7	7	3	7	7												

$\overbrace{11111111} \quad \overbrace{11111111} \quad \overbrace{11111111} \quad \overbrace{11111111}$	A 16-bit binary number																
<table style="margin: auto; border-collapse: collapse;"> <tr> <td style="text-align: center; padding: 0 5px;"> </td><td style="text-align: center; padding: 0 5px;"> </td><td style="text-align: center; padding: 0 5px;"> </td><td style="text-align: center; padding: 0 5px;"> </td><td style="text-align: center; padding: 0 5px;"> </td><td style="text-align: center; padding: 0 5px;"> </td><td style="text-align: center; padding: 0 5px;"> </td><td style="text-align: center; padding: 0 5px;"> </td> </tr> <tr> <td style="text-align: center; padding: 0 5px;">1</td><td style="text-align: center; padding: 0 5px;">7</td><td style="text-align: center; padding: 0 5px;">7</td><td style="text-align: center; padding: 0 5px;">7</td><td style="text-align: center; padding: 0 5px;">7</td><td style="text-align: center; padding: 0 5px;">7</td><td colspan="2"></td> </tr> </table>									1	7	7	7	7	7			True Octal representation (177777)
1	7	7	7	7	7												

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.

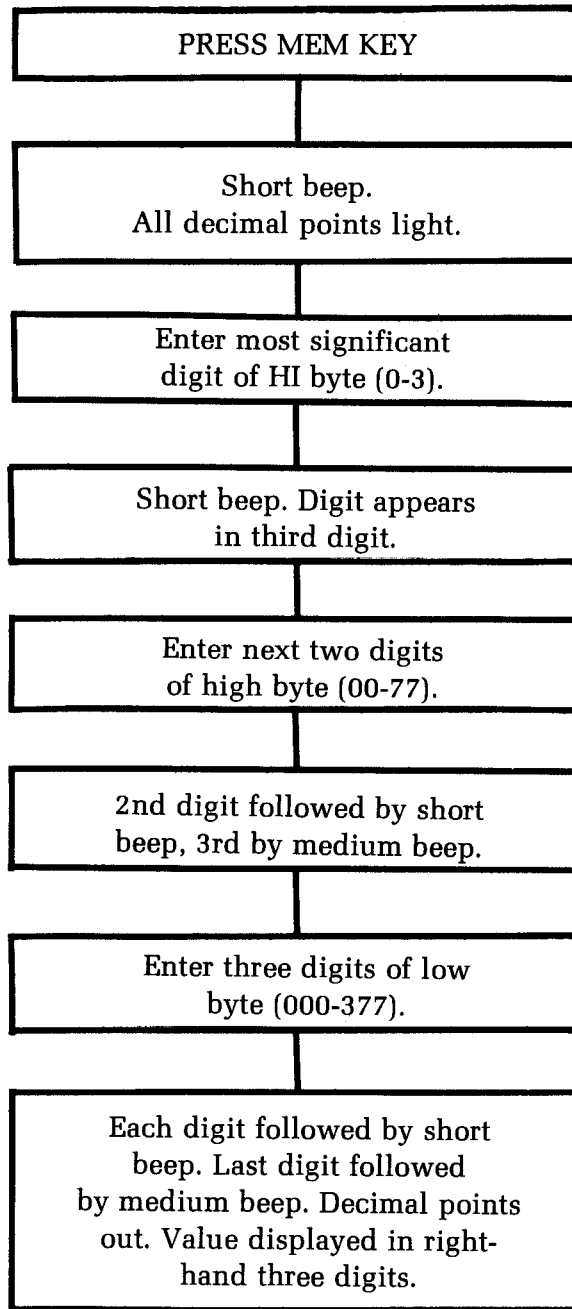


Figure 1-2

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 - H8 FRONT PANEL MONITOR #01.00.00. HEATH X89SM V1.1...06/21/77. 15:43:50 01-APR-77 PAGE 1

4. *** FAM/8 - H8 FRONT PANEL MONITOR.

5. *
6. * JGL, 05/01/76.
7. *
8. * FOR *WINTER* INC.

9. *
10. * COPYRIGHT 05/1976, WINTER CORPORATION,
11. * 902 N. 9TH ST,
12. * LAFAYETTE, IND.

14. *** FAM/8 - H8 FRONT PANEL MONITOR.

15. * THIS PROGRAM RESIDES (IN ROM) IN THE LOW 1024 BYTES OF THE HEATH
16. * H8 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. *

23. *** INTERRUPTS.

24. *
25. * FAM/8 IS THE PRIMARY PROCESSOR FOR ALL INTERRUPTS.
26. * THEY ARE PROCESSED AS FOLLOWS:

27. * RST USE

28. *
29. *
30. * 0 MASTER CLEAR. (NEVER USED FOR I/O OR RST)

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

- 37. * (STACK+0) = RETURN ADDRESS (TO FAM/8)
- 38. * (STACK+2) = (STACKPTR+14)
- 39. * (STACK+4) = (AF)
- 40. * (STACK+6) = (BC)
- 41. * (STACK+8) = (DE)
- 42. * (STACK+10) = (HL)
- 43. * (STACK+12) = (PC)
- 44. * THE USER'S ROUTINE SHOULD RETURN TO FAM/8 VIA
45. * A *RET* WITHOUT ENABLING INTERRUPTS.
- 46. *

47. *
48. * 2 SINGLE STEP INTERRUPTS GENERATED
49. * BY FAM/8 ARE PROCESSED BY FAM/8.
50. * ANY SINGLE STEP INTERRUPT RECEIVED WHEN IN
51. * USER MODE CAUSES A JUMP THROUGH *UIVEC*+3.
52. * STACK UPON USER ROUTINE ENTRY:
53. * (STACK+0) = (STACKPTR+12)
54. * (STACK+2) = (AF)
55. * (STACK+4) = (BC)

FAM/8 - HB FRONT PANEL MONITOR \$01.00.00. HEATH XBASM V1.1 06/21/77
INTRODUCTION, 15:43:51 01-APR-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**6
67 *
68 * 4 I/O 4. CAUSES A DIRECT JUMP THROUGH *UIVEC**9
69 *
70 * 5 I/O 5. CAUSES A DIRECT JUMP THROUGH *UIVEC**12
71 *
72 * 6 I/O 6. CAUSES A DIRECT JUMP THROUGH *UIVEC**15
73 *
74 * 7 I/O 7. CAUSES A DIRECT JUMP THROUGH *UIVEC**18

```

PAM/8 - HB FRONT PANEL MONITOR #01.00.00. HEATH XBASH V1.1 06/21/77
 ASSEMBLY CONSTANTS. 15:43:52. 01-APR-77. PAGE 3

77 ** ASSEMBLY CONSTANTS

79 ** I/O PORTS

80 000.360 EQU 3600 PAD INPUT PORT
 81 000.360 EQU 3600 CONTROL OUTPUT PORT
 82 000.360 EQU 3600 DIGIT SELECT OUTPUT PORT
 83 000.361 EQU 3610 SEGMENT SELECT OUTPUT PORT
 84 000.371 EQU 3710 TAPE CONTROL IN
 85 000.371 EQU 3710 TAPE CONTROL OUT
 86 000.370 EQU 3700 TAPE DATA IN
 87 000.370 EQU 3700 TAPE DATA OUT

90 ** ASCII CHARACTERS.

91 000.026 EQU 0260 SYNC CHARACTER
 92 000.002 EQU 0020 STX CHARACTER

95 ** FRONT PANEL HARDWARE CONTROL BITS.

96 000.020 EQU 00010000B SINGLE STEP INTERRUPT
 97 000.040 EQU 00100000B MONITOR LIGHT
 98 000.100 EQU 01000000B CLOCK INTERRUPT ENABLE
 99 000.200 EQU 10000000B SPEAKER ENABLE

102 ** DISPLAY MODE FLAGS (IN #DISPMOD*)

103 000.000 EQU 0 MEMORY READ
 104 000.001 EQU 1 MEMORY WRITE
 105 000.002 EQU 2 REGISTER READ
 106 000.003 EQU 3 REGISTER WRITE
 107 000.000 EQU XTEXT TAPE DEFINITIONS

110X ** TAPE EQUIVALENCES.

111X 000.001 EQU 1 RECORD TYPE - MEMORY DUMP IMAGE
 112X 000.002 EQU 2 RECORD TYPE - BASIC PROGRAM
 113X 000.003 EQU 3 RECORD TYPE - COMPRESSED TEXT

116X ** BLOCK SIZE FOR INTER-PRODUCT COMMUNICATION.

117X 002.000 EQU 512
 118X
 119X

PAN/8 - HB FRONT PANEL MONITOR #01.00.00. HEATH XBASM V1.1 06/21/77
 ASSEMBLY CONSTANTS. 15:43:56 01-APR-77 PAGE 4

121 ** MACHINE INSTRUCTIONS.

122 MI.HLT EQU 01110110B HALT
 000.166
 123 MI.RET EQU 11001001B RETURN
 000.311
 124 MI.IN EQU 11011011B INPUT
 000.333
 125 MI.OUT EQU 11010011B OUTPUT
 000.323
 126 MI.LDA EQU 00111010B LDA
 000.072
 127 MI.ANI EQU 11100110B ANI
 000.346
 128 MI.LXID EQU 00010001B LXI D
 000.021

131 ** USER OPTION BITS.

132 * THESE BITS ARE SET IN CELL .MFLAG.
 133 *
 134 UO.HLT EQU 10000000B DISABLE HALT PROCESSING
 000.200
 135 UO.NFR EQU 00000100B NO REFRESH OF FRONT PANEL
 000.100
 136 UO.DDU EQU 00000010B DISABLE DISPLAY UPDATE
 000.002
 137 UO.CLK EQU 00000001B ALLOW CLOCK INTERRUPT PROCESSING
 000.001

140 XTEXT U8251 DEFINE 8251 USART BITS

143X ** 8251 USART BIT DEFINITIONS.

144X *
 145X
 146X ** MODE INSTRUCTION CONTROL BITS.

000.100 UMI.1R EQU 01000000B 1 STOP BIT
 000.200 UMI.1B EQU 10000000B 1 1/2 STOP BITS
 000.300 UMI.2B EQU 11000000B 2 STOP BITS
 000.040 UMI.1FE EQU 00100000B EVEN PARITY
 000.020 UMI.1FA EQU 00010000B USE PARITY
 000.000 UMI.1S EQU 00000000B 5 BIT CHARACTERS
 000.004 UMI.1L5 EQU 00000100B 6 BIT CHARACTERS
 000.010 UMI.1L7 EQU 00001000B 7 BIT CHARACTERS
 000.014 UMI.1L8 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
 160X

161X ** COMMAND INSTRUCTION BITS.

162X UCI.1R EQU 01000000B INTERNAL RESET
 163X UCI.1R EQU 01000000B INTERNAL RESET
 164X UCI.R0 EQU 00100000B READER-ON CONTROL FLAG
 165X UCI.ER EQU 00010000B ERROR RESET
 166X UCI.RE EQU 00000100B RECEIVE ENABLE
 167X UCI.IE EQU 00000010B ENABLE INTERRUPTS FLAG
 168X UCI.1E EQU 00000001B TRANSMIT ENABLE
 169X

170X ** STATUS READ COMMAND BITS.

171X USR.1FE EQU 00100000B FRAMING ERROR
 172X USR.0E EQU 00010000B OVERRUN ERROR
 173X USR.0E EQU 00010000B OVERRUN ERROR
 174X USR.1FE EQU 00001000B PARITY ERROR
 175X USR.1XE EQU 00000100B TRANSMITTER EMPTY
 176X USR.1XR EQU 00000010B RECEIVER READY
 177X USR.1XR EQU 00000001B TRANSMITTER READY
 000.040
 000.020
 000.010
 000.004
 000.002
 000.001

FAM/8 - HB FRONT PANEL MONITOR #01.00.00.
HARDWARE INTERRUPT VECTORS

HEATH XBASM V1.0 02/18/77
13:23:25 01-APR-77 PAGE 6

180 *** INTERRUPT VECTORS.
181 *
182

184 ** LEVEL 0 - RESET
185 *
186 * THIS INTERRUPT MAY NOT BE PROCESSED BY A USER PROGRAM.
187

188 ORG 00A
189
190 INIT0 LXI D,FRSR0M (DE) = ROM COPY OF PRS CODE
191 LXI H,FRSRAM+FRSL-1 (HL) = RAM DESTINATION FOR CODE
192 JMP INIT INITIALIZE
193 ERRPL INIT-1000A BYTE IN WORD 10A MUST BE 0
377.073

195 ** LEVEL 1 - CLOCK
196 EQU IOR INTERRUPT ENTRY POINT
197 INT1 IOR
198

199 ERRNZ *-IIO INT0 TAKES UP ONE BYTE
200 CALL SAVALL SAVE USER REGISTERS
201 MVI D,0
202 JMP CLOCK PROCESS CLOCK INTERRUPT
203 ERRPL CLOCK-1000A EXTRA BYTE MUST BE 0
377.201

205 ** LEVEL 2 - SINGLE STEP
206 *
207 * IF THIS INTERRUPT IS RECEIVED WHEN NOT IN MONITOR MODE,
208 * THEN IT IS ASSUMED TO BE GENERATED BY A USER PROGRAM
209 * (SINGLE STEPPING OR BREAKPOINTING). IN SUCH CASE, THE
210 * USER PROGRAM IS ENTERED THROUGH (UIVEC+3
211

212 INT2 EQU 20A LEVEL 2 ENTRY
213

214 ERRNZ *-21A INT1 TAKES EXTRA BYTE
215 CALL SAVALL SAVE REGISTERS
216 LDAX D (A) = (CTLFLG)
217 SET CTLFLG
218 JMP STPRTN STEP RETURN
000.020
000.000
000.021 315 132 000
000.024 032
040.011
000.025 303 244 001

220 *** I/O INTERRUPT VECTORS.
221 *
222 *
223 * INTERRUPTS 3 THROUGH 7 ARE AVAILABLE FOR GENERAL I/O USE.
224 *
225 * THESE INTERRUPTS ARE NOT SUPPORTED BY FAM/8, AND SHOULD
226 * NEVER OCCUR UNLESS THE USER HAS SUPPLIED HANDLER ROUTINES
227 * (THROUGH UIVEC)

FAM/8 - H8 FRONT PANEL MONITOR #01.00.00. HEATH XBASM V1.0 02/18/77
 HARDWARE INTERRUPT VECTORS 13:23:26 01-APR-77 PAGE 7

000.030 228 30A 30A
 000.030 303 045 040 229 INT3 JMP UIVEC+6 JUMP TO USER ROUTINE
 000.033 064 064 064 230 DB 744413' HEATH PART NUMBER 444-13
 000.033 064 064 064 231 DB 744413' HEATH PART NUMBER 444-13

000.040 233 40A 40A
 000.040 303 050 040 234 INT4 JMP UIVEC+9 JUMP TO USER ROUTINE
 000.043 100 112 107 235 DB 1000:1120:1070:1140:1000 SUPPORT CODE

000.050 238 50A 50A
 000.050 303 053 040 239 INT5 JMP UIVEC+12 JUMP TO USER ROUTINE
 000.050 303 053 040 240 DB 1000:1120:1070:1140:1000 SUPPORT CODE
 000.050 303 053 040 241 DB 1000:1120:1070:1140:1000 SUPPORT CODE
 000.050 303 053 040 242 ** DLY - DELAY TIME INTERVAL.
 000.050 303 053 040 243 *
 000.050 303 053 040 244 * ENTRY (A) = MILLISECOND DELAY COUNT/2
 000.050 303 053 040 245 * EXIT NONE
 000.050 303 053 040 246 * USES A:F
 000.050 303 053 040 247 *
 000.053 365 248 DLY PUSH FSW SAVE COUNT
 000.054 257 249 XRA A DONT SOUND HORN
 000.055 303 143 002 250 JMP HRNO PROCESS AS HORN

000.060 252 60A 60A
 000.060 303 056 040 253 INT6 JMP UIVEC+15 JUMP TO USER ROUTINE
 000.060 303 056 040 254 DB 1000:1120:1070:1140:1000 SUPPORT CODE
 000.060 303 056 040 255 DB 1000:1120:1070:1140:1000 SUPPORT CODE
 000.063 076 320 256 60. MVI A:CR,SSI:CB,CLI:CB,SPK OFF MONITOR MODE LIGHT
 000.065 303 235 001 257 JMP SSI RETURN TO USER PROGRAM

000.070 259 70A 70A
 000.070 303 061 040 260 INT7 JMP UIVEC+18 JUMP TO USER ROUTINE

FAM/8 - HB FRONT PANEL MONITOR #01.00.00. HEALTH XBASM V1.0 02/18/77
 MASTER CLEAR PROCESSING 13:23:28 01-APR-77 PAGE 8

```

263 ** INIT - INITIALIZE SYSTEM
264 *
265 * INIT IS CALLED WHENEVER A HARDWARE MASTER-CLEAR IS INITIATED.
266 *
267 * SETUP FAM/8 CONTROL CELLS IN RAM.
268 * DECODE HOW MUCH MEMORY EXISTS; SETUP STACKPOINTER, AND
269 * ENTER THE MONITOR LOOP.
270 *
271 * ENTRY FROM MASTER CLEAR
272 * EXIT INTO FAM/8 MAIN LOOP
273
274
275 INIT LDAX D COPY #PRSR0M* INTO RAM
276 MOV M,A MOVE BYTE
277 DCX H DECREMENT DESTINATION
278 INR E INCREMENT SOURCE
279 JNZ INIT IF NOT DONE
280
281 SINCX EQU 4000A SEARCH INCREMENT
282
283 MVI D,SINCX/256 (DE) = SEARCH INCREMENT
284 LXI H,START-SINCX (HL) = FIRST RAM - SEARCH INCREMENT
285
286 * DETERMINE MEMORY LIMIT.
287
288 MOV M,A RESTORE VALUE READ
289 DAD D INCREMENT TRIAL ADDRESS
290 MOV A,M (A) = CURRENT MEMORY VALUE
291 DCX M TRY TO CHANGE IT
292 CMP M
293 JNE INITI IF MEMORY CHANGED
294
295 DCX H
296 SPHL SET STACKPOINTER = MEMORY LIMIT -1
297 PUSH H SET *PC* VALUE ON STACK
298 LXI H,ERROR
299 PUSH H SET RETURN ADDRESS
300
301 * CONFIGURE LOAD/DUMP UART.
302
303 MVI A,UMI,1B+UMI,1B+UMI,16X
304 OUT OP,1PC SET 8 BIT, NO PARITY, 1 STOP, XI6
    
```

FAM/8 - HB FRONT PANEL MONITOR \$01.00.00. HEATH X8ASM V1.0 02/18/77
 INTERRUPT TIME SUBROUTINES 13:23:29 01-APR-77 PAGE 9

```

307 ** SAVALL - SAVE ALL REGISTERS ON STACK.
308 *
309 * SAVALL IS CALLED WHEN AN INTERRUPT IS ACCEPTED, IN ORDER TO
310 * SAVE THE CONTENTS OF THE REGISTERS ON THE STACK.
311 *
312 * ENTRY CALLED DIRECTLY FROM INTERRUPT ROUTINE.
313 * ALL REGISTERS PUSHED ON STACK.
314 * IF NOT YET IN MONITOR MODE, REGPTR = ADDRESS OF REGISTERS
315 * ON STACK.
316 * (DE) = ADDRESS OF CTLFLG
317 *
318
319 SAVALL XTHL D SET H,L ON STACK TOP
320 PUSH D
321 PUSH B
322 PUSH FSW
323 XCHG (D,E) = RETURN ADDRESS
324 LXI H,10 (H,L) = ADDRESS OF USERS SP
325 DAD SP SET ON STACK AS REGISTER
326 PUSH H SET RETURN ADDRESS
327 PUSH D
328 LXI D,CTLFLG (A) = CTLFLG
329 LDAX D
330 CMA
331 ANI CB,MTL+CB,SSI SAVE REGISTER ADDR IF USER OR SINGLE-STEP
332 RZ RETURN IF WAS INTERRUPT OF MONITOR LOOP
333 LXI H,2
334 DAD SP (H,L) = ADDRESS OF STACKPTR ON STACK
335 SHLD REGPTR
336 RET
    
```

```

338 ** CUI - CHECK FOR USER INTERRUPT PROCESSING.
339 *
340 * CUI IS CALLED TO SEE IF THE USER HAS SPECIFIED PROCESSING
341 * FOR THE CLOCK INTERRUPT.
342 *
343
344 * SET MFLAG REFERENCE TO MFLAG
345 CUII LDAX B (A) = MFLAG
346 ERRNZ UU,CLK-1 CODE ASSUMED = 01
347 RRC
348 CC UIVEC IF SPECIFIED, TRANSFER TO USER
349
350 * RETURN TO PROGRAM FROM INTERRUPT.
351
352 INTXIT POP FSW REMOVE FAKE STACK REGISTER
353 POP FSW
354 POP B
355 POP D
356 POP H
357 EI
358 RET
    
```

FAM/8 - HB FRONT PANEL MONITOR #01.00.00. HEALTH_XBASM_V1.0_02/18/77
 PROCESS CLOCK INTERRUPTS 13:23:31 01-APR-77 PAGE 10

```

361 ***      CLOCK - PROCESS CLOCK INTERRUPT
362 *
363 *
364 *
365 *
366 *
367
368
000.201 052.033.040 369      LHLD      TICCNT
000.204 043          370      INX      H
000.205 042.033.040 371      SHLD     TICCNT      INCREMENT TICCOUNT
372
373 **
374 *
375 *
376 *
377 *
378 *
379
380
000.210 041 010 040 381      LXI      H,MFLAG
000.213 176          382      MOV      A,M
000.214 107          383      MOV      P,A
000.215 346 100     384      ANI      UO,NFR
000.217 043          385      INX      H
000.000          386      ERNZ     CTLFLG-MFLAG-1
000.220 176          387      MOV      A,M
000.221 112          388      MOV      C,D
000.222 302 237.000 389      JNZ     CLN3
000.225 043          390      INX      H
000.000          391      ERNZ     REFINO-CTLFLG-1
000.226 065          392      DCR      M
000.227 302 234 000 393      JNZ     CLN2
000.232 066 011     394      MVI      M,9
000.234 136          395      MOV      E,M
000.235 031          396      DAD      D
000.236 113          397      MOV      C,E
000.237          398      EQU     *
000.237 261          399      ORA     C
000.240 323 360     400      OUT     OF,DIG
000.242 176          401      MOV      A,M
000.243 323 361     402      OUT     OF,SEG
000.244          403
000.244          404 *
000.245 056 033     405      SEE IF TIME TO DECODE DISPLAY VALUES.
000.247 176          406      MVI      L,#TICCNT
000.250 346 037     407      MOV      A,M
000.252 314 161.003 408      ANI      37G
000.253          409      CZ      UFD
000.254          410      EVERY 32 INTERRUPTS
000.255 001 041 040 411 *      EXIT CLOCK INTERRUPT.
000.260 012          412
000.261 346 040     413      LXI      B,CTLFLG
000.263 302 172 000 414      LDAX     B
000.264          415      ANI      CR,MJL
000.265          416      JNZ     INTXIT      IF IN MONITOR MODE
    
```

(B) = CURRENT FLAG
 SEE IF FRONT PANEL REFRESH WANTED

(A) = CTLFLG
 (C) = 0 IN CASE NO PANEL DISPLAY
 IF NOT
 (H,L) = (REFIND)

DECREMENT DIGIT INDEX
 IF NOT WRAP-AROUND
 WRAP DISPLAY AROUND

(H,L) = ADDRESS OF PATTERN

(A) = CTLNLG
 (A) = INDEX + FIXED BITS
 SELECT DIGIT

SELECT SEGMENT

SEE IF TIME TO DECODE DISPLAY VALUES.

L,#TICCNT

EVERY 32 INTERRUPTS
 UPDATE FRONT PANEL DISPLAYS

(A) = CTLFLG
 IF IN MONITOR MODE

FAM/B - HB FRONT PANEL MONITOR #01.00.00. HEATH XBASM V1.0 02/18/77
 PROCESS CLOCK INTERRUPTS 13:23:34 01-APR-77 PAGE 11

```

000.266 013 417 ICX R
000.000 418 ERRNZ CTLFLG-,MFLAG-1
000.267 012 419 LDAX R (A) = MFLAG
000.000 420 ERRNZ UO.HLT-2000 ASSUME HIGH-ORDER
000.270 027 421 RAL
000.271 332 313 000 422 JC CLK4 SKIP IT
423
424 * NOT IN MONITOR MODE, CHECK FOR HALT
425
000.274 076 012 426 MVI A,10 (A) = INHDX OF *P* 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) = PC CONTENTS
000.304 033 431 ICX R
000.305 032 432 LDAX D
000.306 376 166 433 CPI MI.HLT. CHECK FOR HALT.
000.310 312 322 000 434 JE ERROR IF HALT, RE IN MONITOR MODE
435
436 * CHECK FOR RETURN TO MONITOR KEY ENTRY.
437
000.313 438 CLK4 EQU *
000.313 333 360 439 IN IP.PAD.
000.315 376 056 440 CPI 56R SEE IF '0' AND '*'
000.317 302 165 000 441 JNE CUI1 IF NOT, ALLOW USER PROCESSING OF CLOCK
    
```


FAM/8 - H8 FRONT PANEL MONITOR #01.00.00. HEATH X8ASM V1.0 02/18/77
 MTR - MAIN EXECUTIVE LOOP. 13:23:37 01-APR-77 PAGE 13

```

001.004 173      MOV      A,E      (A) = CODE
498
499
500 *          HAVE A COMMAND (NOT A VALUE)
501
001.005 326 004  SUI      A      (A) = COMMAND
001.007 332 322 000  JC      ERROR  IF BAD
504 001.012 137      MOV      E,A
505 001.013 345      PUSH     H      SAVE ABUSS VALUE
001.014 041 035 001  LXI      H,MTR4
001.017 026 000      MVI      D,0
508 001.021 031      DAD      D
509 001.022 136      MOV      E,M
510 001.023 031      DAD      D
511 001.024 343      XTHL
512 001.025 021 005 040  LXI      D,REGI
513 040.007      SET      DSPMOD
514 001.030 012      LDAX   R
515 001.031 346 002  ANI      2
516 001.033 012      LDAX   R
517 001.034 311      RET
518
519
001.035      EQU      *
520 001.035 165      DB      GO-*
521 001.036 141      DB      IN-*
522 001.037 143      DB      OUT-*
524 001.040 165      DB      SSTEP-*
525 001.041 220      DB      RMEM-*
526 001.042 332      DB      WMEM-*
527 001.043 067      DB      NEXT-*
528 001.044 104      DB      LAST-*
529 001.045 102      DB      ABORT-*
530 001.046 060      DB      R#M-*
531 001.047 116      DB      MEMM-*
532 001.050 034      DB      REGM-*
    
```

```

JUMP TABLE
4 - GO
5 - INPUT
6 - OUTPUT
7 - SINGLE STEP
8 - CASSETTE LOAD
9 - CASSETTE JUMP
+ - NEXT
- - LAST
* - ABORT
/ - DISPLAY/ALTER
# - MEMORY MODE
. - REGISTER MODE

534 **          PROCESS MEMORY/REGISTER ALTERATIONS.
535 *
536 *          THIS CODE IS ENTERED IF
537 *
538 *          1) AM IN ALTER MODE, AND
539 *          2) A KEY FROM 0-7 WAS ENTERED.
540
541 MTR5      RRC
542 001.052 173      MOV      A,E      (A) = VALUE
001.053 332 067 001  JC      MTR6      IS REGISTER
544 001.056 067      STC          INDICATE 1ST DIGIT IS IN (A)
001.057 315 066 003  CALL   IDB      INPUT OCTAL BYTE
001.062 043      INX          DISPLAY NEXT LOCATION
546
    
```

FAM/8 - H8 FRONT PANEL MONITOR *01.00.00, HEATH XRASM.V1.0...02/1R/77 14
MTR - MAIN EXECUTIVE LOOP, 13:23:39 01-APR-77 PAGE 14

```

548 ** SAE STORE ARUSS AND EXIT.
549 *
550 * ENTRY (HL) = ARUSS VALUE
551 * EXIT TO (RET)
552 * USES NONE
553 *
001.063 042 024 040 SAE SHLD ARUSS
001.066 311 RET
554 SAE
555 RET
556 *
557 * ALTER REGISTER
558
559 MTR6 PUSH FSW SAVE CODE
001.070 315 047 003 CALL LRA LOCATE REGISTER ADDRESS
001.073 247 ANA A
001.074 312 322 000 JZ ERROR NOT ALLOWED TO ALTER STACKPOINTER
001.077 043 INX H
001.100 361 POP FSW RESTORE VALUE AND CARRY FLAG
001.101 303 062 003 JMP IOA INPUT OCTAL ADDRESS

```

FAM/8 - HG FRONT PANEL MONITOR. #01.00.00. HEALTH XBASH V1.1. 06/21/77. 15:44:14 01-APR-77 PAGE 15
MONITOR TASK SUBROUTINES.

```

569 ** REGM - ENTER REGISTER DISPLAY MODE.
570 *
571 * ENTRY (A) = DSPMOD
572 * (BC) = #DSPMOD
573
574 REGM MVI A,2 SET DISPLAY REGISTER MODE
575 SET DSPMOD
576 STAX B SET DISPLAY REGISTER MODE
577 ERRNZ DSPMOD-DSPROT-1 (BC) = #DSPROT
578 DCX B
579 XRA A
580 STAX B SET ALL PERIODS ON
581 CALL RCK READ KEY ENTRY
582 DCX A DISPLACE
583 CPI 6 NOT 1-6
584 JNC ERROR
585 RLC
586 STAX D SET NEW REG IND
587 SET REGI
588 RET

```

```

590 ** R#W - TOGGLE DISPLAY/ALTER MODE.
591 *
592 * ENTRY (A) = DSPMOD
593 * (BC) = ADDRESS OF DSPMOD
594
595 SET DSPMOD
596 XRI 1
597 STAX B
598 RET

```

```

600 ** NEXT - INCREMENT DISPLAY ELEMENT.
601 *
602 * ENTRY (HL) = (ARUSS)
603 * (DE) = ADDRESS OF REGIND
604
605 NEXT INX H
606 JZ SAE IF MEMORY, STORE ARUSS AND EXIT
607
608 * IS REGISTER MODE.
609
610 SET REGI
611 LDAX D (A) = REGI
612 ADI 2 INCREMENT REG INNEX
613 STAX D WRAP TO *SP*
614 CPI 12
615 RC IF NOT TOO LARGE, EXIT
616 XRA A OVERFLOW
617 STAX D
618 ABORT RET

```


FAM/B - HB FRONT PANEL MONITOR #01.00.00. HEATH X8ASM V1.1 06/21/77
 MONITOR TASK SUBROUTINES. 15:44:16 01-APR-77 PAGE 16

```

620 ** LAST - DECREMENT DISPLAY ELEMENT.
621 *
622 * ENTRY (HL) = (ABUSS)
623 * (DE) = ADDRESS OF REGIND
624
001.150 053 H
001.151 312.063 001 JZ SAE IF MEMORY, STORE AND EXIT
627
628 * IS REGISTER MODE.
629
040.005 SET REGI
001.154 032 LDAX D (A) = REGI
001.155 326.002 SUI 2
001.157 023 STAX D
001.160 320 RNC
001.161 076 012 MVI A,10 IF OK
001.163 022 STAX D UNDERFLOW TO #PCX
001.164 311 RET
    
```

```

640 ** MEMM - ENTER DISPLAY MEMORY MODE.
641 *
642 * ENTRY (BC) = ADDRESS OF ISPMOD
643
001.165 257 XRA A (A) = 0
040.007 SET ISPMOD
001.166 002 STAX B SET DISPLAY MEMORY MODE
000.000 ERRNZ ISPMOD-ISPROT-1
001.167 013 DCX B (BC) = #ISPROT
001.170 002 STAX B SET ALL PERIODS ON
001.171 041 025 040 LXI H,ABUSS+1
001.174 303 062 003 JMP IOA INPUT OCTAL ADDRESS
    
```

```

653 ** IN - INPUT DATA BYTE.
654 *
655
656 ** OUT - OUTPUT DATA BYTE.
657 *
658 * ENTRY (HL) = (ABUSS)
659
001.177 006 333 IN MVI B,MI,IN
001.201 021 DB MI, LXID SKIP NEXT INSTRUCTION
001.202 006 323 OUT MVI B,MI,OUT
001.204 174 MOV A,H (A) = VALUE
001.205 145 MOV H,L (H) = PORT
001.206 150 MOV L,B (L) = IN/OUT INSTRUCTION
001.207 042 002 040 SHLD IOWRK
001.212 315 002 040 CALL IOWRK PERFORM IO
001.215 154 MOV L,H (L) = PORT
001.216 147 MOV H,A (H) = VALUE
001.217 303 063 001 JMP SAE STORE ABUSS AND EXIT
    
```

FAN/8 - H8 FRONT PANEL MONITOR *01.00.00.
 60 AND *STEP* FUNCTIONS

HEATH X8ASM V1.0 02/18/77
 13:23:43 01-APR-77 PAGE 18

```

675 **      GO - RETURN TO USER MODE
676 *
677 *      ENTRY NONE
678 *
001.222 303 063 000 JMF 60
ROUTINE IS IN WASTE SPACE
    
```

```

681 **      SSTEP - SINGLE STEP INSTRUCTION.
682 *
683 *      ENTRY NONE
684 *
001.225      SSTEP EQU *      SINGLE STEP
001.225 363      DI          DISABLE INTERRUPTS UNTIL THE RIGHT TIME
001.226 072 011 040 LDA      CTLFLG
001.231 356 020 XRI      CB,SSI  CLEAR SINGLE STEP INHIBIT
001.233 323 360 OUT      OP,CTL  PRIME SINGLE STEP INTERRUPT
001.235 062 011 040 STA      CTLFLG  SET NEW FLAG VALUES
001.240 341      POP      H      CLEAN STACK
001.241 303 172 000 JMP      INTXIT  RETURN TO USER ROUTINE FOR STEP
    
```

```

694 **      STPRTN - SINGLE STEP RETURN
695 *
001.244      STPRTN EQU *
001.244 366 020 ORI      CB,SSI  DISABLE SINGLE STEP INTERRUPTION
001.246 323 360 OUT      OP,CTL  TURN OFF SINGLE STEP ENABLE
040.011      SET      CTLFLG
001.250 022      STAX      D
001.251 346 040 ANI      CB,MTL  SEE IF IN MONITOR MODE
001.253 302 344 000 JNZ      MTR
001.256 303 042 040 JMP      UIVED+3  TRANSFER TO USER'S ROUTINE
    
```

```

705 **      RMEM - LOAD MEMORY FROM TAPE.
706 *
707
001.261 041 244 002 LXI      H,TFABT
001.264 042 031 040 SHLD   TFERRX  SETUP ERROR EXIT ADDRESS
710 *      JMP      LOAD
    
```

FAN/B - H8 FRONT PANEL MONITOR #01.00.00. HEATH XBASH V1.1 06/21/77
 LOAD - LOAD MEMORY FROM TAPE. 15:44:19 01-APR-77 PAGE 19

712	***	LOAD - LOAD MEMORY FROM TAPE.
713	*	
714	*	READ THE NEXT RECORD FROM THE CASSETTE TAPE.
715	*	
716	*	USE THE LOAD ADDRESS IN THE TAPE RECORD.
717	*	
718	*	ENTRY (HL) = ERROR EXIT ADDRESS
719	*	USER P-REG (IN STACK) SET TO ENTRY ADDRESS
720	*	TO CALLER IF ALL OK
721	*	TO ERROR EXIT IF TAPE ERRORS DETECTED.
722	*	
723		
001.267	LOAD	ERU *
001.267		LXI R,1000A-RT.MI*256-256 (BC) = - REQUIRED TYPE AND #
001.272	LOAD	CALL SRS SCAN FOR RECORD START
001.275		MOV L,A (HL) = COUNT
001.276		XCHG C (DE) = COUNT, (HL) = TYPE AND #
001.277		DCR B (C) = - NEXT #
001.300		INR B
001.301		MOV A,H
001.302		PUSH B SAVE TYPE AND #
001.303		PUSH PSM SAVE TYPE CODE
001.304		ANI 177Q CLEAR END FLAG BIT
001.306		ORA L
001.307		MVI A,2
001.311		JNE TFERR SEQUENCE ERROR
001.314		CALL RNF IF NOT RIGHT TYPE OR SEQUENCE
001.317		MOV B,H READ ADDR
001.320		MOV C,A (BC) = P-REG ADDRESS
001.321		MVI D
001.323		PUSH D SAVE (DE)
001.324		CALL LRA LOCATE REG ADDRESS
001.327		POP D RESTORE (DE)
001.330		MOV M,C SET P-REG IN MEM
001.331		INX H
001.332		MOV M,B
001.333		CALL RNF
001.336		MOV L,A READ ADDRESS
001.337		SHLD START (HL) = ADDRESS, (DE) = COUNT
001.342	LOAD	CALL RNB READ BYTE
001.343		MOV M,A
001.346		SHLD ABUSS SET ABUSS FOR DISPLAY
001.351		INX D
001.352		DCR D
001.353		MOV A,D
001.354		ORA E
001.355		JNZ LOA1 IF MORE TO GO
001.360		CALL CTC CHECK TAPE CHECKSUM
762	*	
763	*	READ NEXT BLOCK
764	*	
001.363		POP FSW FSW
001.364		POP B (A) = FILE TYPE BYTE
001.365		RLC (BC) = -(LAST TYPE, LAST #)

FAM/8 - HB FRONT PANEL MONITOR \$01.00.00.
LOAD - LOAD MEMORY FROM TAPE.....

HEATH X8ASH VI.1 06/21/77

15:44:21 01-APR-77 PAGE: 20

001.366 332 133 002 768 JC TFT
001.371 303 272 001 769 JMP LOAD

ALL DONE - TURN OFF TAPE
READ ANOTHER RECORD.....

PAM/8 - HB FRONT PANEL MONITOR #01.00.00.
DUMP - DUMP MEMORY TO MAG/PAPER TAPE

```

002.115 172      828      MOV      A,D
002.116 263      829      E
002.117 302 104 002 830      JNZ      WME2      IF MORE TO GO
002.118 302 104 002 831      *
002.119 302 104 002 832      *
002.120 302 104 002 833      *
002.121 302 104 002 834      *
002.122 052 027 040 834      LHL D   CRCSUM
002.123 315 017 003 835      CALL  WNF      WRITE IT
002.124 315 017 003 836      CALL  WNF      FLUSH CHECKSUM
002.125 315 017 003 837      JMP      TFT
    
```

```

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

```

002.143 343      858      HRNO
002.144 325      859      XTHL
002.145 353      860      PUSH   D
002.146 041 011 040 861      LXI    H,CTLFLG
002.151 256      862      XRA    M
002.152 136      863      MOV    E,M
002.153 167      864      MOV    M,A
002.154 056 033 865      MVI    L,#TICCNT
002.156 172      866      MOV    A,D
002.157 206      868      ADD    M
002.160 276      869      CMF    M
002.161 302 160 002 870      JNE    HRN2
002.164 056 011 871      MVI    L,#CTLFLG
002.166 163      872      MOV    M,E
002.167 321      873      POP    D
002.170 341      874      POP    H
002.171 311      875      RET
    
```

839 ** TFT - TURN OFF TAPE.

840 *
841 * STOP THE TAPE TRANSPORT.
842 *
843

847 ** HORN - MAKE NOISE.

848 *
849 * ENTRY (A) = (MILLISECOND COUNT)/2
850 * EXIT NONE
851 * USES A/F
852
853

```

002.136 076 144 854      ALARM  MVI    A,200/2  200 MS BEEP
002.140 365      855      HRN    PUSH   PSW
002.141 076 200 856      MVI    A,CR,SPK  TURN ON SPEAKER
    
```

```

002.143 343      858      HRNO
002.144 325      859      XTHL
002.145 353      860      PUSH   D
002.146 041 011 040 861      LXI    H,CTLFLG
    
```

```

002.151 256      862      XRA    M
002.152 136      863      MOV    E,M
002.153 167      864      MOV    M,A
002.154 056 033 865      MVI    L,#TICCNT
    
```

```

002.156 172      866      MOV    A,D
002.157 206      868      ADD    M
002.160 276      869      CMF    M
002.161 302 160 002 870      JNE    HRN2
    
```

```

002.164 056 011 871      MVI    L,#CTLFLG
002.166 163      872      MOV    M,E
002.167 321      873      POP    D
002.170 341      874      POP    H
002.171 311      875      RET
    
```

WAIT REQUIRED TICCOUNTS

TURN HORN OFF.

SAVE (HL); (H) = COUNT
SAVE (DE)
(D) = LOOP COUNT

(E) = OLD CTLFLG VALUE
TURN ON HORN

(A) = CYCLE COUNT

FAM/B - HB FRONT PANEL MONITOR #01.00.00. HEATH XBASM V1.1 06/21/77
 TAPE PROCESSING SUBROUTINES 15:44:25 01-APR-77 PAGE 23

```

880 **      CTC - VERIFY CHECKSUM.
881 *
882 *      ENTRY TAPE JUST BEFORE CRC
883 *      EXIT TO CALLER IF OK
884 *      TO *TFERR* IF BAD
885 *      USES A/F,H,L
886
002.172 315 325 002 888 CTC          RMP          READ NEXT PAIR
002.175 052 027 040 889 LHL          CRCSUM
002.200 174          890 MOV          A,H
002.201 265          891 OR          L
002.202 310          892 RZ
002.203 076 001     893 MVI          A,1          RETURN OF OK
                                CHECKSUM ERROR
                                (R) = CODE
894 *      JMP.          TFERR.
896 **      TFERR - PROCESS TAPE ERROR.
897 *
898 *      DISPLAY ERR NUMBER IN LOW BYTE OF ARUSS
899 *
900 *      IF ERROR NUMBER EVEN, DON'T ALLOW *
901 *      IF ERROR NUMBER ODD, ALLOW *
902 *
903 *      ENTRY (A) = NUMBER
904
905
002.205 062 024 040 906 TPERR          STA          ARUSS
002.210 107          907 MOV          B,A          (B) = CODE
002.211 315 133 002 908 CALL          TFT          TURN OFF TAPE
909
910 *      IS #, RETURN (IF PARITY ERROR)
911
912
913 TER3          DB          MI,ANI          FALL THROUGH WITH CARRY CLEAR
914          MOV          A,B
002.216 017          915 RRC
002.217 330          916 RC          RETURN IF OK
917
918 *      BEEP AND FLASH ERROR NUMBER
919
002.220 334 136 002 920 TER1          CC          ALARM IF PROPER TIME
002.223 315 252 002 921 CALL          IPXIT          SEE IF *
002.226 333 360 922 IN          IP, PAD
002.230 376 057 923 CPI          0010111B          CHECK FOR *
002.232 312 215 002 924 JE          TERS          IF *
002.235 072 034 040 925 LDA          TICCNT+1
002.240 037          926 RAR
002.241 303 220 002 927 JMP.          TERS
    
```

FAM/8 - HB FRONT PANEL MONITOR #01.00.00. HEATH XBASM V1.0 02/18/77
 TAPE PROCESSING SUBROUTINES 13:23:52 01-APR-77 PAGE 24

```

929 ** TPART - ABORT TAPE LOAD OR DUMP.
930 *
931 * ENTERED WHEN LOADING OR DUMPING, AND THE '*' KEY
932 * IS STRUCK.
933
002,244 257 TPART XRA A
002,245 323 371 OUT OP,TPC OFF TAPE
002,247 303 322 000 JMP ERROR
    
```

```

939 ** TPXIT - CHECK FOR USER FORCED EXIT.
940 *
941 * TPXIT CHECKS FOR AN '*' KEYPAD ENTRY. IF SO, TAKE
942 * THE TAPE DRIVER ABNORMAL EXIT.
943 *
944 * ENTRY NONE
945 * TO *RET* IF NOT '*'
946 * (A) = PORT STATUS
947 * TO (TPERRX) IF '*' DOWN
948 *
949 * USES A,F
950
002,252 333 360 IN IP,FAD
002,254 376 157 CPI 01101111R * READ TAPE STATUS
002,256 333 371 IN IP,TPC NOT '*', RETURN WITH STATUS
002,260 300 RNE TPERRX ENTER (TPERRX)
002,261 052 031 040 LHLD
002,264 351 FCHL
    
```

```

958 ** SRS - SCAN RECORD START
959 *
960 * SRS READS BYTES UNTIL IT RECOGNIZES THE START OF A RECORD.
961 *
962 * THIS REQUIRES
963 * AT LEAST 10 SYNC CHARACTERS
964 * 1 SIX CHARACTER.
965 *
966 * THE CRC-16 IS THEN INITIALIZED.
967 *
968 * ENTRY NONE
969 * EXIT TAPE POSITIONED (AND MOVING), CRC SUM = 0
970 * (DE) = HEADER BYTES
971 * (HA) = RECORD COUNT
972 * USES A,F,D,E,H,L
973
002,265 EQU *
002,265 026 000 MVI D,0
002,267 142 MOV H,D
002,270 152 MOV L,D (HL) = 0
974
975 SRS
976 SRS1
977
978
    
```


PAM/8 - H8 FRONT PANEL MONITOR #01.00.00. HEATH XBASH VI.0 02/18/77
 TAPE PROCESSING SUBROUTINES 13:23:54 01-APR-77 PAGE 25

```

002:271 315 331 002 979 SRS2 CALL RNB READ NEXT BYTE
002:274 024 024 INR D
002:275 376 026 981 CFI A,SYN
002:277 312 271 002 982 JE SRS2 HAVE SYN
002:302 376 002 983 CFI A,STX
002:304 302 265 002 984 JNE SRS1 NOT STX - START OVER
002:307 076 012 985 MVI A,10
002:311 272 987 CMF D
002:312 322 265 002 988 JNC SRS1 SEE IF ENOUGH SYN CHARACTERS
002:315 042 027 040 989 SHLD CRCSUM NOT ENOUGH
002:320 315 325 002 990 CALL RNF CLEAR CRC-16
002:323 124 991 MOV D,H READ LEADER
002:324 137 992 MOV E,A
002:324 137 993 JMP RNF READ COUNT
    
```

```

995 ** RNF - READ NEXT PAIR.
996 *
997 * RNF READS THE NEXT TWO BYTES FROM THE INPUT DEVICE.
998 *
999 * ENTRY NONE
1000 * EXIT (H,A) = BYTE PAIR
1001 * USES A,F,H
1002 *
1003 *
    
```

```

002:325 315 331 002 1004 RNF CALL RNB READ NEXT BYTE
002:330 147 1005 MOV H,A
1006 * JMP RNB READ NEXT BYTE
    
```

```

1008 ** RNB - READ NEXT BYTE
1009 *
1010 * RNB READS THE NEXT SINGLE BYTE FROM THE INPUT DEVICE.
1011 * THE CHECKSUM IS TAKEN FOR THE CHARACTER.
1012 *
1013 * ENTRY NONE
1014 * EXIT (A) = CHARACTER
1015 * USES A,F
1016 *
1017 *
1018 RNB MVI A,UCI,RO+UCI,ER+UCI,RE TURN ON PEADER FOR NEXT BYTE
002:331 076 064 1019 RNB OUT OP,TFC
002:333 323 371 1020 RNB1 CALL TFXIT CHECK FOR *, READ STATUS
002:335 315 252 002 1021 ANI USR,RXR
002:340 346 002 1022 JZ RMB1 IF NOT READY
002:342 312 335 002 1023 IN IF,IFD INPUT DATA
002:345 333 370 1024 * JMP RNB CHECKSUM
    
```


PAK/B - HB FRONT PANEL MONITOR #01.00.00. HEATH XBASM V1.0 02/18/77
 TAPE PROCESSING SUBROUTINES 13:23:58 01-APR-77 PAGE 27

```

1074 ** WNF - WRITE NEXT PAIR.
1075 *
1076 * WPT WRITES THE NEXT TWO BYTES TO THE CASSETTE DRIVE.
1077 *
1078 * ENTRY (HAL) = BYTES.
1079 * EXIT WRITTEN.
1080 * USES A,F.
1081
1082
1083
1084 MOV A,H
1085 CALL WNR
1086 JMP WNF
1087
1088 ** WNB - WRITE BYTE
1089 *
1090 * WNB WRITES THE NEXT BYTE TO THE CASSETTE TAPE.
1091 *
1092 * ENTRY (A) = BYTE
1093 * EXIT NONE.
1094 * USES F
1095
1096
1097 WNB FUSH PSW
1098 WNB1 CALL TFXIT CHECK FOR *, READ STATUS
1099 ANI USR,TRX IF MORE TO GO
1100 JZ WNB1
1101 MVI A,UCI+ER+UCI+TE ENABLE TRANSMITTER
1102 OUT DP,TPC TURN ON TAPE
1103 POP PSW
1104 OUT DP,TFD OUTPUT DATA
1105 JMP JMF COMPUTE CRC
    
```

FAM/8 - H8 FRONT PANEL MONITOR #01.00.00. HEATH XBASM V1.0 02/18/77
 SUBROUTINES 13:23:59 01-APR-77 PAGE 28

```

1109 ** LRA - LOCATE REGISTER ADDRESS.
1110 *
1111 * ENTRY NONE.
1112 * EXIT (A) = REGISTER INDEX
1113 * (H,L) = STORAGE ADDRESS
1114 * (B,E) = (0,A)
1115 * USES A,D,E,H,L,F
1116
1117
1118
003.047 072 005 040 1119 LRA LDA REGI
003.052 137 000 000 1120 LRA MOV E,A
003.053 026 000 1121 MVI D,0
003.055 052 035 040 1122 LHL D REGPTR
003.060 031 1123 DAD D (DE) = (REGPTR)+(REGI)
003.061 311 1124 RET
    
```

```

1126 ** IOA - INPUT OCTAL ADDRESS.
1127 *
1128 * ENTRY (H,L) = ADDRESS OF RECEPTION DOUBLE BYTE.
1129 * EXIT TO *RET* IF *ERRK*.
1130 * TO *RET*+1 IF OK, VALUE IN MEMORY.
1131 * USES A,D,E,H,L,F
1132
1133
003.062 315 066 003 1134 IOA CALL IOB INPUT BYTE
003.065 053 1135 DCX H
    
```

```

1137 ** IOB - INPUT OCTAL BYTE.
1138 *
1139 * READ ONE OCTAL BYTE FROM THE KEYS.
1140 *
1141 * ENTRY (H,L) = ADDRESS OF BYTE TO HOLD VALUE
1142 * 'C' SET IF FIRST DIGIT IN 'A'
1143 * EXIT TO *RET* IF ALL OK
1144 * TO *ERRK* IF *ERRK*
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 IOB1 CNC RCK READ CONSOLE KEYS
1151
003.073 376 010 1152 CPI 8
003.075 322 322 000 1153 JNC ERROR IF ILLEGAL DIGIT
1154
003.100 137 1155 MOV E,A (E) = VALUE
003.101 176 1156 MOV A,M
003.102 007 1157 RLC SHIFT 3
003.103 007 1158 RLC
    
```

PAM/8 - HS FRONT PANEL MONITOR #01.00.00. HEATH XBASH V1.0 02/18/77
 SUBROUTINES 13:24:01 01-APR-77 PAGE 29

```

003.104 007      RLC
003.105 346 370  ANI      3700
003.107 263      ORA      E
003.110 167      MOV      M,A
003.111 025      DCR      D
003.112 302 070 003  JNZ      I0B1
003.115 076 017      MVI      A,30/2
003.117 303 140 002  JMP      HORN

1168 **      JDD      DECODE FOR OCTAL DISPLAY.
1169 *
1170 *      ENTRY      (H,L) = ADDRESS OF LED REFRESH AREA
1171 *      (B) = *OR* PATTERN TO FORCE ON BARS OR PERIODS
1172 *      (A) = OCTAL VALUE
1173 *      (H,L) = NEX DIGIT ADDRESS
1174 *      USES      A,B,C,D,H,L
1175 *
1176
1177 JDD
1178 MVI      D,D0DA/256
1179 MVI      C,3
1180 DDDA
1181 RAL
1182 RAL
1183 PUSH     PSW
1184 ANI      7
1185 ADI      #D0DA
1186 MOV      E,A
1187 LDAX   D
1188 XRA      B
1189 ANI      1778
1190 XRA      B
1191 MOV      M,A
1192 INX      H
1193 MOV      A,B
1194 RLC
1195 MOV      B,A
1196 POP      PSW
1197 DCR      C
1198 JNZ      DDD1
1199 POP      D
1200 RET

      DECODE FOR OCTAL DISPLAY.
      (H,L) = ADDRESS OF LED REFRESH AREA
      (B) = *OR* PATTERN TO FORCE ON BARS OR PERIODS
      (A) = OCTAL VALUE
      (H,L) = NEX DIGIT ADDRESS
      USES      A,B,C,D,H,L

      JDD
      MVI      D,D0DA/256
      MVI      C,3
      DDDA
      RAL
      RAL
      PUSH     PSW
      ANI      7
      ADI      #D0DA
      MOV      E,A
      LDAX   D
      XRA      B
      ANI      1778
      XRA      B
      MOV      M,A
      INX      H
      MOV      A,B
      RLC
      MOV      B,A
      POP      PSW
      DCR      C
      JNZ      DDD1
      POP      D
      RET

      (A) = VALUE
      IF MORE TO GO
      RETURN
    
```

PAM/B - HB FRONT PANEL MONITOR #01.00.00. HEATH XBASM V1.0 02/18/77
 UFD - UPDATE FRONT PANEL DISPLAYS. 13:24:02 01-APR-77 PAGE 30

```

1203 ** UFD - UPDATE FRONT PANEL DISPLAYS.
1204 *
1205 *
1206 *
1207 * UFD IS CALLED BY THE CLOCK INTERRUPT PROCESSOR WHEN IT IS
1208 * TIME TO UPDATE THE DISPLAY CONTENTS. CURRENTLY, THIS IS DONE
1209 * EVERY 32 INTERRUPTS, OR ABOUT 32 TIMES A SECOND.
1210 * ENTRY (H,L) = ADDRESS OF REFCNT
1211 * EXIT NONE
1212 * USES ALL
1213
1214
1215 UFD *
1216 MVI A,DU,DDU
1217 ANA B
1218 RNZ
1219
1220 MVI L,#DSPROT
1221 MOV A,M
1222 RLC
1223 MOV M,A
1224 MOV B,A
1225 INX H
1226 ERRCNZ DSPMOD-DSPROT-1 (A) = DSPMOD
1227 MOV A,M
1228 ANI 2
1229 LHLD ABUSS
1230 JZ UFD1 IF MEMORY
1231 *
1232 * AN DISPLAYING REGISTERS.
1233
1234 CALL LRA LOCATE REGISTER ADDRESS
1235 PUSH H
1236 LXI H,DISPA
1237 DAD D (H,L) = ADDRESS OF REG. NAME PATTERNS
1238 MOV A,M
1239 INX H
1240 MOV H,M
1241 MOV L,A (H,L) = REG NAME PATTERN
1242 XTHL
1243 ORA H CLEAR 'Z'
1244 MOV A,M
1245 INX H
1246 MOV H,M
1247 MOV L,A (HL) = ADDRESS OF REGISTER PAIR CONTENTS
1248
1249 * SETUP DISPLAY
1250
1251 UFD1
1252 PUSH FSW
1253 XCHG
1254 LXI H,ALERS
1255 MOV A,D
1256 CALL J00
1257 MOV A,E
1258 CALL J0D
1259 POP FSW
    
```



FAM/B - H8 FRONT PANEL MONITOR #01.00.00. HEATH XBASM V1.0 02/18/77
UFD - UPDATE FRONT PANEL DISPLAYS. 13:24:04 01-APR-77 PAGE 31

003,245	032	1259	LDAX	D	
003,246	312	1260	JZ	D0D	IF MEMORY, DECODE BYTE VALUE
		1261			
		1262	*		IS REGISTER. SET REGISTER NAME.
		1263			
003,251	066	377	MVI	M,377H	CLEAR DIGIT
003,253	341	1264	POP	H	
003,254	042	040	SHLD	DLEDS+1	
003,257	311	1267	RET		

FAM/B - HB FRONT PANEL MONITOR \$01.00.00. HEATH XBASH VI.1 06/21/77
 RCK - READ CONSOLE KEYPAD. 15:44:39 01-APR-77 PAGE 32

1271 ** RCK - READ CONSOLE KEYPAD.
 1272 *
 1273 * RCK IS CALLED TO READ A KEYSTROKE FROM THE CONSOLE KEYPAD.
 1274 * WHENEVER A KEY IS ACCEPTED.
 1275 * RCK PERFORMS DEBOUNCING, AND AUTO-REPEAT. A *BIP* IS SOUNDED
 1276 * WHEN A VALUE IS ACCEPTED.
 1277 *

KEY PAD VALUES:

1278 * 1111 1110 - 0
 1279 * 1111 1100 - 1
 1281 * 1111 1010 - 2
 1282 * 1111 1000 - 3
 1283 * 1111 0110 - 4
 1284 * 1111 0100 - 5
 1285 * 1111 0010 - 6
 1286 * 1111 0000 - 7
 1287 * 1110 1111 - 8
 1288 * 1100 1111 - 9
 1289 * 1010 1111 - 1
 1290 * 1000 1111 - 2
 1291 * 0110 1111 - *
 1292 * 0100 1111 - /
 1293 * 0010 1111 - #
 1294 * 0000 1111 - .
 1295 *
 1296 *
 1297 *

ENTRY NONE
 EXIT TO CALLER WHEN A KEY IS HIT
 (A) = 0 - '0'
 1 - '1'
 2 - '2'
 3 - '3'
 4 - '4'
 5 - '5'
 6 - '6'
 7 - '7'
 8 - '8'
 9 - '9'
 10 - '+'
 11 - '-'
 12 - '*'
 13 - '/'
 14 - '#'
 15 - '.'

1300 *
 1301 *
 1302 *
 1303 *
 1304 *
 1305 *
 1306 *
 1307 *
 1308 *
 1309 *
 1310 *
 1311 *
 1312 *
 1313 *
 1314 *
 1315 *
 1316 *
 1317 *
 1318 *
 1319 RCK EQU *
 1320 PUSH H
 1321 PUSH B
 1322 MVI C,400/20 WAIT.400.MS
 1323 LXI H,RCKA
 1324 IN IP,PAD INFUT PAD VALUE
 1325 RCK1 MOV B,A (B) = VALUE
 1326

003,260
 003,260 345
 003,261 305
 003,262 016.024
 003,264 041 026 040
 003,267 333 360
 003,271 107

FAM/8 - HB FRONT PANEL MONITOR #01.00.00.
 RCK - READ CONSOLE KEYPAD.

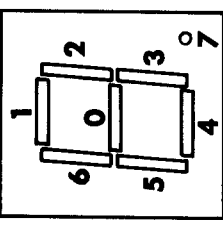
HEATH XBASM V1.1 06/21/77
 15:44:41 01-APR-77 PAGE 33

003.272	074.012	1327	MVI	A120/2	
003.274	315 053 000	1328	CALL	DLY	WAIT 20 MS
003.277	170	1329	MOV	A1B	
003.300	276	1330	CMF	M	
003.301	302 310 003	1331	JNE	RCK2	HAVE A CHANGE
003.304	015	1332	ICR	C	
003.305	302.267.003	1333	JNZ	RCK1	WAIT N CYCLES
		1334			
		1335	*	HAVE KEY VALUE	
		1336			
003.310	167	1337	RCK2		
003.311	356 376	1338	MOV	M1A	UPDATE RCKA
003.313	017	1339	XRI	3760	INVERT ALL BUT GROUP 0 FLAG
003.314	322 326 003	1340	RCC		
003.317	017	1341	JNC	RCK3	HIT BANK 0
003.320	017	1342	RCC		
003.321	017	1343	RCC		
003.322	017	1344	RCC		
003.323	322 267 003	1345	JNC	RCK1	NO HIT AT ALL
003.326	107	1346	MOV	B+A	(B) = CODE
003.327	076.002	1347	MVI	A14/2	
003.331	315 140 002	1348	CALL	HORN	MAKE BIP
003.334	170	1349	MOV	A1B	
003.335	346 017	1350	ANI	170	
003.337	301	1351	POP	B	
003.340	341	1352	POP	H	
003.341	311	1353	RET		RETURN

PAM/B...HB FRONT PANEL MONITOR...\$01.00.00...HEATH XBASM V1.1.06/21/77
 SEGMENT PATTERNS AND CONSTANTS. 15:44:42 01-APR-77 PAGE 34

1357 ** DISPLAY SEGMENT CODING:

1358 * BYTE = 76 543 210
 1359 *
 1360 *
 1361 *
 1362 *
 1363 *
 1364 *
 1365 *
 1366 *



1370 ** REGISTER INDEX TO 7-SEGMENT PATTERN

003.342	244 230	DS	0	1001100010100100B	SP
003.344	220 234	DW	1001100100100000B	AF	
003.346	206 215	DW	10001101100000110B	BC	
003.350	302 214	DW	1000110011000010B	DE	
003.352	222 217	DW	100011110010010B	HL	
003.354	230 316	DW	1100111010011000B	PC	

1380 ** OCTAL TO 7-SEGMENT PATTERN

003.356	001	DS	0	000000001E	0
003.357	163	DE	01110011B	1	
003.360	110	DE	01001000B	2	
003.361	140	DE	01100000B	3	
003.362	062	DE	00110010B	4	
003.363	044	DE	00100100B	5	
003.364	004	DE	00000100B	6	
003.365	161	DE	01110001B	7	
003.366	000	DE	00000000B	8	
003.367	040	DE	00100000B	9	

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

1395 *					
1396 *					
1397 *					
1398					
1399		ORG	4000A-7		
1400					
003.371					
003.371	001	PRSRM	EQU	*	REFIND
003.372	000			0	CILFLG
003.373	000			0	MFLAG



PAM/8 - 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	DSFMDJ
003.375	000	1406	DB	0	ISPROT
003.376	012	1407	DB	10	REGI
003.377	311	1408	DB	MI	MI.RET
000.000		1409			
		1410			ERRNZ *-4000A

FAM/B - HB FRONT PANEL MONITOR *01.00.00.
 RAM CELLS

HEATH XBASM 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.
1415 *
1416
040.000   1417   START   DS   2   40000A   8192   DUMP STARTING ADDRESS
040.000   1418   IOWRK   DS   2
040.002   1419   PRSRAM  EQU   *
040.004   1420   DS      DS   1
040.004   1421   DS      DS   1
040.004   1422
040.005   1423   REGI    DS   1
040.006   1424   DSPROT  DS   1
040.007   1425   DSPMOD  DS   1
040.010   1426   MFLAG   DS   1
040.010   1427   *
040.011   1428   *
040.011   1429
040.011   1430   CTLFLG  DS   1
040.012   1431   REFIND  DS   1
000.007   1432   PRSL    EQU   *-FRSRAM
040.013   1433   FLEDS   EQU   *
040.013   1434   DS      DS   1
040.013   1435   ALEDS   DS   1
040.014   1436   DS      DS   1
040.015   1437   DS      DS   1
040.015   1438
040.016   1439   DS      DS   1
040.017   1440   DS      DS   1
040.020   1441   DS      DS   1
040.021   1442
040.021   1443   DLEDS   DS   1
040.022   1444   DS      DS   1
040.023   1445   DS      DS   1
040.024   1446
040.024   1447   ABUSS   DS   2
040.026   1448   RCKA    DS   1
040.027   1449   CRCSUM  DS   2
040.031   1450   TPERRX  DS   2
040.033   1451   TICCNT  DS   2
040.035   1452
040.035   1453   REGPTR  DS   2
040.037   1454
040.037   1455   UIVEC   DS   0
040.037   1456   DS      DS   3
040.042   1457   DS      DS   3
040.045   1458   DS      DS   3
040.050   1459   DS      DS   3
040.053   1460   DS      DS   3
040.056   1461   DS      DS   3
040.061   1462   DS      DS   3
040.064   1463
040.064   1464   END
    
```

ASSEMBLY COMPLETE.

1464 STATEMENTS

0 ERRORS DETECTED

22310 BYTES FREE

IOA	003062	565	651	1134L
IOB	003066	545	1134	1149L
IOB1	003070	1150L	1164	
IDMRK	040002	666	667	1419L
IP.PAD	000360	81E	439	922
IP.TPC	000371	85E	953	951 1325
IP.TPD	000370	87E	1023	
LAST	001150	528	625L	
LOA0	001272	726L	769	
LOA1	001342	752L	759	
LOAD	001267	724E		
LRA	003047	560	1119L	1234
LRA	003052	427	743	813 1120L
LST2	001154	631L		
MEMM	001165	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	506	520E	
NEXT	001132	527	604L	
UP:CTL	000360	82E	689	698
OF.DIG	000360	83E	400	
OF.SEG	000361	84E	402	
OP.TPC	000371	86E	304	787 845 936 1019 1102
OP.TPD	000370	88E	1104	
OUT	001202	523	662L	
PRSL	000007	191	1432E	
PRSRAM	040004	191	1420E	1432
PRSR0M	003371	190	1401E	
R\$W	001126	530	595L	
RCK	003260	489	580	1150 1319E
RCK1	003267	1325L	1333	1345
RCK2	003310	1331	1337L	
RCK3	003326	1340	1346L	
RCKA	040026	1323	1448L	
REFIND	040012	391	1431L	
REGI	040005	512	586	609 630 1119 1423L
REGM	001104	532	573L	
REGPTR	040035	335	467	1122 1453L
RMEM	001261	525	708L	
RNB	002331	752	979	1004 1018L
RNB1	002335	1020L	1022	
RNP	002325	738	748	888 990 1004L
RT.HF	000002	113E		
RT.CT	000003	114E		
RT.MI	000001	112E	725	797
SAE	001063	554L	605	626 670
SAVALL	000132	200	215	319L
SINCR	004000	281E	283	284

CROSS REFERENCE TABLE XREF V1.0 PAGE 39

SRS	002265	726	975E			
SKS1	002265	976L	984	988		
SKS2	002271	979L	982			
SS1	001235	257	690L			
SSTEP	001225	524	685E			
START	040000	284	750	799	1418L	
STPRIN	001244	218	696E			
TER1	002220	920L	927			
TER3	002215	913L	924			
TFT	002133	768	844L	908		
TICCNT	040033	369	371	406	865	935 1451L
TPABT	002244	708	783	935L		
TPERR	002205	737	906L			
TPERRX	040031	709	784	955	1450L	
TPXIT	002252	921	951L	1020	1098	
UCI*ER	000020	165E	1018	1101		
UCI*IE	000002	167E				
UCI*IR	000100	163E				
UCI*RE	000004	166E	1018			
UCI*RO	000040	164E	1018			
UCI*TE	000001	168E	786	1101		
UFD	003161	409	1215E			
UFD1	003227	1230	1251L			
UIVEC	040037	229	234	239	253	260 348 703 1455L
UMI*16X	000002	158E	303			
UMI*1R	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*CLK	000001	138E	346			
UO*IDU	000002	137E	461	1216		
UO*HLT	000200	135E	420			
UO*NFR	000100	136E	384	461		
USR*FE	000040	172E				
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	526	782E			
WNB	003024	790	794	824	1084	1097L
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

```

040.100 040.100A 40100A RESET PAMB
000.322 EQU 322A MAKE NOISE
002.140 HORN EQU 3140A READ CONSOLE KEYPAD
003.280 RCK EQU 3260A OCTAL TO 7-SEGMENT PATTERN
003.356 DODA EQU 3356A USER FLAG OPTIONS
040.010 .MFLAG EQU 40010A FRONT PANEL L.E.D. PATTERNS
040.013 FPLEDS EQU 40013A USER INTERRUPT VECTOR
040.037 UIVEC EQU 40037A
000.001 UO.CLK EQU 1A ALLOW CLOCK INTERRUPT PROCESSING
000.002 UO.DDU EQU 2A DISABLE DISPLAY UPDATE
000.303 MI.JMP EQU 303A MACHINE INSTRUCTION (8080) JUMP
000.377 LEDOFF EQU 377A BLANK L.E.D. DISPLAY
*** *****

```

* * DISABLE UPDATING OF L.E.D. DISPLAY

* * AND TURN OFF L.E.D.'S

```

* *
* *
* *
040.100 040.100 MVI A,UO.DDU DISABLE NORMAL UPDATING
040.102 STA .MFLAG DONE
040.105 LXI H,FLEDS L.E.D. DISPLAY ADDRESS
040.110 MVI B,9 COUNT L.E.D.'S
040.112 MVI A,LEDOFF TURN OFF L.E.D.
040.114 MOV M,A O.K. - GO
040.115 INX B NEXT L.E.D. ADDRESS
040.116 DCR B ALL DONE - ??
040.117 JNZ BLANK NO - DO AGAIN!
*** *****

```

* * READ A DECIMAL INTEGER FROM H8 FRONT PANEL

* * IF NOT DECIMAL -- RETURN TO PAM-8.

* * INCREMENT THE INTEGER (A PROGRAM REQUIREMENT)

* * STORE THE DIGIT.

```

* *
040.122 040.122 CALL RCK READ CONSOLE KEYPAD
040.125 040.125 CPI JOP TEST IF ZERO THRU NINE
040.127 040.127 JNC ERROR ABORT TO PAM-8
040.132 040.132 INR A <A>=<A>+1
040.133 040.133 STA DIGIT STORE INTEGER
*** *****

```

* * INITIALIZE CLOCK COUNTER.

* * PROGRAM REQUIRES ONE INTERRUPT BEFORE DISPLAY

```

* *
040.136 040.136 LXI H,1 H=0 & L=1
040.141 040.141 SHLD TICK INITIALIZE COUNT
*** *****

```

* * 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!



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

- Decrements 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

2

2

2