

**SOFTWARE REFERENCE  
MANUAL**

**H88/WH89 COMPUTER**

**MONITOR**

**MTR-88**



## Table of Contents

Introduction .....	4
Theory of Operation .....	5
Displaying and Altering Memory .....	7
Program Execution Control .....	9
Load/Dump Routines .....	11
Advanced Control .....	14
Floppy Boot .....	16
Appendix A .....	17
MTR-88 Listing .....	17
Appendix B .....	71
MTR-88 Demo .....	71
The Sample Program .....	72
Appendix C .....	75
Octal Definitions .....	75

## INTRODUCTION

This manual describes the functions and operation of the H88 Monitor Program MTR-88 that is contained in a read-only memory (ROM) that is in your H88/H89. Some of the major features of MTR-88 are:

- Memory contents display and alteration.
- Program execution control.
- Cassette load and dump routines.
- Floppy diskette boot-strap routine.

In addition, MTR-88 can be instructed (by means of a flag byte maintained in read/write memory) to bypass some or all of its normal functions. In this manner, a sophisticated user can augment or replace these functions.

Since the H89 is an expansion of the H88, all the features of MTR-88 are available on an H89 also.

## THEORY OF OPERATION

This section supplements the information in the "Operations" and "Circuit Description" sections of your H88 Operations Manual. In order to use all of the features of MTR-88, it is necessary to understand the Z80 operation codes and the circuit of your H88. This section gives you details of the operation of MTR-88. The listing of MTR-88 is given in Appendix A and a sophisticated example is given in Appendix B.

### Power Up and Reset

MTR-88 initializes the H88 whenever you power-up or RESET. To power-up, use the switch on the back of the H88. To RESET, simultaneously press the RESET key and the right-hand SHIFT key on the keyboard. Both power-up and RESET cause a level zero interrupt (highest priority). MTR-88 sounds the audio alert and resets to its normal state. During the initialization procedure, MTR-88 determines the high limit of continuous RAM in your H88. Once this high limit has been determined, the Z80's stack pointer is set to this value. Then MTR-88 enters a loop waiting for you to enter a command.

### Clock Interrupts

The Clock Interrupt is a crucial element in the operation of the H88. It is a level one interrupt and is generated on the H88 CPU board every 2 ms (millisecond). MTR-88 maintains "TICCNT" which counts up one every 2 ms. See the listing in Appendix A for the location of TICCNT.

Note that MTR-88 uses interrupts, so you should not disable interrupts for a long period of time. MTR-88 also requires a stack pointer at the top of memory with at least 80 bytes.

## General Operations

When you RESET or power-up your H88 or H89, MTR-88 responds by clearing the screen and displaying "H:". This tells you that it is ready to respond to your typed commands. When you type in something, MTR-88 will either accept it or give a beep, indicating an error.

If the letter you enter is the first letter of one of MTR-88's commands, it will display the remaining letters of the word and start the appropriate program in MTR-88. If the letter is not the start of a command, MTR-88 will sound the horn and re-display the "H:".

The DELETE key will kill a partially entered line and cause MTR-88 to return to the "H:" prompt. You can use this to correct typing errors.

NOTE: In this manual, the symbol "Δ" means a space and "Ⓢ" means a RETURN.

The following is a list of the acceptable MTR-88 commands. You type the first letter of the command, and MTR-88 will supply the remainder of the word. You have to press the (carriage) RETURN key before MTR-88 will respond.

### TABLE OF MTR-88 COMMANDS

Boot	— Boot HDOS from a diskette
Dump	— Dump a program to cassette
Go	— Start a program
Load	— Load a program from cassette
Program Counter	— Set an address in the PC
Substitute	— Inspect or change memory

These commands are described more in the remainder of this manual.

## DISPLAYING AND ALTERING MEMORY

One of the major features of MTR-88 is its ability to examine the contents of any H88 memory location and to modify the contents of that location if it is in RAM. This feature is described now.

The Substitute command is used to display memory locations. After a memory location has been displayed, its value can be changed before you proceed to something else. There is an example showing the Substitute procedure at the end of the description. You may jump ahead to it at any time.

To start the substitution process, first type "S". MTR-88 will respond by completing the word "Substitute". You should then enter the address of the memory location you want to inspect, followed by a RETURN. This address **must** be given in split-octal. Refer to Appendix C for the definitions of octal and split-octal.

MTR-88 will respond by re-displaying the address. Following the address, MTR-88 will display the contents of that memory location in octal.

Once the value of the memory location has been displayed, you may change it. To change it, simply type in the new value (in octal). The new value will be inserted after you complete the next step.

**NOTE:** MTR-88 will use the last three digits that you enter. That is, the entry "12345" will be entered as "345". You may use this to correct errors as entries are made.

After you have inspected or changed the value of a memory location, you have three options. First, you can cause MTR-88 to advance to the next memory location and display it by pressing the Space Bar. Second, you can cause MTR-88 to retreat to the previous memory location and display it by pressing the minus key, "-". Finally, you can cause MTR-88 return to its initial "H:" by pressing the RETURN key.

The following example shows these features. To help you follow what you enter and what the computer responds, your entries and the computer's responses are shown on different lines. If a new line is really used, the new line will start at the left of the page. Otherwise, the output is shown just down a line.

**EXAMPLE**

H:			computer
	S		you
	ubstitute		computer
		2146 ☺	you
002146	041		computer
		Δ	you
002147	011		computer
		Δ	you
002150	040		computer
		-	you
002147	011		computer
		☺	you
H:			computer
	S		you
	ubstitute		computer
		40100 ☺	you
040100	xxx		computer
		123 Δ	you
040101	xxx		computer
		-	you
040100	123		computer
		☺	you
H:			computer



## PROGRAM EXECUTION CONTROL

MTR-88 allows you to start a program that you have loaded into memory. It also offers a form of breakpointing.

The standard way of starting a program is to use the Go command. After you type in "G", MTR-88 responds "o". You should then type in the address (in split octal) where you want execution of your program to start. For example, if you have loaded a program at 040.100, you can start it with:

```
H: Go 40100 Ⓢ
```

MTR-88 allows another method of starting programs. MTR-88 maintains in its working memory a value for the Program Counter. If you enter "G" and then a RETURN after MTR-88 prints "o", MTR-88 will use the value in the PC as the starting address of your program.

To set the value in the Program Counter, you use the "P" command. After you enter "P", MTR-88 will respond "rogram Counter" and you can then enter the value you want. For example:

```
H: Program Counter 40100 Ⓢ  
H: Go Ⓢ
```

Your program will now be started at 40100.

If you do not enter a number after "P", but simply press RETURN, then MTR-88 will display the current value of the PC on the next line. You can change the PC by typing in a new value or you can leave it un-altered by pressing RETURN. For example:

```
H: Program Counter Ⓢ  
277377 40100 Ⓢ
```

(You type the second number.)

When you are debugging an assembly language program, you can use MTR-88 to set breakpoints at various places in the program. To set a breakpoint, use the Substitute command and put an HLT (166 octal) instruction where you want your program to stop.

When your program reaches the breakpoint HLT instruction, it will return to MTR-88, display an "H", and then advance to a new line and display "H:". You can now inspect or change memory using the "Substitute" command.

To continue your program, you will first have to restore the byte in the location where you placed the breakpoint HLT. Since the computer had to execute the HLT instruction, the PC will point one beyond where you placed the HLT. To continue, you will have to decrease the PC value by one.

Do this by entering the "P" command and a RETURN. When the current value of the PC is shown, subtract one from it, and enter this value as the new value for the PC. Remember that you have to subtract in octal, so ten minus one is seven!

Alternatively, you can use the "Go" command to start the program from whatever address you want, including from the place where you put the HLT.

Note that if the program that you are debugging uses keyboard interrupts, MTR-88 and your program will "fight" for keyboard input! Your program will always see every character because it gets them by an interrupt. MTR-88 is continually testing if a character is available, and it will never see some of the characters that you enter. This can become very confusing, particularly in the debug program.

## LOAD/DUMP ROUTINES

MTR-88 contains routines that let you load and dump memory contents from or to a cassette tape. These "boot strap" routines allow you to quickly and easily use your computer without entering a complex program by hand. These routines contain sophisticated error checking techniques that let you know if a problem has occurred.

### Loading From Tape

To load from a cassette tape, ready the tape reader with the tape to be loaded. Then RESET MTR-88 and type "L". MTR-88 will respond "oad". When you enter a RETURN, MTR-88 will load the tape.

No change will be seen on the screen until MTR-88 finds the first file on the tape. The load routine places the entry point address into the H88's Program Counter and then continues loading. As data is loaded into memory, the address at which it is placed is shown on the screen. You can watch it change and see when the load is complete.

If the load is successful, MTR-88 will sound the alarm once to alert you. If a loading error occurs, MTR-88 will sound the alarm repeatedly. RESET the H88 and try loading the tape again.

You may RESET during loading, but the load will be invalid. To get a good load, you will have to start the procedure over.

### Dumping a Tape

Before MTR-88 can dump a tape, it needs to have the first and last address of the section of memory that you want to have dumped. It also needs to have the starting address (which need not be the first address) so that when the program is loaded, the PC can be set, and "Go" can be used to start the program.

First, place the address of the program's starting address in the PC as described earlier. Later, when you load the tape, this value will be placed back in the PC so you can enter "Go" and start the program.

Next, give a "Dump" command by typing "D". MTR-88 will respond with "ump". You should then type in the first address you want saved on the tape, followed by the minus or dash character (-), and finally the last address you want saved. When you press the RETURN, MTR-88 will start recording the data on the tape.

The detail steps to DUMP to a tape are:

1. Ready a tape in the recorder and press the record button.
2. Enter the starting address of the program in the PC.
3. Enter the dump first and last addresses as:

Dump 40100-43264

4. Press RETURN.

If you give a starting address, you must give an ending address separated by a dash. However, if you do not give a starting address, MTR-88 will use the starting and finishing address that were last used for a Load or a Dump. This is described next in Copying a Tape.

## Copying a Tape

To copy a tape, simply load the tape as described in "Loading From Tape". Then ready the dump tape to receive the copy. Finally, type "D". When MTR-88 responds "ump", simply press RETURN. MTR-88 will remember the first, last, and starting addresses from the load.

You can modify the program before you dump it, but if the first or last addresses are different, you will have to enter them when you dump. The same is true for the PC.

## Tape Errors

MTR-88 detects two types of tape errors: record errors and checksum errors. In either case, when an error is detected, the tape transport stops and an error number is printed on the screen. The error numbers are 001 for a checksum error and 002 for a record error. The alarm is repeatedly sounded when an error is detected. RESET the H88 to stop the alarm and return to MTR-88's command mode.

## 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 file.
- A read error that causes a portion of the data to be lost, and records are not read in the proper sequence.

## Checksum Errors

A checksum error occurs when the Cyclical Redundancy Check (CRC) checksum that follows a record does not match the CRC calculated by MTR-88. This error means that the record is either recorded incorrectly or the load was faulty. In either case, the load should be tried again. If repeated loads result in repeated failures, the tape is probably defective.

## ADVANCED CONTROL

One of the advanced features of MTR-88 is its provisions allowing sophisticated users to augment or replace MTR-88's functions. This is usually done in conjunction with assembly language programs, although it is sometimes possible to use these features in BASIC using the PEEK and POKE commands. The sample program in Appendix B shows how to use several of MTR-88's advanced features.

The following discussion refers to symbols and locations in MTR-88. In order to make the most of this information, you should refer to the listing of MTR-88 that is in Appendix A. Note that at the end of the listing the definitions of RAM locations from 40.000 to 40.077 are given. Following these is a symbol reference table that will help you find where symbols are used in the program.

### The Tick Counter (TICCNT)

MTR-88 maintains in memory a 16-bit (2 byte) tick counter named TICCNT. This counter is incremented when the clock interrupts occur. As long as interrupts are enabled, this will occur every 2 ms. You may set TICCNT to any value and change it as often as you like. The low-order byte of TICCNT is in location 40.033 (8219 decimal) and the high-order byte is in 40.034.

### Using Interrupts

All H88 interrupts cause control to be transferred into the lowest 64 bytes of memory. Since MTR-88 occupies this area, it processes all interrupts first. Except for level zero interrupts (RESET function), you can supply a routine to process interrupts yourself.

Control is passed out of MTR-88 through the UIVECs that are located at 40.037 and following. Each vector is three bytes long, and contains a JMP instruction to an interrupt processing routine. MTR-88 calls or jumps to the appropriate UIVEC, and control is passed to the processing routine. The exit from an interrupt processing routine should be the return instruction, RET.

## I/O Interrupts

Interrupts numbered 3 through 7 are I/O interrupts of devices that you connect to your H88. MTR-88 does not process these interrupts, but simply passes them on to a program in RAM by jumping to the appropriate UIVEC.

All Heath software (except MTR-88) uses interrupt 3 for input and output to and from the keyboard and screen. These programs set UIVEC themselves. If you want to use interrupts, your program has to place the appropriate jump in the appropriate UIVEC. See the sample program in Appendix B.

## Clock Interrupts

The level one interrupt is generated by hardware in your H88 every 2 ms. MTR-88 always processes these interrupts, but you can force it to pass control to your routine once it is done.

To do this, set the appropriate jump in the first UIVEC locations. Then set the UO.CLK bit (001) in .MFLAG (40.010). MTR-88 will then pass each clock interrupt to your routine when it finishes its own processing. This is done in the example in Appendix B.

## Single Instructions and Breakpoint Interrupts

Level two interrupts are generated by the single-instruction hardware contained in the H88. When a single-instruction interrupt occurs, MTR-88 processes it, and jumps to the location specified by the second UIVEC. This interrupt has no effect on MTR-88.

If you have set up UIVEC for level two interrupts, you can use RST-2 as a breakpoint instruction. Control will be returned to the location specified by the second UIVEC. These features are used by the DEBUG programs supplied by Heath.

## FLOPPY BOOT

MTR-88 contains the code necessary to boot-up HDOS from a floppy disk. If you enter "B" after the "H:" prompt, then MTR-88 will respond "oot". When you then press RETURN, MTR-88 will jump to location 30.000 which is the entry point for the HDOS boot-up routine.

Unless you have the floppy disk controller board installed in your H88, there will be no ROM at 30.000, and the results of the "B" command are unpredictable. If you perform a "B" command, and do not have a floppy interface card, you should RESET your H88 to put it back in a known state.



## APPENDIX A

### MTR-88 LISTING

This appendix contains a listing of MTR-88. MTR-88 resides in the low 2K (2048) bytes of the H88 or H89 computer's memory. It contains all the control for primitive keyboard input and screen output as well as cassette tape load and dump facilities. MTR-88 needs RAM locations available in locations 40.000 through 40.077, and it also needs 80 bytes of stack area in high memory.

The first few pages of the listing show definitions that are used. The last portion of the listing contains references to the symbols that are used in MTR-88. Just before this cross reference listing is the definition of RAM locations in 40.000 through 40.077.

Note that most of the PAM-8 entry points are preserved in MTR-88. (PAM-8 is the equivalent of MTR-88 on the H8 computer.) This was done to allow compatibility between H8 and H88 programs. Of course, H8 front panel routines will not operate, but they will return properly.

Because PAM-8 entry points have been preserved, the MTR-88 code has to jump around in a somewhat arbitrary manner. Also, the Memory Test and Floppy Disk Rotational Speed Test routines are scattered throughout memory. The listing of these two routines are not shown. The Memory Test entry point is 7.375 and the Floppy Speed Test entry point is 7.372.

```

4 *** MTR88 - H88 MONITOR ISSUE 09.00.00
5 *
6 * MTR88 IS AN ADAPTATION OF PAM/B ORIGINALLY WRITTEN FOR THE
7 * HEATH HB COMPUTER BY J. G. LETWIN IN 1976 AND MODIFIED BY
8 * R. N. BORCHARDT IN 1979 FOR USE IN THE HEATH H88/H89
9 * COMPUTERS.
10 *
11 * MTR88 PROVIDES COMPATIBILITY WITH PAM/B SUCH THAT ALL ROUTINES
12 * HAVE RETAINED PREVIOUSLY DESCRIBED ENTRY POINTS AND ENTRY AND
13 * EXIT CONDITIONS. ROUTINES WHICH ARE NOT APPLICABLE SUCH AS
14 * THOSE PERTAINING TO THE FRONT PANEL DISPLAY HAVE BEEN DELETED.
15 *
16 *
17 * COPYRIGHT 05/1976, WINTEK CORPORATION,
18 * 902 N. 9TH ST.
19 * LAFAYETTE, IND.
20 *
21 * COPYRIGHT 01/1979, HEATH COMPANY
22 * BENTON HARBOR, MI.
23 *
24 *
25 ROMDD EQU 30000A HDOS ROOT ROM ADDRESS
26
27
28

```

```

30 *** MTR88 - H88/H89 MONITOR.
31 *
32 * THIS PROGRAM RESIDES (IN ROM) IN THE LOW 2048 BYTES OF THE HEATH
33 * H88/H89 COMPUTERS.

```

```

35 *** INTERRUPTS.
36 *
37 * MTR88 IS THE PRIMARY PROCESSOR FOR ALL INTERRUPTS.
38 * THEY ARE PROCESSED AS FOLLOWS:
39 *
40 * RST USE
41 *
42 * 0 MASTER CLEAR, (NEVER USED FOR I/O OR RST)
43 *
44 * 1 CLOCK INTERRUPT, NORMALLY TAKEN BY MTR88,
45 * SETTING BIT #00 CLK* IN BYTE #.MELAG* ALLOWS
46 * USER PROCESSING (VIA A JUMP THROUGH #UIVEEC*),
47 * UPON ENTRY OF THE USER ROUTINE, THE STACK
48 * CONTAINS:
49 * (STACK+0) = RETURN ADDRESS (TO MTR88)
50 * (STACK+2) = (STACK+14)
51 * (STACK+4) = (AF)
52 * (STACK+6) = (BC)
53 * (STACK+8) = (DE)
54 * (STACK+10) = (HL)

```

```

55 * (STACK+12) = (PC)
56 * THE USER'S ROUTINE SHOULD RETURN TO MTR88 VIA
57 * A *RET* WITHOUT ENABLING INTERRUPTS.
58 *
59 * 2 SINGLE STEP INTERRUPTS RECEIVED WHEN IN
60 * USER MODE CAUSES A JUMP THROUGH *UIVEC**3.
61 * STACK UPON USER ROUTINE ENTRY:
62 * (STACK+0) = (STACKPRT+12)
63 * (STACK+2) = (AF)
64 * (STACK+4) = (BC)
65 * (STACK+6) = (DE)
66 * (STACK+8) = (HL)
67 * (STACK+10) = (PC)
68 * THE USER'S ROUTINE SHOULD HANDLE IT'S OWN RETURN
69 * FROM THE INTERRUPT. THAT IS, *EI* FOLLOWED BY *RET*.
70 *

```

```

71 * THE FOLLOWING INTERRUPTS ARE VECTORED DIRECTLY THROUGH *UIVEC**,
72 * THE USER ROUTINE MUST HAVE SETUP A JUMP IN *UIVEC* BEFORE ANY
73 * OF THESE INTERRUPTS MAY OCCUR. RETURN IS VIA *EI* AND THEN *RET*
74 *
75 * 3 I/O 3. CAUSES A DIRECT JUMP THROUGH *UIVEC**+6
76 *
77 * 4 I/O 4. CAUSES A DIRECT JUMP THROUGH *UIVEC**+9
78 *
79 * 5 I/O 5. CAUSES A DIRECT JUMP THROUGH *UIVEC**+12
80 *
81 * 6 I/O 6. CAUSES A DIRECT JUMP THROUGH *UIVEC**+15
82 *
83 * 7 I/O 7. CAUSES A DIRECT JUMP THROUGH *UIVEC**+18
84 *

```

87. \*\* ASSEMBLY CONSTANTS.

89 \*\* IO PORTS.

90  
 91 \*\*\* ALL REFERENCES TO THE HB FRONT PANEL PORTS ARE TRAPPED BY THE  
 92 \* Z80 NMI OF THE H88/H89. OP.CTL WILL STILL PERFORM AS IN AN HB  
 93 \* IN RESPECT TO THE CLOCK AND SINGLE STEP CONTROL. FOR MORE  
 94 \* INFORMATION SEE THE NMI ROUTINE.  
 95  
 96 IF.PAD EQU 3600 PAD INPUT PORT  
 97 OP.CTL EQU 3600 CONTROL OUTPUT PORT  
 98 OP.DIG EQU 3600 DIGIT SELECT OUTPUT PORT  
 99 OP.SEG EQU 3610 SEGMENT SELECT OUTPUT PORT  
 100  
 101 \* H88/H89 CONTROL PORT  
 102 H88.CTL EQU 3620 H88/H89 PORT FOR CLOCK AND SINGLE STEP  
 103 H88B.CK EQU 00000010B 2MS CLOCK ENABLE/DISABLE  
 104 H88B.SS EQU 000000001B SINGLE STEP ENABLE/DISABLE  
 105  
 106 H88.SW EQU 3620 8 POSITION DIP SWITCH  
 107 H88S.BR EQU 11000000B BAUD RATE SWITCHES  
 108 H88S.M EQU 00100000B MEMORY TEST/NORMAL OPERATION SWITCH

110 \*\* CASSETTE PORTS.

111  
 112 IF.TPC EQU 3710 TAPE CONTROL IN  
 113 OP.TPC EQU 3710 TAPE CONTROL OUT  
 114 IF.TPD EQU 3700 TAPE DATA IN  
 115 OP.TPD EQU 3700 TAPE DATA OUT

117 \*\* ASCII CHARACTERS.

118  
 119 A.SYN EQU 0260 SYNC CHARACTER  
 120 A.STX EQU 0020 STX CHARACTER  
 121 A.BEL EQU 0070 BELL CHARACTER  
 122 A.BKS EQU 0100 BACKSPACE CHARACTER  
 123 A.LF EQU 0120 LINE FEED CHARACTER  
 124 A.CR EQU 0150 CARRIAGE RETURN CHARACTER  
 125 A.ESC EQU 0330 ESCAPE CHARACTER  
 126 A.DEL EQU 1770 DELETE OR RUBOUT CHARACTER

128 \*\* FRONT PANEL HARDWARE CONTROL BITS.  
 129                    SINGLE STEP INTERRUPT  
 000.020 EQU 00010000B  
 000.040 EQU 00100000B  
 131 CB.HTL EQU 01000000B  
 132 CB.CLI EQU 10000000B  
 133 CB.SPK EQU 10000000B  
                   CLOCK INTERRUPT ENABLE  
                   SPEAKER ENABLE

135 \*\* DISPLAY MODE FLAGS (IN \*DSPMOD\*)  
 136 DM.MR EQU 0 MEMORY READ  
 137 DM.MW EQU 1 MEMORY WRITE  
 000.001 EQU 0 REGISTER READ  
 000.002 EQU 1 REGISTER WRITE  
 140 DM.RW EQU 3  
 000.003 EQU 3  
 000.000 XTEXT TAPE  
                   TAPE DEFINITIONS

163 \*\* MACHINE INSTRUCTIONS.  
 164 MI.HLT EQU 01110110B HALT  
 000.166 MI.RET EQU 11001001B RETURN  
 000.311 MI.IN EQU 11011011B INPUT  
 000.333 MI.OUT EQU 11010011B OUTPUT  
 000.323 MI.LDA EQU 00111010B LDA  
 000.072 MI.ANI EQU 11100110B ANI  
 000.346 MI.LXID EQU 00010001B LXI D  
 000.021 MI.JMP EQU 11000011B JMP  
 000.303 MI.LDXA EQU 11011101B LD IX, (BYTE A)  
 000.335 MI.LDYB EQU 00100001B LD IX, (BYTE B)  
 000.041 MI.LDYA EQU 11111101B LD IY, (BYTE A)  
 000.375 MI.LDYB EQU 00100001B LD IY, (BYTE B)  
 000.010 MI.EXAF EQU 00001000B EX AF,AF  
 000.010 MI.JIXA EQU 11011101B JP (IX) (BYTE A)  
 000.335 MI.JIXB EQU 11010011B JP (IX) (BYTE B)  
 000.351 MI.JIYA EQU 11111101B JP (IY) (BYTE A)  
 000.375 MI.JIYB EQU 11101001B JP (IY) (BYTE B)  
 000.351

183 \*\* USER OPTION BITS.  
 184 \*                    THESE BITS ARE SET IN CELL .MFLAG.  
 185 \*  
 186 UO.HLT EQU 10000000B DISABLE HALT PROCESSING  
 000.200 UO.NFR EQU CB.CLI NO REFRESH OF FRONT PANEL  
 000.100 UO.DDU EQU 00000010B DISABLE DISPLAY UPDATE  
 000.002 UO.CLK EQU 00000001B ALLOW PRIVATE INTERRUPT PROCESSING  
 000.001



HEATH HBASM V1.4 01/20/78 PAGE 6  
 08:59:11 17-MAY-79

MTR88...HB8 MONITOR \$02.00.00.  
 HARDWARE INTERRUPT VECTORS

INTERRUPT VECTORS:

303.\*\*\*  
 304.\*  
 305

307.\*\* LEVEL 0 - RESET  
 308.\* THIS INTERRUPT MAY NOT BE PROCESSED BY A USER PROGRAM.  
 309.\*  
 310  
 311  
 312 INIT0 JMP INIT0X DO HB8 EXTENSION OF INITIALIZATION  
 313 INIT0.0 LXI H,FRAMFRSL-1 (HL) = RAM DESTINATION FOR CODE  
 314 JMP INIT INITIALIZE  
 315 ERRPL INIT-1000A BYTE IN WORD 10A MUST BE 0  
 316  
 317

LEVEL 1 - CLOCK

319.\*\*  
 320 INT1 EQU 100 INTERRUPT ENTRY POINT  
 321  
 322 ERNZ \*-110 INTO TAKES UP ONE BYTE  
 323  
 324 CALL SAVALL SAVE USER REGISTERS  
 325 MVI D,0 PROCESS CLOCK INTERRUPT  
 326 JMP CLOCK EXTRA BYTE MUST BE 0  
 327 ERRPL CLOCK-1000A  
 328  
 377.201

LEVEL 2 - SINGLE STEP

330.\*\*  
 331.\* IF THIS INTERRUPT IS RECEIVED WHEN NOT IN MONITOR MODE,  
 332.\* THEN IT IS ASSUMED TO BE GENERATED BY A USER PROGRAM.  
 333.\* (SINGLE STEPPING OR BREAKPOINTING). IN SUCH CASE, THE  
 334.\* USER PROGRAM IS ENTERED THROUGH JUMPER3  
 335.\*  
 336  
 337 INT2 EQU 20A LEVEL 2 ENTRY  
 338 ERNZ \*-21A INT1 TAKES EXTRA BYTE  
 339  
 340  
 341 CALL SAVALL SAVE REGISTERS  
 342 LDAX D C,TLFLG (A) = (CTLFLG)  
 343 SET STPRTN STEP RETURN  
 344 JMP

```

346 *** I/O INTERRUPT VECTORS.
347 *
348 * INTERRUPTS 3 THROUGH 7 ARE AVAILABLE FOR GENERAL I/O USE.
349 *
350 * THESE INTERRUPTS ARE NOT SUPPORTED BY MTR88, AND SHOULD
351 * NEVER OCCUR UNLESS THE USER HAS SUPPLIED HANDLER ROUTINES
352 * (THROUGH UIVEC)
353 *
000.030     ORG     30A
000.030     JMP     UIVEC+6     JUMP TO USER ROUTINE
000.033     DB     '44440'     HEATH PART NUMBER 444-40
.....
360     ORG     40A
361
362     INT4     JMP     UIVEC+9     JUMP TO USER ROUTINE
000.040     DB     440,1220,1160,1020,440 SUPPORT CODE
.....
000.050     ORG     50A
000.050     JMP     UIVEC+12     JUMP TO USER ROUTINE
.....
372 **     DLY - DELAY TIME INTERVAL.
373 *
374 *     ENTRY (A) = MILLISECOND DELAY COUNT/2
375 *     EXIT NONE
376 *     USES A/F
377 *
000.000     ERNZ     *-53A
.....
000.053     PUSH     PSW     SAVE COUNT
000.054     XRA     A     DONT SOUND HORN
000.055     JMP     HRNO     PROCESS AS HORN
.....
000.060     ORG     60A
000.060     JMP     UIVEC+15     JUMP TO USER ROUTINE
.....
000.063     MVI     A,CB,SS1+CB,CLI+CB,SPK OFF MONITOR MODE LIGHT
000.065     JMP     SS1     RETURN TO USER PROGRAM
.....

```





```

400 ** INIT - INITIALIZE SYSTEM.
401 *
402 * INIT IS CALLED WHENEVER A HARDWARE MASTER-CLEAR IS INITIATED.
403 *
404 * SETUP MTR88 CONTROL CELLS IN RAM.
405 * DECODE HOW MUCH MEMORY EXISTS, SETUP STACKPOINTER, AND
406 * ENTER THE MONITOR LOOP.
407 *
408 * ENTRY FROM MASTER CLEAR
409 * EXIT INTO MTR88 MAIN LOOP
410
411 ERRNZ *-730
412
413 INIT
414   LDAX D COPY #PRSRON* INTO RAM
415   MOV M,A MOVE BYTE
416   DCX H DECREMENT DESTINATION
417   INR E INCREMENT SOURCE
418   JNZ INIT IF NOT DONE
419   SINC R SEARCH INCREMENT
420
421   MOV D,SINCR/256 (DE) = SEARCH INCREMENT
422   LXI H,START-SINCR (HL) = FIRST RAM - SEARCH INCREMENT
423
424 * DETERMINE MEMORY LIMIT.
425
426   MOV M,A RESTORE VALUE READ
427   DAD D INCREMENT TRIAL ADDRESS
428   MOV A,M (A) = CURRENT MEMORY VALUE
429   DCX M TRY TO CHANGE IT
430   CMP M
431   JNE INITI IF MEMORY CHANGED
432
433   INIT2 DCX H
434
435   SPHL SET STACKPOINTER = MEMORY LIMIT -1
436
437   PUSH H SET #PC* VALUE ON STACK
438   LXI H,ERROR
439   PUSH H SET 'RETURN ADDRESS'
440
441 * CONFIGURE LOAD/DUMP UART
442
443   MVI A,UNI.18+UNI.18+UNI.16X
444   OUT OP.TFC SET 8 BIT, NO PARITY, 1 STOP, XI6
    
```

MTRBB - HBB MONITOR \$09.00.00. HEATH HBASM V1.4 01/20/78 PAGE 10  
 INTERRUPT TIME SUBROUTINES 08:59:13 17-MAY-79

```

447 ** SAVALL - SAVE ALL REGISTERS ON STACK.
448 *
449 * SAVALL IS CALLED WHEN AN INTERRUPT IS ACCEPTED, IN ORDER TO
450 * SAVE THE CONTENTS OF THE REGISTERS ON THE STACK.
451 * ENTRY - CALLED DIRECTLY FROM INTERRUPT ROUTINE.
452 * ALL REGISTERS PUSHED ON STACK.
453 * IF NOT YET IN MONITOR MODE; REGPTR = ADDRESS OF REGISTERS
454 * ON STACK.
455 * (DE) = ADDRESS OF CTLFLG
456 *
000.000 ERRNZ *-132A
457
458
459
460 SAVALL XTHL SET H,L ON STACK TOP
461 PUSH D
462 PUSH B
463 PUSH PSW (D,E) = RETURN ADDRESS
464 XCHG LXI H,10 (H,L) = ADDRESS OF USERS' SP
465 DAD SP
466
467 ** REPLACE THESE INSTRUCTIONS WITH A JUMP AROUND THE NMI VECTOR JUMP
468 *
469 *
470 * PUSH H SET ON STACK AS REGISTER
471 * PUSH D SET RETURN ADDRESS
472 * LXI D,CTLFLG
473 * LDAX D (A) = CTLFLG
474
000.143 JMP SAVALLX GO TO SAVALL EXTENSION
475
476
477 ** ENTRY POINT FOR THE Z80 NMI
478 *
479
480
481 ERRNZ *-66H Z80 NMI ADDRESS
482
000.000
483 NMIENT JMP NMI
484
000.146 JMP NMI
485
000.000 ERRNZ SAVALLR-151A DO NOT CHANGE ORGANIZATION
486
000.151 SAVALLR EQU * SAVALL EXTENSION RETURN ADDRESS
487
000.151 057 CMA
488 000.152 346 060 ANI
489 000.154 310 RZ
490 000.155 041 002 000 LXI H,2
491 000.160 071 DAD SP
492 000.161 042 035 040 SHLD REGPTR
493 000.164 311 RET
494
495

```

MTR00 - H08 MONITOR #09.00.00. INTERRUPT TIME SUBROUTINES

```

497 **      CUI - CHECK FOR USER INTERRUPT PROCESSING,
498 *
499 *      CUI IS CALLED TO SEE IF THE USER HAS SPECIFIED PROCESSING
500 *      FOR THE CLOCK INTERRUPT,
501
000.000      ERRNZ *-165A
502
040.010      SET      .MFLAG      REFERENCE TO MFLAG
503
000.165 012  LDAX      B           (A) = .MFLAG
000.000      ERRNZ  UO:CLK-1     CORE ASSUMED = 01
504
000.166 017  RRC           IF SPECIFIED, TRANSFER TO USER
000.167 334 037 040      CC      UIVEC
505
510 *      RETURN TO PROGRAM FROM INTERRUPT.
511
000.000      ERRNZ *-172A
512
000.172 361  INTXIT  POP      PSW      REMOVE FAKE STACK REGISTER
000.173 361  POP      PSW
000.174 301  POP      B
000.175 321  POP      D
000.176 341  POP      H
000.177 373  EI
000.200 311  RET

```



MTR88 - H88 MONITOR #09.00.00. HEATH HBASM V1.4.01/20/78 PAGE 12  
 PROCESS CLOCK INTERRUPTS 08:59:13 17-MAY-79

```

523 ***          CLOCK - PROCESS CLOCK INTERRUPT.....
524 *           CLOCK IS ENTERED WHENEVER A 2-MILLISECOND CLOCK INTERRUPT IS
525 *           PROCESSED.
526 *           TICKNT IS INCREMENTED EVERY INTERRUPT.
527 *           ERRNZ *-201A
528 *           ERRNZ *-201A
529
530
531
532 *           LHALD TICCNT
533 *           INX H
534 *           SHLD TICCNT
535 *           LDA CTLFLG
536 *           OUT OF.CTL
537 *           CLEAR CLOCK INTERRUPT FLIP-FLOP
538 *           EXIT CLOCK INTERRUPT.
539 *
540
541 *           LXI B,CTLFLG
542 *           LDAX B
543 *           ANI CB.MTL
544 *           JNZ INXIT
545 *           INX B
546 *           ERKNZ CTLFLG-.MFLAG-1
547 *           LDAX B
548 *           ERKNZ UD.HLT-2008
549 *           RAL
550 *           JC CLK4
551 *           NOT IN MONITOR MODE. CHECK FOR HALT
552 *
553 *           MVI A,10
554 *           CALL LRA
555 *           MOV E,H
556 *           INX H
557 *           MOV D,M
558 *           DCX D
559 *           LDAX D
560 *           CPI MI.HLT
561 *           JNZ CUI1
562 *           MVI A,A.BEL
563 *           CALL MCC
564 *           MVI A,'H'
565 *           CALL A,'H'
566 *           CALL MCC
567 *           JMP ERROR
568 *           JE ERROR
569 *           IF HALT, BE IN MONITOR MODE
570 ***
571 *
572 *           NONE OF THE ABOVE, SO ALLOW USER PROCESSING OF CLOCK INTERRUPT
573
574 *           EQU CLK4
575 *           JMP CUI1
576 *           LON
577 *           ALLOW USER PROCESSING OF CLOCK
578 *
579 *
580 *
581 *
582 *
583 *
584 *
585 *
586 *
587 *
588 *
589 *
590 *
591 *
592 *
593 *
594 *
595 *
596 *
597 *
598 *
599 *
600 *
    
```

```

604 *** ERROR - COMMAND ERROR.
605 *
606 * ERROR IS CALLED AS A 'BAIL-OUT' ROUTINE.
607 *
608 * IT RESETS THE OPERATIONAL MODE, AND RESTORES THE STACK POINTER.
609 *
610 * NONE
611 * TO MTR LOOP
612 * CTLFLG SET
613 * MFLAG CLEARED
614 * USES ALL
615 *
000.000 ERRNZ *-322A
616
617
000.322 EQU *
000.323 LXI H,MFLAG
000.324 MOV A,M (A) = MFLAG
000.325 ANI 3770-UO,DDU-UO,NFR RE-ENABLE DISPLAYS
000.326 MOV M,A REPLACE
000.327 INX H
000.328 MVI M,CB,SSI+CB,MTL+CB,CLI+CB,SPK RESTORE *CTLFLG*
000.329 ERRNZ CTLFLG-MFLAG-1
000.330
000.331 LHL D REGPTR
000.332 EI
000.333 LHL D
000.334 SPHL RESTORE STACK POINTER TO EMPTY STATE.
000.335 CALL ALARM ALARM FOR 200 MS
000.336
000.337
000.338
000.339
000.340
000.341

```

```

631 ** MTR - MONITOR LOOP.
632 *
633
634 ERRNZ *-344A
635
636 MTR *
637 EI
638
639 MTR1 *
640 LXI H,MTR1
641 PUSH H
642 LXI H,MSG,PR
643 CALL TYPMSG
644
645 MTR.2 CALL RCC
646 ANI 01011111B READ A CONSOLE CHARACTER
647 LXI H,MTR.A MAKE SURE ITS UPPER CASE TO MATCH TABLE
648 MVI B,MTRAL LOOK UP CHARACTER IN *MTR.A*
649 MTR.3 CMP M, (B) = LENGTH OF TABLE
650 JZ MTR.4 SEE IF CHARACTER FROM CONSOLE = TABLE ENTRY
651 IF EQUAL
652 INX H POINT TO NEXT TABLE ENTRY
653 INX H
654 INX H
655 DCR B SEE IF PAST END OF TABLE
656 JNZ MTR.3 IF NOT PAST

```

MTR88 - H8B MONITOR \$09.00.00.  
 MTR - MAIN EXECUTIVE LOOP.

HEATH HBASH V1.4 01/20/78 PAGE 14  
 08:59:14 17-MAY-79

```

657 MVI A,A,BEL ELSE, DING ERROR
658 CALL WCC
659 JMP MTR.2 TRY AGAIN
661 MTR.4 WRITE CHARACTER BACK TO CONSOLE
662 INX H GET ROUTINE ADDRESS LSB
663 MOV A,H GET MSB
664 INX H
665 MOV H,H (HL) = ROUTINE ADDRESS
666 MOV L,A GO TO ROUTINE
667 PCHL
668
669
670
671 EQU * JUMP TABLE
672 MTR.4 *
673 'G' GO TO USER ROUTINE
674 DW 6088
675
676
677 'L' CASSETTE LOAD
678 DW SRMEM
679
680 'D' SET UP CASSETTE DUMP
681 DW SWMEM
682
683 'S' SUBSTITUTE MEMORY MODE
684 DW SUBM
685
686 'F' PROGRAM COUNTER ALTER MODE
687 DW FCA
688
689 'B' ROOT HDOS
690 DW BOOT
691 MTRAL *MTRA/3 NUMBER OF TABLE ENTRIES /JMT 790507/
692 EQU L
693
705 ** SAE - STORE ABUSS AND EXIT.
706 * ENTRY (HL) = ABUSS VALUE
707 * EXIT TO (RET)
708 * USES NONE
709 *
710 ERRNZ *-1063A
711
712 SAE SHLD ABUSS
713
714 RET
    
```

```

717 ** SRMEM - H88/H89 ENTRY POINT FOR A CASSETTE LOAD
718 *
719
001.067 041 170 006 LXI H,MSG,LD COMPLETE MESSAGE
001.072 315 100 006 CALL TYPMSG
001.075 315 003 006 CALL WCR WAIT FOR A CARRIAGE RETURN
001.100 303 261 001 JMP RMEM LOAD TAPE
723
725 ** PCA - PROGRAM COUNTER ALTER
726 *
727 * PCA INPUTS AND/OR DISPLAYS THE CURRENT USER PROGRAM VALUE AND ALLOWS
728 * A NEW VALUE TO BE ENTERED OR RETAINS THE CURRENT VALUE IF
729 * A CR IS TYPED
730 *
731 * ENTRY NONE
732 * EXIT NONE
733 * USES A,D,E,H,L,F
734
735
001.103 041 214 006 LXI H,MSG,PC COMPLETE PC MESSAGE
001.106 315 100 006 CALL TYPMSG
001.111 076 012 MVI A,10 GET LOCATION OF USER PC
001.113 315 052 003 CALL LRA.
001.116 136 MOV E,M (D,E) = USER PC VALUE
001.117 043 INX H
001.120 126 MOV D,M
001.121 353 XCHG (H,L) = USER PC VALUE
744
001.122 315 150 005 CALL IROC INPUT NEXT CHARACTER
001.125 332 137 001 JC FCA1 IF FIRST CHARACTER WAS OCTAL, INPUT NEW PC
747
001.130 315 313 005 CALL TOA ELSE, OUTPUT CURRENT VALUE
001.133 315 150 005 CALL IROC SEE IF USER WANTS TO CHANGE IT NOW
001.136 320 RNC IF NO CHANGE, EXIT
751
752 * ENTER NEW USER PC VALUE
753
001.137 353 XCHG (H,L) = ADDRESS OF USER PC VALUE
001.140 026 015 MVI D,A,CR END BYTE WITH A RETURN
001.142 315 062 003 CALL TOA INPUT NEW ADDRESS
001.145 311 RET EXIT
757
759 ** GO88 - GO TO USER ROUTINE FROM H88 MONITOR
760 *
761 * GO88 WAITS FOR A CARRIAGE RETURN OR A NEW ADDRESS. TERMINATED WITH
762 * A CARRIAGE RETURN. IF NO ADDRESS IS ENTERED, GO88 TRANSFERS
763 * CONTROL TO THE ADDRESS SPECIFIED BY THE USER PC VALUE
764
001.146 041 165 006 LXI H,MSG,GO COMPLETE GO MESSAGE
765

```



MTR88 - H88 MONITOR #09.00.00. HEATH HS88M V1.4 01/20/78 PAGE 16  
 MONITOR TASK SUBROUTINES. 08:59:15 17-MAY-79

```

001.151 315 100 006 767 CALL TYPMSG
001.154 315 150 005 768 CALL IROC
001.157 322 177 001 769 JNC G088.1
001.162 365 076 012 771 PUSH A:10
001.163 076 012 772 CALL LRA
001.165 315 052 003 773 INX H
001.170 043 774 FSW D:A:CR
001.171 361 775 MOV D:A:CR
001.172 026 015 776 CALL IOA
001.174 315 062 003 777 WCC G088.1
001.177 315 302 003 778 CALL A:A:LF
001.202 076 012 779 CALL WCC
001.204 315 302 003 780 JMP GO
001.207 303 232 001 781
    
```

001.212 110 103 101 783 DB 1108,1038,1018,0468,1038,1018,1078,0568 DESIGN CODE

```

000.000 785 ** GO - RETURN TO USER NONE
786 * ENTRY NONE
787 * ENTRY NONE
788 ERRNZ *-1222A
789
001.222 303 063 000 791 JMF GO
    
```

```

793 ** SSTEP - SINGLE STEP INSTRUCTION
794 * ENTRY NONE
795 * ENTRY NONE
796 ERRNZ *-1225A
797
001.225 363 799 SSTEP EQU *
001.226 072 011 040 800 DI
001.231 356 020 801 LVA CTLFLAG
001.233 323 360 802 XRI CR,SSI
001.235 062 011 040 803 SSI OP,CIL
001.240 341 804 STA CTLFLAG
001.241 303 172 000 805 FOP H
806 JMP INTXIT
    
```

```

SINGLE STEP
DISABLE INTERRUPTS UNTIL THE RIGHT TIME
CLEAR SINGLE STEP INHIBIT
PRIME SINGLE STEP INTERRUPT
SET NEW FLAG VALUES
CLEAN STACK
RETURN TO USER ROUTINE FOR STEP
    
```

```

808 ** STPRN - SINGLE STEP RETURN.
809
810 ERRNZ *-1244A
811
812 STPRN EQU *-1244A
813 ORI CR,SSI *
814 OUT OP,CTL DISABLE SINGLE STEP INTERRUPTION
815 SET CILFLG TURN OFF SINGLE STEP ENABLE
816 STAX D
817 ANI CR,MPL SEE IF IN MONITOR MODE
818 JNZ MTR
819 JMP UIVEC+3 TRANSFER TO USER'S ROUTINE

000:000
001:244
001:244 366 020
001:246 323 360
040:011
001:250 022
001:251 346 040
001:253 302 344 000
001:256 303 042 040

821 ** RMEM - LOAD MEMORY FROM TAPE.
822 *
823
824 ERRNZ *-1261A
825
826 RMEM LXI H,TPABY
827 SHLD TPERRX SETUP ERROR EXIT ADDRESS
828 JMP LOAD

830 *** LOAD - LOAD MEMORY FROM TAPE.
831 *
832 * READ THE NEXT RECORD FROM THE CASSETTE TAPE.
833 *
834 * USE THE LOAD ADDRESS IN THE TAPE RECORD.
835 *
836 * ENTRY (HL) = ERROR EXIT ADDRESS
837 * USER P-REG (IN STACK) SET TO ENTRY ADDRESS
838 * TO CALLER IF ALL OK
839 * TO ERROR EXIT IF TAPE ERRORS DETECTED.
840
841 ERRNZ *-1267A
842
843
844 LOAD EQU *
845 LXI B,1000A-RT.MI*256-256 (BC) = - REQUIRED TYPE AND #
846 CALL SRS SCAN FOR RECORD START
847 MOV L,A (HL) = COUNT
848 XCHG C (DE) = COUNT, (HL) = TYPE AND #
849 DCR R (C) = - NEXT #
850 MOV R,A,H
851 MOV A,R
852 PUSH R
853 FSW
854 ANI .177R
855 ORA L
856 MVI A,2
857 JNE TFERR
    SAVE TYPE AND #
    SAVE TYPE CODE
    CLEAR END FLAG BIT
    SEQUENCE ERROR
    IF NOT RIGHT TYPE OR SEQUENCE
    
```

MTR88 - H8B MONITOR #09.00.00..... HEALTH HBASM V1.4 01/20/78 ..... PAGE 18  
 MONITOR TASK SUBROUTINES..... 08:59:16 17-MAY-79

```

001.314 315 325 002 858 CALL RNP READ ADDR
001.317 104 859 MOV BPH
001.320 117 860 MOV C,A (BC) = P-REG ADDRESS
001.321 076 012 861 MVI A,10
001.323 325 862 PUSH D SAVE (DE)
001.324 315 052 003 863 CALL LRA. LOCATE REG ADDRESS
001.327 321 864 POP D RESTORE (DE)
001.330 161 865 MOV M,C SET P-REG IN MEM
001.331 043 866 INX H
001.332 160 867 MOV M,B
001.333 315 325 002 868 CALL RNP READ ADDRESS
001.336 157 869 MOV L,A (HL) = ADDRESS, (DE) = COUNT
001.337 042 000 040 870 SHLD START
001.342 315 331 002 871 L0A1 READ BYTE
001.345 167 872 MOV M,A
873
874 * SHOW H8B THAT SOMETHING IS HAPPENING
875
876
877 CALL TFRSF DISPLAY TO H8B USER THAT WE ARE LOADING
001.346 315 024 006 878
001.351 043 879 INX H
001.352 033 880 DCX D
001.353 172 881 MOV A,D
001.354 263 882 ORA E
001.355 302 342 001 883 JNZ L0A1 IF MORE TO GO
001.360 315 172 002 884 CALL CTC CHECK TAPE CHECKSUM
885
886 * READ NEXT BLOCK
887
888
889 POP FSW (A) = FILE TYPE BYTE
001.364 301 890 POP B (BC) = -(LAST TYPE, LAST #)
001.365 007 891 RLC
001.366 332 133 002 892 JC JFT ALL DONE - TURN OFF TAPE
001.371 303 272 001 893 JMP L0A0 READ ANOTHER RECORD
    
```

896	***	DUMP	...	DUMP MEMORY TO MAG TAPE	
897	*				
898	*	DUMP	...	SPECIFIED MEMORY RANGE TO MAG TAPE	
899	*				
900	*	ENTRY	(START) = START ADDRESS		
901	*		(ABUSS) = END ADDRESS		
902	*		USER PC = ENTRY POINT ADDRESS		
903	*	EXIT	TO CALLER		
904					
000.000		ERRNZ	*-1374A		
905					
001.374		EGU	* H,TFABT		
001.374		LXI	TPERRX	SETUP ERROR EXIT	
001.377		SHLD			
909					
910		ERRNZ	*-2002A		
911					
912					
002.002		MVI	A,UCI,TE		
002.004		OUT	OP,TPC	SETUP TAPE CONTROL	
002.006		MVI	A,A,SYN		
002.010		MVI	H,32	(H) = # OF SYNC CHARACTERS	
002.012		CALL	WNB		
002.015		DCR	H		
002.016		JNZ	WNEI	WRITE SYN HEADER	
002.021		MVI	A,A,STX		
002.023		CALL	WNB	WRITE SIX	
002.026		MOV	L,H	(HL) = 00	
002.027		SHLD	CRCSUM	CLEAR CRC 16	
002.032		LXI	H,RT,MI+80H*256+1	FIRST AND LAST MI RECORD	
002.035		CALL	WNP	WRITE HEADER	
002.040		LHLD	START		
002.043		XCHG		(D,E) = START ADDRESS	
002.044		LHLD	ABUSS	(H,L) = STOP ADDR	
002.047		INX	H	COMPUTE WITH STOP+1	
002.050		MOV	A,L		
002.051		SUB	E		
002.052		MOV	L,A		
002.053		MOV	A,H		
002.054		SBB	D	(HL) = COUNT	
002.055		MOV	H,A	WRITE COUNT	
002.056		CALL	WNF		
002.061		PUSH	H		
002.062		MVI	A,10	SAVE (DE)	
002.064		PUSH	D	LOCATE F-REG ADDRESS	
002.065		CALL	L,RA		
002.070		MOV	A,M		
002.071		INX	H		
002.072		MOV	H,M		
002.073		MOV	L,A	(HL) = CONTENTS OF PC	
002.074		CALL	WNF	WRITE HEADER	
002.077		POP	H	(HL) = ADDRESS	
002.100		POP	D	(DE) = COUNT	
002.101		CALL	WNE		
949					
002.104		MOV	A,M	WRITE BYTE	
002.105		CALL	WNB		
950					
951					

MTR88 - H88 MONITOR #09.00.00. HEATH HBASH V1.4 01/20/78 PAGE 20  
 DUMP - DUMP MEMORY TO MAG/PAPER TAPE 08:59:17 17-MAY-79

```

952 * SHOW H88 USER THAT DUMP IS DUMPING
953 *
954 * CALL TFDSP DISPLAY DUMP
955 *
002.110 315 024 006
002.113 043 H
002.114 033 D
002.115 172 MOV A,D
002.116 263 ORA E
002.117 302 104 002 JNZ WME2 IF MORE TO GO
961 *
962 * WRITE CHECKSUM
963 *
964 * LHLD CRCSUM
965 * CALL WNP WRITE IT
966 * CALL WNP FLUSH CHECKSUM
967 * JMP TFT
968 *

970 ** TFT - TURN OFF TAPE.
971 *
972 * STOP THE TAPE TRANSPORT.
973 *
974 *
975 * ERRNZ *-2133A
976 *
002.133 257 TFT A
002.134 323 371 OUT OP,TPC TURN OFF TAPE

980 ** HORN - MAKE NOISE.
981 *
982 * ENTRY (A) = (MILLISECOND.COUNT)/2
983 * EXIT NONE
984 * USES A,F
985 *
986 * ERRNZ *-2136A
987 *
988 * ALARM EQU *
989 * ALARMB BRANCH TO A JUMP TO NOISE TO DING BELL
990 *
991 * ERRNZ *-2140A
992 *
993 * HORN PUSH PSW
994 * MVI A,CR,SFK TURN_ON_SPEAKER
995 *
996 * HRNO XTHL (HL),H) = COUNT
997 * PUSH D SAVE (DE)
998 * XCHG (D) = LOOP.COUNT
999 * LXI H,CTLFLG
1000 * XRA M
1001 * MOV E,M (E) = OLD CTLFLG VALUE
    
```

MTR88 - H8B MONITOR #09.00.00.  
DUMP - DUMP MEMORY TO MAG/PAPER TAPE

HEATH HBASHM V1.4 01/20/78  
08:59:17 17-MAY-79

PAGE 21

```

002.153 167 1002 MOV MIA JURN ON HORN
002.154 056 033 1003 MVI L.#TICCNT
002.156 172 1005 MOV A:D (A) = CYCLE COUNT
002.157 206 1006 ADD M
002.160 276 1007 HRN2 M WAIT REQUIRED TICCOUNTS
002.161 302 160 002 1008 JNE HRN2
002.164 303 045 006 1010 JMP HRNX JUMP TO AN EXTENSION OF HORN SO ROOM
1011 * CAN BE MADE FOR A JUMP TO NIOSE
1012
1013
002.167 303 053 006 1014 ALARMB JMP NOISE SEND A BELL TO THE CONSOLE
1015

```

MTR88 - H88 MONITOR \$09.00.00.  
 TAPE PROCESSING SUBROUTINES

HEATH HBASM V1.4 01/20/78  
 08:59:17 17-MAY-79

PAGE 22

```

1020 **      CTC - VERIFY CHECKSUM.
1021 *      ENTRY TAPE JUST BEFORE CRC
1022 *      EXIT TO CALLER IF OK
1023 *
1024 *      TO *TPERR* IF BAD
1025 *      USES A,F,H,L
1026
1027
1028 ERRNZ *-2172A
1029
000.000
002.172 315 325 002 CTC          RNF          READ NEXT PAIR
002.175 052 027 040 LHLDB      CRCSUM
002.200 174          MOV          A,H
002.201 265          ORA          L
002.202 310          RZ
002.203 076 001     MVI          A,I          RETURN OF OK
                                           CHECKSUM ERROR
                                           (R) = CODE
1036 *      JMP          TPERR
1037
1038 **      TPERR - PROCESS TAPE ERROR.
1039 *
1040 *      DISPLAY ERR NUMBER IN LOW BYTE OF ADDRESS
1041 *
1042 *      IF ERROR NUMBER EVEN, DON'T ALLOW *
1043 *      IF ERROR NUMBER ODD, ALLOW *
1044 *
1045 *      ENTRY (R) = PATTERN
1046
1047 ERRNZ *-2205A
1048
002.205 107          MOV          B,A          (R) = CODE
002.206 315 063 006 CALL          TPERMSG      DISPLAY ERROR NUMBER ON CONSOLE
002.211 315 133 002 CALL          TFI          TURN OFF TAPE
1052
1053 *      IS #1 RETURN (IF PARITY ERROR)
1054
1055 DB          MI,ANI          FALL THROUGH WITH CARRY CLEAR
002.214 346          MOV          A,R
002.215 170
1057
1058 RRC
1059 RC
1060
1061 *      BEEP AND FLASH ERROR NUMBER
1062
002.220 334 136 002 CC          ALARM
002.223 315 252 002 CALL          TPXIT      ALARM IF PROPER TIME
                                           SEE IF *
002.224 333 360     IN          IF,PAD      CHECK FOR #
002.230 376 057     CFI          00101111B
002.232 312 215 002 JE          TERS          IF #
002.235 072 034 040 LDA          TICNT+1
002.240 037          RAK
002.241 303 220 002 JMP          TER1
    
```

```

1072 **      TFABT - ABORT TAPE LOAD OR DUMP.
1073 *
1074 *      ENTERED WHEN LOADING OR DUMPING AND THE * KEY
1075 *      IS STRUCK.
1076
1077
1078 ERRNZ *-2244A
1079
1080 TFABT XRA A
1081 OUT OP:TPC
1082 JMP ERROR OFF TAPE
1083
1084 **      TPXIT - CHECK FOR USER FORCED EXIT.
1085 *
1086 *      TPXIT CHECKS FOR AN * KEYPAD ENTRY. IF SO, TAKE
1087 *      THE TAPE DRIVER ABNORMAL EXIT.
1088 *
1089 *      ENTRY NONE
1090 *      EXIT TO *RET* IF NOT *
1091 *      (A) = PORT STATUS
1092 *      TO (TFERRX) IF * DOWN
1093 *      USES A:F
1094
1095 ERRNZ *-2252A
1096
1097
1098 TPXIT IN IP:PAD
1099 CPI 01101111B * READ TAPE STATUS
1100 IN IP:TPC * NOT *; RETURN WITH STATUS
1101 RNE
1102 LHLD TFERRX
1103
1104 ERRNZ *-2264A
1105
1106 FCHL ENTER (TFERRX)
1107
1108 **      SRS - SCAN RECORD START
1109 *
1110 *      SRS READS BYTES UNTIL IT RECOGNIZES THE START OF A RECORD.
1111 *
1112 *      THIS REQUIRES
1113 *      AT LEAST 10 SYNC CHARACTERS
1114 *      1 SIX CHARACTER
1115 *
1116 *      THE CRC-16 IS THEN INITIALIZED.
1117 *
1118 *      ENTRY NONE
1119 *      EXIT TAPE POSITIONED (AND MOVING), CRCSUM =0
1120 *      (DE) = HEADER BYTES
1121 *      (HA) = RECORD COUNT

```



MTR88 - H88 MONITOR \*09,00,00. TAPE PROCESSING SUBROUTINES. HEATH HBASM V1.4 01/20/78 PAGE 24  
 08:59:18 17-MAY-79 SRS

```

1122 *      USES      A,F,D,E,H,L
1123      ERRNZ      *-2265A
1124
000,000
1125
002,265      SRS      EQU      *
002,265      SRS1     MVI      D,0
002,267      MOV      H,D
1128
002,270      MOV      L,D
002,271      CALL     RNB
002,271      SRS2     (HL) = 0
002,274      INR      D
002,275      CFI      A,SYN
002,277      JE       SRS2
002,302      CFI      A,STX
002,304      JNE      SRS1
002,307      MVI      A,10
002,311      CMP      D
002,312      JNC      SRS1
002,315      SHLD   CRCSUM
002,320      CALL     RNF
002,323      MOV      D,H
002,324      JMP      E,A
1143      RNF
1144 *
1146 **      RNF - READ NEXT PAIR.
1147 *      RNF READS THE NEXT TWO BYTES FROM THE INPUT DEVICE.
1148 *
1149 *
1150 *      ENTRY     NONE
1151 *      EXIT      (H,A) = BYTE PAIR
1152 *      USES      A,F,H
1153
000,000      ERRNZ      *-2325A
1154
002,325      CALL     RNB
002,330      MOV      H,A
1157      JMF      RNB
1158 *
1160 **      RNB - READ NEXT BYTE
1161 *
1162 *      RNB READS THE NEXT SINGLE BYTE FROM THE INPUT DEVICE.
1163 *      THE CHECKSUM IS TAKEN FOR THE CHARACTER.
1164 *
1165 *      ENTRY     NONE
1166 *      EXIT      (A) = CHARACTER
1167 *      USES      A,F
1168
000,000      ERRNZ      *-2331A
1170
002,331      MVI      A,UCI,RO+UCI,ERUCI,RE...TURN ON READER FOR NEXT BYTE
1171      RNB
    
```

002.333 323.371 1172 RNB1 \* OP.TFC
002.335 315.252 002 1173 RNB1 \* CALL TFXIT
002.340 346.002 1174 RNB1 \* USR.RXR CHECK FOR \*, READ STATUS
002.342 312.335 002 1175 RNB1 \* JZ IF NOT READY
002.345 333.370 1176 RNB1 \* IN IP.TPD INPUT DATA
1177 \* JMP CRC CHECKSUM

1179 \*\* CRC - COMPUTE CRC-16
1180 \*
1181 \* CRC COMPUTES A CRC-16 CHECKSUM FROM THE POLYNOMIAL
1182 \*
1183 \* (X + 1) \* (X^15 + X + 1)
1184 \*
1185 \* SINCE THE CHECKSUM GENERATED IS A DIVISION REMAINDER,
1186 \* A CHECKSUMED DATA SEQUENCE CAN BE VERIFIED BY RUNNING
1187 \* THE DATA THROUGH CRC, AND THEN RUNNING THE PREVIOUSLY OBTAINED
1188 \* CHECKSUM THROUGH CRC. THE RESULTANT CHECKSUM SHOULD BE 0.
1189 \*
1190 \* ENTRY (CRCSUM) = CURRENT CHECKSUM
1191 \* (A) = BYTE
1192 \* (CRCSUM) UPDATED
1193 \* (A) UNCHANGED.
1194 \*
1195 \*
1196 \* ERRNZ \*-2347A

000.000
002.347 305 1197 CRC
002.350 006 010 1198 CRC
002.352 345 1200 PUSH B,B SAVE (BC)
002.353 052 027 040 1201 LHL D CRCSUM (B) = BIT COUNT
002.356 007 1202 CRC1 RLC
002.357 117 1203 MOV C,A (C) = BIT
002.361 207 1204 MOV A,L
002.362 157 1205 ADD A (A) = BYTE
002.363 174 1206 MOV L,A (CRCSUM) UPDATED
002.364 027 1207 MOV A,H (A) UNCHANGED.
002.365 147 1208 RAL
002.366 027 1209 MOV H,A
002.367 251 1210 RAL
002.370 017 1211 XRA C
002.371 322 004 003 1212 RRC
002.374 174 1213 JNC CRC2 IF .NOT. TO .XOR
002.375 356 200 1214 MOV A,H
002.377 147 1215 XRI 2000
003.000 175 1216 MOV H,A
003.001 356 005 1217 MOV A,L
003.003 157 1218 XRI 50
003.004 171 1219 MOV L,A
003.005 005 1220 MOV A,C
003.006 302 356 002 1221 RCR B
003.011 042 027 040 1222 JNZ CRC1 IF MORE TO GO
003.014 341 1223 SHLD CRCSUM
1224 POP H RESTORE (HL)

MTR88 - H88 MONITOR \*09.00.00.  
 TAPE PROCESSING SUBROUTINES

HEATH HBASH, V1.4 01/20/78 PAGE 26  
 08:59:19 17-MAY-79

003.015 301 FOR R RESTORE (BC)  
 003.016 311 RET EXIT

1228 \*\* WNF - WRITE NEXT PAIR.  
 1229 \*  
 1230 \* WNF WRITES THE NEXT TWO BYTES TO THE CASSETTE DRIVE.  
 1231 \*  
 1232 \* • ENTRY (H,L) = BYTES.  
 1233 \* EXIT WRITTEN.  
 1234 \* USES A,F.  
 1235  
 000.000 ERRNZ \*-3017A  
 1236  
 1237  
 003.017 174 MOV A,H  
 003.020 315 024 003 CALL WNB  
 003.023 175 MOV A,L  
 1241 \* JMP WNB WRITE NEXT BYTE

1243 \*\* WNB - WRITE BYTE.  
 1244 \*  
 1245 \* WNB WRITES THE NEXT BYTE TO THE CASSETTE TAPE.  
 1246 \*  
 1247 \* ENTRY (A) = BYTE.  
 1248 \* EXIT NONE.  
 1249 \* USES F.  
 1250  
 000.000 ERRNZ \*-3024A  
 1251  
 1252

003.024 365 WNB PUSH PSW CHECK FOR \*, READ STATUS  
 003.025 315 252 002 CALL TFXIT  
 003.030 346 001 ANI USR, TXR IF MORE TO GO  
 003.032 312 025 003 JZ WNB1 WNB1 A,UCI, ERHUCI, TE ENABLE TRANSMITTER.  
 003.035 076 021 MVI A,UCI, ERHUCI, TE TURN ON TAPE  
 003.037 323 371 OUT OP, IPC  
 003.041 361 FOP PSW OUTPUT DATA  
 003.042 323 370 OUT OP, IPD COMPUTE CRC  
 003.044 303 347 002 JMP WNB

```

1265 ** LRA - LOCATE REGISTER ADDRESS.
1266 *
1267 * ENTRY NONE.
1268 * EXIT (A) = REGISTER INDEX
1269 * (H,L) = STORAGE ADDRESS
1270 * (D,E) = (0,A)
1271 * USES A,D,E,H,L,F
1272 *
1273 *
1274 * ERRNZ *-3047A
1275 *
1276 * LRA REGI
1277 * LRA E,A
1278 * MOV D,0
1279 * LHL REGPTR
1280 * DAI D
1281 * RET
(D,E) = (REGPTR)+(REGI)

```

```

1283 ** IOA - INPUT OCTAL ADDRESS.
1284 *
1285 * ENTRY (H,L) = ADDRESS OF RECEPTION DOUBLE BYTE.
1286 * (D) = TERMINATING CHARACTER
1287 * EXIT NONE
1288 * USES A,D,E,H,L,F
1289 *
1290 *
1291 * ERRNZ *-3062A
1292 *
1293 * IOA JMP IOA1
1294 * NOP
RETAIN_H8_ORG

```

```

1296 ** IOB - INPUT OCTAL BYTE.
1297 *
1298 * READ ONE OCTAL BYTE FROM THE KEYSET.
1299 *
1300 * ENTRY (H,L) = ADDRESS OF BYTE TO HOLD VALUE
1301 * (C) = SET IF FIRST DIGIT IN (A)
1302 * EXIT NONE
1303 * USES A,D,E,H,L,F
1304 *
1305 *
1306 * ERRNZ *-3066A
1307 *
1308 * IOB MVI M,0
1309 * IOB1 CMC
1310 * IOB11 * SEE IF CHARACTER IS A VALID OCTAL VALUE
1311 * JC IOB2
1312 *
1313 *
1314 *
LESS_THAN_ZERO?
IF (A) < 0? SEE IF A TERMINATING CHARACTER

```

MTR88 - H88 MONITOR #09.00.00. HEATH.H8ASM.V1.4.01/20/78. PAGE 28.  
SUBROUTINES 08:59:20 17-MAY-79

```

003.100 376 070 1315 CPI '8'
003.102 322 070 003 1316 JNC IOB1
1317 *
1318 * HAVE AN OCTAL DIGIT
1319 *
003.105 315 302 003 1320 CALL WCC
003.110 346 007 1321 ANI 00000111B ECHO CHARACTER
003.112 137 1322 MOV E'A MASK FOR BINARY VALUE
003.113 176 1323 MOV A,M '(E)' = VALUE
003.114 007 1324 RLC GET OLD VALUE
003.115 007 1325 RLC SHIFT 3
003.116 007 1326 RLC
003.117 303 126 003 1327 JMP IOB1.5 JUMP AROUND AN H88/H89 TO H8 FAKE ROUTINE
003.118 1328 **
1329 ** FAKE OUT ROUTINE FOR CALLERS OF *DOD* FROM THE H8 FRONT PANEL
1330
1331
1332 ERRNZ *-3122A
1333
003.122 043 1334 DDD INX H
003.123 043 1335 INX H
003.124 043 1336 INX H
003.125 311 1337 RET
1338
1339
1340 * CONTINUE
1341
003.126 346 370 1342 IOB1.5 ANI 11111000B TOSS OLD LSB DIGIT
003.130 263 1343 ORA E REPLACE WITH NEW VALUE
003.131 167 1344 MOV M,A M,A
003.132 303 070 003 1345 JMP IOB1 INPUT ANOTHER CHARACTER
1346
1347 * CHECK FOR A CARRIAGE RETURN TO TERMINATE BYTE
1348 *
003.135 376 015 1349 IOB2 CPI A.CR CARRIAGE RETURN?
003.137 310 1350 RZ RETURN IF CARRIAGE RETURN /JMT 790507/
003.140 257 1351 XRA A CLEAR CARRY /JMT 790507/
003.141 030 325 1352 JR IOB1 GET A NEW CHARACTER /JMT 790507/
1353
1433 L

```

```

1437 ** RCK - READ CONSOLE KEYPAD
1438 *
1439 * RCK IS CALLED TO READ A KEYSTROKE FROM THE CONSOLE FRONT PANEL KEYPAD.
1440 * SINCE THE H88/89 DOES NOT HAVE A FRONT PANEL, THIS ROUTINE IS PROVIDED
1441 * ONLY TO MAINTAIN COMPATIBILITY WITH FAM-8.
1442 * RCK WILL IMMEDIATELY RETURN WITH A VALUE OF 0 (ZERO) IN THE ACCUMULATOR.
1443 *
1444 * ENTRY NONE
1445 * EXIT (A) = 0
1446 * USES A,F
1447 *
1448 * RCK MUST HAVE SAME ENTRY AS RCK IN FAM-8
1449 * ERRNZ *-3260A
1450 *
000,000 EQU *
003,260 XRA A
003,260 RET
003,261
003,261
1455

```

MTR88 - HBB MONITOR #09.00.00.  
CONSOLE CHARACTER ROUTINES.

```

1459 **      RCC - READ CONSOLE CHARACTER.
1460 *
1461 *      RCC IS CALLED TO READ A KEYSTROKE FROM THE CONSOLE.
1462 *      IF A RUBOUT/DELETE IS RECEIVED, EXIT IS TO *ERROR*.
1463 *
1464 *      ENTRY NONE
1465 *      TO ERROR - IF A DELETE OR RUBOUT IS ENCOUNTERED
1466 *      TO CALLER - WHEN A KEY IS HIT
1467 *      (A) = ASCII KEY VALUE
1468 *      USES A7F
1469
1470
1471
1472 RCC EQU *
1473
1474 IN SC:ACE+UR.LSR INPUT ACE LINE STATUS REGISTER
1475 ANI UC,DR SEE IF THERE IS A DATA READY
1476 JZ,RCC1
1477
1478 IN SC:ACE+UR.RBR ELSE, INPUT CHARACTER
1479 ANI 01111111B TOSS ANY PARITY
1480 CPI A,BEL
1481 JZ ERROR IF RUBOUT, EXIT TO ERROR
1482
1483 RET ELSE, EXIT TO CALLER
1484
1485 **      WCC - WRITE CONSOLE CHARACTER
1486 *
1487 *      WRITE A CHARACTER TO THE CONSOLE UART PORT
1488 *
1489 *      ENTRY (A) = ASCII CHARACTER TO OUTPUT
1490 *      EXIT NONE
1491 *      USES NONE
1492
1493
1494 WCC PUSH FSW SAVE CHARACTER
1495 WCC1 IN SC:ACE+UR.LSR INPUT ACE STATUS
1496 ANI UC,THE SEE IF TRANSMITTER HOLDING REGISTER IS EMPTY
1497 JR Z,WCC1
1498
1499 POP FSW GET CHARACTER
1500 OUT SC:ACE+UR.THR OUTPUT TO CONSOLE
1501 RET
1502
1503 LDN L
1504

```

```

1550 ** IO ROUTINES TO BE COPIED INTO AND USED IN RAM.
1551 *
1552 * MUST CONTINUE TO 377A FOR PROPER COPY.
1553 * THE TABLE MUST ALSO BE BACKWARDS TO THE FINAL RAM
1554
000.000 ERRNZ 4000A-7-*
1556
003.371 FRSR0M EQU *
1557 DB 1 REFIND
1558 DB 0 CTUFLG
1559 DB 0
1560 DB 0 MELAG
1561 DB 0 DSPMOD
1562 DB 0 DSPROT
1563 DB 10 REGI
1564 DB MI.RET
1565
000.000 ERRNZ *-4000A
1566
1567

```



HEATH.HBASM V1.4..01/20/78.....PAGE.....32.  
08:59:22 17-MAY-79

MTR88 - H88 MONITOR #02.00.00.  
H88/H89 ADDITIONAL ROUTINES

```

1570 ***      INITOX      ..... EXTENSION OF INIT0.ID SUPPORT.H88.....
1571
1572 INITOX      MVI      A,00000010R
1573      OUT      H88.CTL
1574
1575 *      SET UP ACE FOR CONSOLE COMMUNICATIONS
1576 *
1577      MVI      A,UC.DLA      SET DIVISOR LATCH ACCESS BIT
1578      OUT
1579      SC.ACE+UR.LCR      (H,L) = BEGINNING OF BAUD RATE TABLE
1580      H,BRTAB      INPUT SWITCHES FOR DESIRED BAUD RATE
1581      H88.SW      MASK FOR BAUD RATE SWITCHES ONLY
1582      ANI      H88S.BR      SHIFT FOR A *2 FOR TABLE
1583      RRC
1584      RRC
1585      RRC
1586      RRC
1587      ADD      L      ADD DISPLACEMENT FROM BEGINNING OF TABLE
1588      MOV      L,A
1589      MOV      A,M      GET MSB OF DIVISOR
1590      OUT      SC.ACE+UR.DLM      GET LSB
1591      INX      H
1592      MOV      A,M
1593      SC.ACE+UR.DLL
1594      A,UC.SRW      SET 8 BITS, 1 STOP BIT, NO PARITY
1595      OUT      SC.ACE+UR.LCR
1596      MVI      A,0
1597      MOV      SC.ACE+UR.IER
1598
1599 *      WAIT A WHILE TO ALLOW THE CONSOLE RESET TO FINISH SO IT CAN
1600 *      ACCEPT THE FIRST PROMPT.
1601 *
1602      LXI      R,65000A      APPROX 100 MS
1603      INITOX1 DCR      C
1604      JNZ      INITOX1
1605
1606      PCR      R
1607      JNZ      INITOX1
1608
1609 *      INPUT SWITCH TO SEE IF TO BEGIN OPERATION OR MEMORY TEST
1610 *
1611      IN      H88.SW      GET SWITCHES
1612      ANI      H88S.M      MASK FOR MEMORY TEST ONLY
1613      JZ      DYMEN      IF TO PERFORM MEMORY TESTS
1614
1615 *      REPLACE WHAT WAS ORIGINALLY AT THE JUMP WHICH GOT US HERE
1616 *
1617      LXI      D,FRSR0M      (DE) = ROM COPY OF FRS CODE
1618      JMP      INIT0.D      RETURN TO ORIGINAL CODE
    
```

1620 \*\* BRTAB - BAUD RATE DIVISOR TABLE

```

1621 *
1622 BRTAB EQU *
1623
1624 BR96 DB 0.12 8400 BAUD
1625 BR97 DB 0.6 19200 BAUD
1626 BR98 DB 0.3 38400 BAUD
1627 BR99 DB 0.3 58000 BAUD
1628
1629 SET #256
1630 ERRENZ EXTAB/256 TABLE MUST BE IN ONE PAGE

```

1632 \*\*\* SAVALLY - SAVALL EXTENSION TO MAKE ROOM FOR A JUMP TO THE INT HANDLER

```

1633
1634 SAVALLY EQU *
1635 PUSH H REPLACE OUR CODE
1636 PUSH D SET ON STACK AS REGISTER
1637 LXI D,CITFLG SET RETURN ADDRESS
1638 LDAX D
1639 JRC SAVALLR RETURN TO OLD CODE

```

```

1642 **** NMI - NON MASKABLE INTERRUPT
1643 *
1644 * NMI IS USED AS THE TRAP FOR ALL ILLEGAL PORT REQUESTS
1645 *
1646 * PORT ADDRESSES TRAPPED ARE:
1647 *
1648 *     IN 3600 FRONT PANEL KEYBOARD INPUT
1649 *     OUT 3600 FRONT PANEL CONTROL
1650 *     OUT 3610 FRONT PANEL DISPLAY CONTROL
1651 *     IN/OUT 3720 CONSOLE DATA FOR AN 8251A
1652 *     OUT 3730 CONSOLE CONTROL FOR AN 8251A
1653 *
1654 *
1655 * THESE PORT REQUESTS ARE RESPONDED TO AS FOLLOWS:
1656 *
1657 *     IN 3600 RETURNS WITH (A) = 3770 TO SHOW THAT
1658 *     NO FRONT PANEL SWITCHES ARE PRESSED
1659 *
1660 *     OUT 3600 MOVES BIT 6 (CB,CLI) TO BIT 1, AND
1661 *     BIT 4 (CB,SSI) INVERTED; TO BIT 0, AND
1662 *     OUTPUTS THESE BITS TO PORT 3620 TO
1663 *     CONTROL THE CLOCK AND SINGLE STEP INTERLOCKS
1664 *
1665 *     OUTPUTS TO 3610, 3720, AND 3730 JUST RETURN
1666 *
1667 *     INPUTS FROM 3610, 3720, AND 3730 RETURN WITH (A) = 0
1668 *     TO INDICATE AN EMPTY BUSS
1669 *
1670 *
1671 * ENTRY NONE
1672 *
1673 * EXIT NONE
1674 *
1675 * USES (A) ONLY IF 'FAKING' AN INPUT
1676 *
1677 *
1678 * NMI
1679 *
1680 * XTHL SHLD NMIRET GET RETURN ADDRESS FROM STACK
1681 * XTHL XTHL SAVE FOR LATER USE
1682 * PUSH H SAVE REGISTERS
1683 * PUSH B
1684 * PUSH PSW
1685 * MOV B,A SAVE (A) PRIOR TO I/O
1686 * LHLD NMIRET GET RETURN ADDRESS
1687 * DCX H BACK UP TO PORT # WHICH GOT US HERE
1688 * MOV A,M GET PORT #
1689 *
1690 * CPI 3600 PORT 3600
1691 * JZ NMII IF PORT WAS 3600
1692 *
1693 * PORT REFERENCED WAS 3610, 3720, OR 3730
1694 *
1695 * CPI 3610 MAKE SURE PORT IS LEGAL
1696 * JZ NMII.5 IF LEGAL
1697 *
    
```

004.146	374 372	1698	CEI	3720	
004.150	312 160 004	1699	JZ	NMI0.5	
004.153	376 373	1700	CPI	3730	
004.155	302 251 004	1701	JNZ	NMI2.5	IF NONE OF THE ABOVE, EXIT
004.160	053	1702	DCX	H	POINT TO IN/OUT INSTRUCTION
004.161	176	1703	MOV	A:M	SEE IF INPUT OR OUTPUT
004.162	376 323	1704	CPI	MI:OUT	
004.164	312 251 004	1705	JZ	NMI2.5	IF OUTPUT, JUST EXIT
004.167	376 333	1706	CPI	MI:IN	
004.171	302 251 004	1707	JNZ	NMI2.5	IF NOT INPUT EITHER, ILLEGAL SO EXIT
004.174	361	1708	POP	PSW	RESTORE FLAGS
004.175	076 000	1709	MOV	A:0	ELSE, RETURN LIKE AN EMPTY BOSS
004.177	303 252 004	1710	JMP	NMI3	EXIT
004.202	053	1711	DCX	H	POINT TO IN/OUT INSTRUCTION
004.203	176	1712	MOV	A:M	GET I/O INSTRUCTION
004.204	376 333	1713	CPI	MI:IN	INPUT?
004.206	302 217 004	1714	JNZ	NMI1.5	IF NOT 'IN'
004.211	361	1715	POP	PSW	RESTORE FLAGS
004.212	076 377	1716	MVI	A:11111111B	SHOW 'NO KEYS PRESSED'
004.214	303 252 004	1717	JMP	NMI3	EXIT
004.217	376 323	1718	CPI	MI:OUT	MAKE SURE INSTRUCTION IS AN 'OUT'
004.221	302 251 004	1719	JNZ	NMI2.5	IF NOT
004.224	170	1720	MOV	A:B	GET OUTPUT DATA AGAIN
004.225	346 100	1721	ANI	CE:CLI	MOVE CLOCK INFO TO BIT 1
004.227	017	1722	RRC		
004.230	017	1723	RRC		
004.231	017	1724	RRC		
004.232	017	1725	RRC		
004.233	017	1726	RRC		
004.234	117	1727	MOV	C:A	SAVE IN C
004.235	170	1728	MOV	A:B	GET OUTPUT DATA AGAIN
004.236	346 020	1729	ANI	CE:SSI	GET SINGLE STEP INFO
004.240	017	1730	RRC		MOVE TO BIT 0
004.241	017	1731	RRC		
004.242	017	1732	RRC		
004.243	017	1733	RRC		
004.244	241	1734	DRA	C	ADD TO CLOCK DATA
004.245	356 001	1735	XRI	00000001B	INVERT THIS BIT PRIOR TO OUTPUT
004.247	323 362	1736	OUT	H88:CTL	SET IN HARDWARE
004.251	361	1737	POP	PSW	RESTORE (A:F)
004.252	301	1738	POP	R	
004.253	341	1739	POP	H	
004.254	355 105	1740	RETN		ISSO RETURN FROM NMI
		1741	DB	3550,1050	

MTR88 - H88 MONITOR #09.00.00.  
ADDED TASK TIME ROUTINES FOR H88/H89

```

1754 **      ROOT HDOS ENTRY POINT FOR H88
1755 *
1756 *      ENTRY  NONE
1757 *      EXIT   TO HDOS BOOT ROM
1758 *      USES  ALL
1759 *
004.254 041 234 006      LXI  H,MSG,BT      COMPLETE ROOT MESSAGE
004.261 315 100 006      CALL  TYPMSG
004.264 315 003 006      CALL  WCR          WAIT FOR A CARRIAGE RETURN
004.267 076 012 1763     MVI  A,10
004.271 315 052 003 1764     LRA.          GET LOCATION OF USER PC
004.274 021 000 030 1765     LXI  D,R0M00  SET ITS VALUE TO THE BOOT ROM
004.277 163 1766     MOV  M,E
004.300 043 1767     INX  H
004.301 162 1768     MOV  M,D
004.302 041 122 006 1769     TELL USER TO TYPE SPACES TO DETERMINE BAUD RATE.
004.305 315 100 006 1770 *
004.310 303 063 000 1771 *
004.311 000 000 000 1772     LXI  H,MSG,SP
004.312 000 000 000 1773     CALL TYPMSG
004.313 000 000 000 1774     JMP  GO.
004.314 000 000 000 1775     DO IT

```

```

1777 **      SWMEM - SET UP FOR WMEM TO DUMP A CASSETTE FROM THE MONITOR LEVEL.
1778 *
1779 *      SWMEM ALLOWS THE USER TO EITHER ENTER A NEW STARTING AND
1780 *      ENDING ADDRESS FOR THIS DUMP OR USE THE ADDRESSES OF THE
1781 *      PREVIOUS LOAD OR DUMP. THE PREVIOUS ADDRESSES ARE USED IF
1782 *      THE FIRST CHARACTER IS AN ASCII CARRIAGE RETURN. IF THE
1783 *      FIRST CHARACTER IS AN OCTAL CHARACTER, BOTH BEGINNING AND
1784 *      ENDING ADDRESSES MUST BE ENTERED SEPARATED BY A DASH AND
1785 *      FOLLOWED BY A CARRIAGE RETURN.
1786 *
1787 *      ENTRY  USER PC VALUE ON STACK = PROGRAM START ADDRESS FOR THIS TAPE
1788 *      EXIT   TO WMEM
1789 *      USES  ALL
1790 *
1791 *
1792 *
004.313 041 174 006 1793     LXI  H,MSG,DMP      COMPLETE DUMP MESSAGE.
004.316 315 100 006 1794     CALL TYPMSG
004.321 315 150 005 1795     CALL  IROC          INPUT FIRST CHARACTER
004.324 322 351 004 1796     JNC  SWMEM4        IF FIRST CHARACTER IS OCTAL
004.327 041 001 040 1797     LXI  H,START+1    ELSE, INPUT STARTING ADDRESS
004.332 026 055 1799     MVI  D,-          FIRST BYTE MUST END WITH A DASH
004.334 315 042 003 1800     CALL  IOA
004.337 041 035 040 1801     LXI  H,ABUS+1    ENTER ENDING ADDRESS
004.342 067 1802     STC          SHOW NO CHARACTER IN (A)
004.343 077 1803     CMC
004.344 026 015 1804     MVI  D,A,CR      LAST CHARACTER MUST BE A RETURN
004.346 315 062 003 1805     CALL  IOA
004.351 076 012 1806     MVI  A,10        GET USER PC VALUE FOR DISPLAY

```

```

004.353 315.052.003 1807 CALL LRA,
004.354 136 1808 MOV E,M
004.357 043 1809 INX R,
004.360 126 1810 MOV D,M
004.361 353 1811 XCHG
004.362 315 313.005 1812 CALL TOA
004.365 303 374.001 1813 JMP WHEN
1814

```

```

1816 ** SUBM ... SUBSTITUTE MEMORY.
1817 *
1818 * SUBM INPUTS A MEMORY ADDRESS FROM THE CONSOLE AND THEN DISPLAYS
1819 * THAT ADDRESS AND ITS CONTENTS. IF A CARRIAGE RETURN IS THEN TYPED,
1820 * CONTROL RETURNS TO THE MONITOR. IF A SPACE IS TYPED, THE NEXT
1821 * MEMORY LOCATION AND CONTENTS ARE DISPLAYED. IF A MINUS SIGN IS
1822 * TYPED, THE PREVIOUS MEMORY LOCATION AND CONTENTS ARE DISPLAYED.
1823 * IF AN OCTAL CHARACTER IS TYPED, A BYTE IS ENTERED AND PLACED AT THE
1824 * CURRENT MEMORY LOCATION.
1825 *
1826 * ENTRY NONE
1827 * EXIT NONE
1828 * USES A,E,H,L,F
1829 *
1830
1831

```

```

004.370 041 201.006 1832 SUBM LXI H,MSG,SUB COMPLETE SUBSTITUTE MESSAGE.
004.373 315 100.006 1833 CALL TYPMSO
004.376 315 150.005 1834 CALL IRQC INPUT FIRST CHARACTER.
005.001 320 1835 RNC IF A RETURN, EXIT
1836
005.002 041 003.040 1837 LXI H,IOURK+1 ELSE, INPUT STARTING ADDRESS
005.005 026 015 1838 MVI D,A,CR ENDING WITH A RETURN
005.007 315 042.003 1839 CALL IOA
005.012 353 1840 XCHG (H,L) = INPUT ADDRESS.
1841
005.013 315 313.005 1842 SUBM1 CALL IOA TYPE,CSLF, ADDRESS, AND A SPACE
005.016 176 1843 MOV A,M GET MEMORY CONTENTS FOR DISPLAY
005.017 315 343.005 1844 CALL IOB
005.022 076 040 1845 MVI A, SPACE
005.024 315 302.003 1846 CALL WCC
1847
005.027 315 301.005 1848 SUBM2 CALL IOC INPUT FIRST CHARACTER.
005.032 322 075.005 1849 JNC SUBM7 IF FIRST CHARACTER IS OCTAL
1850
005.035 376 040 1851 CPI SPACE7
005.037 302 046.005 1852 JNZ SUBM4 IF NOT A SPACE.
1853
005.042 043 1854 SUBM3 INX H POINT TO NEXT ADDRESS.
005.043 303 013.005 1855 JMP SUBM1 DISPLAY NEXT
1856
005.046 376 055 1857 SUBM4 CPI '-' MINUS?
005.050 302 062.005 1858 JNZ SUBM6 IE NOT
1859

```

MTR88 - H88 MONITOR #09.00.00. HEATH HBASM V1.4 01/20/78 PAGE 38  
 ADDED TASK TIME ROUTINES FOR H88/H89 08:59:25 17-MAY-79

Address	Instruction	Comment
005.053	CALL	ECHO HYPHEN
005.056	DCX	POINT TO PREVIOUS ADDRESS
005.057	JMP	DISPLAY PREVIOUS
005.062	CPI	RETURN?
005.064	RZ	IF RETURN, EXIT
005.065	MVI	ELSE, DING BELL
005.067	CALL	TRY AGAIN
005.072	JMP	
005.075	MVI	ZERO BYTE TO BE BUILT
005.077	CALL	ECHO OCTAL CHARACTER
005.102	ANI	GET BINARY VALUE
005.104	MOV	SAVE PARTIAL
005.105	MOV	GET CURRENT
005.106	RLC	MAKE ROOM FOR NEW CHARACTER
005.107	RLC	
005.110	RLC	
005.111	ANI	TOSS PREVIOUS LSB
005.113	ORA	ADD NEW
005.114	MOV	SAVE NEW TOTAL
005.115	CALL	INPUT NEXT CHARACTER
005.120	JNC	IF OCTAL
005.123	CPI	SPACE?
005.125	JZ	IF SPACE, DISPLAY NEXT BYTE
005.130	CPI	MINUS?
005.132	JZ	IF MINUS, DISPLAY PREVIOUS
005.135	CPI	RETURN?
005.137	RZ	IF RETURN, EXIT
005.140	MVI	ELSE, DING BELL
005.142	CALL	TRY AGAIN
005.145	JMP	
1899	**	IRDC - INPUT A RETURN OR AN OCTAL CHARACTER
1900	*	
1901	*	IRDC INPUTS A CHARACTER FROM THE CONSOLE AND WAITS UNTIL IT
1902	*	RECEIVES EITHER A VALID OCTAL CHARACTER OR A CARRIAGE RETURN
1903	*	
1904	*	ENTRY NONE
1905	*	EXIT (A) = INPUT CHARACTER
1906	*	'C' = SET IF CHARACTER IS OCTAL
1907	*	USES A:F
1908	*	
1909	*	
005.150	IRDC	INPUT CHARACTER
005.153	CALL	RETURN?
005.155	CPI	IF A CR
	RZ	

```

1913
005.156 376 060 CPI '0'
005.160 332 166 005 JC IROC1 IF < OCTAL
1916
005.163 376 070 CPI '8'
005.165 330 RC > 8?
IF OCTAL
1919
005.166 076 007 MVI A:A,BELL ELSE, RING,BELL
005.170 315 302 003 CALL WCC
005.173 303 150 005 JMP IROC TRY AGAIN
1922
1924 ** IOA1 - INPUT OCTAL ADDRESS
1925 *
1926 * IOA1 IS A CONTINUATION OF *IOA* AND INPUTS A SPLIT OCTAL ADDRESS
1927 * WITHOUT REQUIRING LEADING ZEROS
1928 *
1929 * ENTRY (H,L) = ADDRESS + 1 WHERE INPUT ADDRESS IS TO BE PLACED
1930 * (A) = FIRST OCTAL CHARACTER IF 'C' IS SET
1931 * (D,E) = INPUT ADDRESS
1932 * (A) = LAST INPUT CHARACTER
1933 * USES A:P,E;H:L,F
1934
1935
005.176 305 IOA1 PUSH B SAVE (B,C)
005.177 192 MOV R:R (B) = TERMINATION CHARACTER
005.200 345 FUSH H SAVE ADDRESS WHERE INPUT IS TO BE PLACED
005.201 041 000 000 LXI H:0 SET NEW VALUE TO ZERO
005.204 324 262 003 CNC RCC IF CARRY SET, FIRST CHARACTER IS IN ACC
005.207 376 060 CPI '0' MAKE SURE CHARACTER IS OCTAL
005.211 332 242 005 JC IOA3 IF < OCTAL
1943
005.214 376 070 CPI '8'
005.216 322 242 005 JNC IOA3 IF > OCTAL
1946
005.221 315 302 003 CALL WCC ECHO OCTAL CHARACTER
005.224 346 007 ANI 00000111B GET BINARY VALUE
005.226 365 FUSH PSM SAVE NEW CHARACTER VALUE
005.227 051 DAD H SHIFT THREE TO MAKE ROOM FOR NEW CHARACTER
005.230 051 DAD H
005.231 051 DAD H
005.232 365 FUSH PSM SAVE CARRY FROM DAD
005.233 321 POP D SAVE FLAG RESULT IN E
005.234 361 POP PSM RETURN NEW CHARACTER VALUE TO (A)
005.235 205 ADD L:A
005.236 157 MOV L:A
005.237 303 204 005 JMP IOA2 SEE IF MORE CHARACTERS
1959
005.242 270 IOA3 CMP B TERMINATING CHARACTER?
005.243 312 260 005 JZ IOA4 IF EQUAL
1962
005.246 076 007 MVI A:A,BELL ELSE, RING,BELL
005.250 315 302 003 CALL WCC
005.253 067 STC TRY AGAIN

```



MTR88 - H88 MONITOR #09.00.00. HEATH HBASH V1.4 01/20/78 PAGE 40  
 ADDED TASK TIME ROUTINES FOR H88/H89 08:59:26 17-MAY-79 IOAI

```

005.254 077 CMC
005.255 303 204 005 JMP IOA2
1966
1967
1968
1969 * END OF INPUT, PUT VALUE IN MEMORY AND EXIT
1970
005.260 315 302 003 IOA4
005.263 127 WCC ECHO CHARACTER
005.264 325 MOV D,A LAST CHARACTER TO D
1973
005.265 341 POP PSW (PSW) = RESULT OF DAD
005.266 174 MOV A,H MAKE (H) INTO SPLIT OCTAL
1975
005.267 037 RAR
005.270 147 MOV H,A
1977
005.271 172 MOV A,D RESTORE LAST INPUT CHARACTER
005.272 353 XCHG (D,E) = INPUT ADDRESS
1979
005.273 341 POP H (H,L) = LOCATION TO PLACE THIS ADDRESS
1980
005.274 162 MOV M,D
1981
005.275 053 DCX H
1982
005.276 163 MOV M,E
1983
005.277 301 POP B RESTORE (B,C)
1984
005.300 311 RET
1985
    
```

```

1987 ** IOC - INPUT OCTAL CHARACTER
1988 *
1989 *
1990 * ENTRY NONE
1991 * EXIT (A) = INPUT CHARACTER
1992 * (C) = SET IF CHARACTER NOT OCTAL
1993 * USES A,F
1994
1995
1996 IOC INPUT CHARACTER
1997
005.301 315 262 003 CALL RCC
005.304 376 060 CPI '0'
005.306 330 RC IF CHARACTER < OCTAL
1998
005.307 376 070 CPI 'B' CHARACTER > OCTAL?
005.311 077 CMC 'C' IF GREATER THAN
005.312 311 RET
    
```

```

2004 ** IOA - TYPE OCTAL ADDRESS
2005 *
2006 * IOA OUTPUTS TO THE CONSOLE A CRLF, THE SPECIFIED ADDRESS AND A SPACE
2007 *
2008 * ENTRY (H,L) = ADDRESS TO BE DISPLAYED
2009 * EXIT NONE
2010 * USES A,B,C,F
2011
2012
2013 IOA MVI A,A,CR CRLF
005.313 076 015 CALL WCC
005.315 315 302 003 MVI A,A,LF
005.320 076 012
    
```

```

005.322 315.302.003 2014 CALL MCC
005.325 174 2017
005.326 315.343.005 2018 TOA MOV A,H ADDRESS
005.331 175 2019 CALL TOB
005.332 315.343.005 2020 MOV A,L
005.335 076.040 2021 CALL TOB
005.337 315.302.003 2022 MOV A,' ' SPACE
005.342 311 2023 CALL MCC
2024 RET
2025

```

```

2027 ** TOB - TYPE OCTAL BYTE
2028 *
2029 * TOB OUTPUTS TO THE CONSOLE IN OCTAL, THE BYTE IN A
2030 *
2031 * ENTRY (A) = BYTE TO BE OUTPUT
2032 * EXIT NONE
2033 * USES A,F
2034 *
2035
005.343 305 2036 TOB PUSH B
005.344 006.002 2037 MOV B,2 NUMBER OF CHARACTERS - 1
005.346 117 2038 MOV C,A SAVE ORIGINAL BYTE
005.347 267 2039 ORA A ASSURE 'C' = ZERO
005.350 037 2040 RAR
005.351 037 2041 RAR SHIFT TOP BYTE TO LSB
005.352 037 2042 RAR SHIFT MIDDLE BYTE TO LSB
005.353 037 2043 RAR
005.354 037 2044 RAR
005.355 037 2045 RAR
005.356 346.007 2046 RAR
005.359 366.060 2047 ANI 00000111B MASK FOR HALF ASCII
005.362 315.302.003 2048 ORI 00110000B MAKE WHOLE ASCII
005.365 171 2049 CALL WCC OUTPUT TO CONSOLE
005.366 005 2050 MOV A,C GET ORIGINAL BYTE
005.367 302.353.005 2051 JNZ TOB1 IF SECOND BYTE STILL NEEDS TO BE OUTPUT
005.372 346.007 2052 ANI 00000111B
005.374 366.060 2053 ORI 00110000B ELSE, OUTPUT LAST CHARACTER
005.376 315.302.003 2054 CALL MCC
006.001 301 2055 POP B
006.002 311 2056 RET
2057

```

```

2059 ** WCR - WAIT FOR A CARRIAGE RETURN
2060 *
2061 * WCR INPUTS CHARACTERS FROM THE CONSOLE UNTIL A CARRIAGE RETURN
2062 * IS RECEIVED, AND THEN ECHOS A.CRLF.
2063 *
2064 * ENTRY NONE
2065 *

```

MTR88 - H88 MONITOR #09.00.00. HEATH HBASH V1.4 01/20/78 PAGE 42  
 ADDED TASK TIME ROUTINES FOR H88/H89 08159:27 17-MAY-79

2066 \* EXIT NONE  
 2067 \* USES A,F  
 2068  
 2069  
 006.003 315 262 003 WCR CALL RCC INPUT CHARACTER  
 006.006 376 015 2071 CPI A,CR  
 006.010 302 003 006 JNZ WCR IF NOT A CR  
 2073  
 006.013 315 302 003 2074 CALL WCC ELSE, ECHO CR  
 006.016 078 012 2075 MVI A,A:LF LINE-FEED  
 006.020 315 302 003 2076 CALL WCC  
 006.023 311 RET

2079 \*\* TPDSP - TAPE DISPLAY  
 2080 \*  
 2081 \* SHOW H88 USER THAT THERE IS SOME ACTIVITY DURING A LOAD OR A DUMP  
 2082  
 006.024 042 024 040 2083 TPDSP SHLD ABUSS UPDATE ABUSS  
 2084  
 006.027 076 015 2085 MVI A,A:CR RETURN  
 006.031 315 302 003 2086 CREL WCC  
 2087  
 006.034 174 2088 MOV A,H ADDRESS  
 006.035 315 343 005 2089 CALL TOB  
 006.040 175 2090 MOV A,L  
 006.041 315 343 005 2091 CALL TOB  
 2092  
 006.044 311 2093 RET

2095 \*\* HRNX - HORN EXTENSION ROUTINE  
 2096 \*  
 2097 \* THIS IS AN EXTENSION TO \*HORN\* TO MAKE ROOM FOR A JUMP  
 2098  
 006.045 056 011 2099 HRNX MVI L,#CTLFLG TURN OFF HORN  
 006.047 163 2100 MOV M,E  
 006.050 321 2101 POP D  
 2102 POP H  
 006.051 341 2103 RET

2105 \*\* NOISE - DING BELL ON CONSOLE  
 2106 \*  
 2107 \* THIS IS A MODIFICATION TO ALLOW THE H88/H89 TO USE THE CONSOLE BELL  
 2108  
 006.053 076 007 2109 NOISE MVI A,A:BEL  
 006.055 315 302 003 2110 CALL WCC  
 006.060 303 140 002 2111 JMP HORN CONTINUE WITH NORMAL HORN DELAY

```

2113 ** TPERMSG - TAPE ERROR MESSAGE
2114 *
2115 * DISPLAY THE TAPE ERROR NUMBER ON THE CONSOLE
2116
006.063 062.024.040 2117 TPERMSG STA ABUSS SAVE ERROR NUMBER
006.066 076.040 2118 MVI A, ' ' OUTPUT A SPACE
006.070 315.302.003 2119 CALL WCC
006.073 170 2120 MOV A,B
006.074 315.343.005 2121 CALL TOB OUTPUT NUMBER
006.077 311 2122 RET
    
```

```

2124 ** TYPMSG - TYPE MESSAGE TO CONSOLE
2125 *
2126 * TYPMSG OUTPUTS AN ASCII MESSAGE FROM MEMORY TO THE CONSOLE
2127 * UNTIL A NULL IS SENSED
2128 *
2129 * ENTRY (H,L) = ADDRESS OF MESSAGE
2130 * EXIT NONE
2131 * USES A,H,L,F
2132
2133
006.100 176 2134 TYPMSG MOV A,M GET CHARACTER
006.101 267 2135 ORA A SEE IF A NULL
006.102 310 2136 RZ IF NULL, EXIT
    
```

```

006.103 315.302.003 2138 CALL WCC ELSE OUTPUT CHARACTER TO CONSOLE
006.106 043 2139 INX H POINT TO NEXT CHARACTER
006.107 303.100.006 2140 JMP TYPMSG OUTPUT IT
    
```

```

2142 ** MSG.PR - MESSAGE FOR MONITOR PROMPT
2143 *
2144 * CRLF, ' H: '
2145
2146
006.112 015.012.040 2147 MSG.PR DB 'A,CR,A,LF,' H: ',0
    
```

```

2149 ** MSG.SP - MESSAGE TO TELL USER TO TYPE SPACES
2150 *
2151 * 'Type spaces to determine baud rate'
2152
006.122 124 171 160 2153 MSG.SP DB 'Type SPACES to determine baud rate',0
    
```

MTR88 - H88 MONITOR #09.00.00. HEATH HBASM V1.4 01/20/78 PAGE 44  
ADDED TASK TIME ROUTINES FOR H88/H89 MSG.GO 08:59:27 17-MAY-79

2155 \*\* MSG.GO - (G)O  
2156 \* 'GO'  
2157 \* 'GO'  
2158

006.165 157 040 000 2159 MSG.GO DB 'o',o

2161 \*\* MSG.LD - (L)O'D  
2162 \* 'O'D'  
2163 \* 'O'D'  
2164  
006.170 157 141 144 2165 MSG.LD DB 'o'd',o

2167 \*\* MSG.DMP - (D)UMP  
2168 \* 'UMP'  
2169 \* 'UMP'  
2170  
006.174 165 155 160 2171 MSG.DMP DB 'ump',o

2173 \*\* MSG.SUB - (S)UBSTITUTE  
2174 \* 'SUBSTITUTE'  
2175 \* 'SUBSTITUTE'  
2176  
006.201 165 142 163 2177 MSG.SUB DB 'ubstitute',o

2179 \*\* MSG.FC - (P)ROGRAM\_COUNTER  
2180 \* 'PROGRAM\_COUNTER'  
2181 \* 'PROGRAM\_COUNTER'  
2182  
006.214 162 157 147 2183 MSG.FC DB 'rogram Counter',o

2185 \*\* MSG.BT - (B)OOT  
2186 \* 'BOOT'  
2187 \* 'BOOT'  
2188  
006.234 157 157 164 2189 MSG.BT DB 'oot',o  
LON L

2481 \*\* ENTRY POINT FOR FLOPPY DISK ROTATIONAL SPEED TEST  
2482 \*  
000.000 ERRNZ...10000A-6-\* MUST BE SIX BYTES BEFORE END  
2483  
2484  
007.372 303.240.006 2485 ESPEED JMP SPEED

2487 \*\* ENTRY POINT FOR DYNAMIC MEMORY TEST  
2488 \*  
000.000 ERRNZ 10000A-3-\* MUST BE THREE BYTES BEFORE END  
2489  
2490  
007.375 303 116 007 2491 EDYMEM JMP DYMEM  
2492  
000.000 ERRNZ \*-10000A MUST NOT EXCEED 2K BYTES  
2493  
2494



MTR88 - H88 MONITOR #09.00.00. HEATH HBASM V1.4 01/20/78 PAGE 46  
 RAM CELLS 08:59:30 17-MAY-79

```

2497 ** THE FOLLOWING ARE CONTROL CELLS AND FLAGS USED BY THE KEYSET...
2498 * MONITOR.
2499
2500 ORG 40000A 8192
2501 START DS 2 DUMP STARTING ADDRESS
2502 IDWRK DS 2 IN OR OUT INSTRUCTION
2503 PRSRAM EQU * FOLLOWING CELLS INITIALIZED FROM ROM
2504 DS 1 KEY
2505
2506 REGI DS 1 INDEX OF REGISTER UNDER DISPLAY
2507 DSPROT DS 1 PERIOD FLAG BYTE
2508 DSPROD DS 1 DISPLAY MODE
2509
2510 MFLAG DS 1 USER FLAG OPTIONS
2511 * SEE *UO.XXX* BITS DESCRIBED AT FRONT
2512
2513 CTLFLG DS 1 FRONT PANEL CONTROL BITS
2514 REFIND DS 1 REFRESH INDEX (0 TO 7)
2515 PRSL EQU *-PRSRAM END OF AREA INITIALIZED FROM ROM
2516
2517 PFLEDS EQU * FRONT PANEL LED PATTERNS
2518 ALEDS DS 1 ADDR 0
2519 DS 1 ADDR 1
2520 DS 1 ADDR 2
2521
2522 DS 1 ADDR 3
2523 DS 1 ADDR 4
2524 DS 1 ADDR 5
2525
2526 DLEDS DS 1 DATA 0
2527 DS 1 DATA 1
2528 DS 1 DATA 2
2529
2530 ABUSS DS 2 ADDRESS BUSS
2531 RCCA DS 1 RCC SAVE AREA
2532 CRCSUM DS 2 CRC-16 CHECKSUM
2533 TPERRX DS 2 TAPE ERROR EXIT ADDRESS
2534 TICCNT DS 2 CLOCK TIC COUNTER
2535
2536 REGPTR DS 2 REGISER CONTENTS POINTER
2537
2538 UIVEC DS 0 USER INTERRUPT VECTORS
2539 DS 3 JUMP TO CLOCK PROCESSOR
2540 DS 3 JUMP TO SINGLE STEP PROCESSOR
2541 DS 3 JUMP TO I/O 3
2542 DS 3 JUMP TO I/O 4
2543 DS 3 JUMP TO I/O 5
2544 DS 3 JUMP TO I/O 6
2545 DS 3 JUMP TO I/O 7
2546
2547 ** H88/H89 RAM USAGE BEYOND THAT OF MONTRF
2548 *
2549 NMIRET DS 2
2550 END
    
```

ASSEMBLY COMPLETE.  
 2550 STATEMENTS  
 0 ERRORS DETECTED  
 12984 BYTES FREE

MTRBB - HBB MONITOR #09.00.00.

CROSS REFERENCE TABLE

XREF. V1.1

PAGE 47

000004	3435	5045	6735	8155	16295	1630
.MELAG	040010	504	617	625	2510L	
A.BEL	000007	564	658	1540	1867	1895
A.BKS	000010	122E	2397			1920 1963 2109
A.CR	000015	124E	699	776	1349	1429 1804 1838 1864 1892 1911 2013
A.DEL	000177	126E	2071	2085	2147	2282 2467
A.FSC	000033	125E	2281	2290	2292	2298 2300 2455
A.LF	000012	123E	699	699	779	1429 2015 2075 2281 2282 2467
A.STX	000002	120E	920	1134		
A.SYN	000026	119E	915	1132		
ABUSS	040024	713	928	1801	2083	2117 2530L
AC.DLY	000110	244E				
ALARM	002136	629	988E	1063		
ALARMB	002167	989	1014L			
ALEIS	040013	2518L				
BLNSIZ	002000	154E				
BOOT	004256	690	1760L			
BR19.2	004077	1625L				
BR38.4	004101	1626L				
BR56.0	004103	1627L				
BR76	004075	1624L				
BRTAR	004075	1579	1622E	1630		
CB.CLI	000100	132E	188	591	624	1729
CB.MTL	000040	131E	490	543	624	817
CB.SPK	000200	133E	391	624	994	
CB.SSI	000200	130E	391	490	624	802
CLN4	000270	550	574E			813 1737
CLONK	000201	327	328			
CRG	002347	1198L	1261	532L		
CRG1	002356	1202L				
CRG2	003004	1213	1220L			
CRCSUM	040027	923	965	1031	1140	1201 1223 2532L
CYC	002172	885	1030L			
CYLFLG	040011	343	536	541	546	625 801 804 815 999 1637 2099 2513L
CUI1	000165	505L	562			
DELS	040021	2526L				
DLY	000053	381L				
DM.MR	000000	137E				
DM.MW	000001	138E				
DM.RR	000002	139E				
DM.RW	000003	140E				
DOD	003122	1334L				
HS.HOLE	000001	2216E	2241	2247		
ISPMOD	040007	2508L				
ISPROT	040006	2507L				
RUHF	002002	913L				
DY10.5	007245	1544	2424L			
DY3.3	007153	2341	2345L			
DY3.5	007163	2349	2353L			
DY3.7	007173	2357	2361L			
DY5.53	007251	2403	2407L			
DY9.3	003315	597	1506L			
DY9.4	003326	1511	1515L			
DY9.5	003335	1519	1523L			
DY9.8	003350	1528	1532L			
DYASC	003143	1363E	1377	1410	1420	1546 2405 2453



MTR88 - H88 MONITOR #09.00.00.  
 CROSS REFERENCE TABLE

XREF VI.1  
 PAGE 48

DYASCI	003144	1366L	1368				
DYBT	003160	1383L	1513				
DYBT.2	003202	1395	1521	1538	2351	2359	2417
DYBT.4	003221	1408					
DYBT.6	003235	1418					
DYME5	007242	2399E	2408				
DYMEM	007113	1413	2319L	2491			
DYMEM1	007122	2324L					
DYMEM10	003360	1536	1540L	2435			
DYMEM11	007272	2426L	2427	2430	2433		
DYMEM2	007127	2328L	2332				
DYMEM3	007140	2329	2335L				
DYMEM4	007207	588	2370	2374L			
DYMEM5	007212	2375L	2387	2391			
DYMEM6	000273	582L	2415				
DYMEM7	000276	583L	586				
DYMEM9	000307	592L	2377	2382			
DYMSG	007306	599	1530	2343	2372	2447L	2457
DYMSG.5	007316	2451	2455L				
EDYMEM	007375	2471L					
ERRR	000322	438	588	618E	1082	1481	
ESPEED	007372	2485L					
FPLEDS	040013	2517E					
GO	001222	781	791L				
GO	000043	391L	791	1775			
GO88	001146	675	764L				
GO88.1	001177	769	778L				
H88.CTL	000362	102E	1573	1744	2320		
H88.SW	000362	106E	1580	1611			
H88B.CK	000002	103E					
H88B.SS	000001	104E					
H88S.BR	000300	107E	1581				
H88S.M	000040	108E	1612				
HORN	002140	993L	2111				
HRN0	002143	383	996L				
HRN2	002160	1007L	1008				
HRNX	006045	1010	2099L				
INIT	000073	314	316	413L	417		
INIT0	000000	312L					
INIT0.0	000003	313L	1618				
INITOX	004000	312	1572L				
INITOX1	004050	1603L	1604	1607			
INIT1	000107	426L	431				
INIT2	000117	433L					
INT1	000010	321E					
INT2	000020	337E					
INT3	000030	356L					
INT4	000040	363L					
INT5	000050	370L					
INT6	000060	388L					
INT7	000070	397L					
INTXIT	000172	514L	544	806			
IDA	003062	756	777	1293L	1800	1805	1839
IGA1	005176	1293	1936L				
IGA2	005204	1940L	1958	1967			
IGA3	005242	1942	1945	1940L			
IGA4	005260	1961	1961	1971L			
IOB	003066	1308L					

MTR88 - H88 MONITOR #09.00.00.  
 CROSS REFERENCE TABLE

XREF VI.1  
 PAGE 49

IOB1	003070	1309L	1316	1345	1352		
IOB1.5	003126	1327	1342L				
IOB2	003135	1314	1349L				
IOC	005301	1848	1883	1996L			
IOWRK	040002	1837	2225	2265	2502L		
IP_DS	000177	2212E	2240	2246			
IP.PAD	000360	96E	1065	1098			
IP.TFC	000371	112E	1100				
IP.TFD	000370	114E	1178				
IRCC	005150	745	749	768	1795	1834	1910L 1922
IRCC1	005166	1915	1920L				
LDA0	001272	846L	893				
LDA1	001342	872L	883				
LOAD	001267	844E					
LRA	003047	1278L					
LRA.	003052	555	739	773	863	940	1277L 1764 1807
MI.ANI	000346	170E	1055				
MI.EXAF	000010	177E	1365	1371			
MI.HLT	000166	165E	561				
MI.IN	000333	167E	1709	1718			
MI.JIXA	000335	178E	1424	2460			
MI.JIXB	000351	179E	1424	2460			
MI.JIYA	000375	180E	1374				
MI.JIYB	000351	181E	1374				
MI.JMP	000303	172E					
MI.LDA	000072	189E					
MI.LDXA	000335	173E	596	1510	1518	1527	1535 2340 2348 2358 2369 2414
MI.LDXB	000041	174E	596	1510	1518	1527	1535 2340 2348 2356 2369 2414
MI.LDYA	000375	175E	1394	1407	1417	1543	2402 2450
MI.LDYB	000041	176E	1394	1407	1417	1543	2402 2450
MI.LXID	000021	171E					
MI.OUT	000323	188E	1706	1725			
MI.RET	000311	166E	1564				
MSG.BT	006234	1760	2189L				
MSG.DMF	006174	1793	2171L				
MSG.ER	007362	1524	2474L				
MSG.ERR	001047	593	699L				
MSG.G0	006165	766	2159L				
MSG.HSS	007100	2268	2298L				
MSG.LD	006170	720	2165L				
MSG.PAS	003237	1429L	2366				
MSG.PC	006214	736	2183L				
MSG.PR	006112	642	2147L				
MSG.RAM	007324	2337	2465L				
MSG.SP	006122	1772	2153L				
MSG.SFD	006371	2222	2281L				
MSG.SUB	006201	1832	2177L				
MSG.WRK	007062	2262	2390L				
MTR	000344	636E	818				
MIR.2	000357	645L	660				
MIR.3	000371	649L	656				
MIR.4	001014	650	662L				
MIR1	000345	639E	640				
MIRA	001025	647	672E	692			
MTRAL	000006	648	692E				
NMI	004116	683	1678L				
NMIO.5	004160	1696	1699				
NMII	004202	1691	1716L				

MTRBB - HBB MONITOR #09.00.00.  
 XREF V1.1  
 PAGE 50

CROSS REFERENCE TABLE

NMI1.5	004217	1719	1725L		
NMI2	004224	1728L			
NMI2.5	004251	1702	1707	1710	1726
NMI3	004252	1714	1723	1748L	1746L
NMIENT	000146	483L			
NMIRET	040064	1679	1686	2549L	
NOISE	008053	1014	2109L		
ONDR0	000022	2220E	2226		
OP.DC	000177	97E	539	803	814
OP.CYL	000360	2211E	2227		
OP.DIG	000360	98E			
OP.SEG	000361	99E			
OP.TPC	000371	113E	444	914	978
OP.TPD	000370	115E	1260	1081	1172
PCA	001103	687	736L	1258	
PCAI	001137	746	754L		
PRSL	000007	313	2515E		
PRSRAM	040004	313	2503E	2515	
PRSR0M	003371	1557E	1617		
RCC	003262	645	1309	1472E	1910
RCCI	003262	1474L	1476	1940	1996
RCCA	040026	2531L		2076	
RCK	003260	1451E			
REFIND	040012	2514L			
REGI	040005	1276	2506L		
RESPTR	040035	494	627	1279	2536L
RMEM	001261	723	826L		
RNB	002331	872	1130	1156	1171L
RNB1	002335	1173L	1175		
RNP	002325	858	968	1030	1141
ROMDD	030000	26E	1765	1156L	
RT.BD	000005	149E			
RT.BP	000002	146E			
RT.CI	000003	147E			
RT.HI	000001	145E	845	924	
RT.NB	000004	148E			
RT.PD	000006	150E			
SAE	001063	713L			
SAVALL	000132	325	341	460L	
SAVALLR	000151	485	487E	1639	
SAVALLX	004105	475	1634E		
SC.ACE	000350	243E	1366	1372	1474
SC.UART	000372	203E	2248	1478	1495
SINCR	004000	419E	421	422	1500
SPEED	006240	2222L	2485	1578	1590
SPEED1	006257	2228L	2272	1593	1595
SPEED2	006275	2236	2239L		
SPEED3	006300	2240L	2242	2253	
SPEED4	006307	2246L	2248		
SPEED5	006357	2266	2269L		
SRMEM	001067	678	720L		
SRS	002245	846	1126E		
SRS1	002265	1127L	1135	1139	
SRS2	002271	1130L	1133		
SST1	001235	392	804L		
SSTEP	001235	799E			
START	040000	422	870	926	1798
STPRTN	001244	344	2324	2374	2501L
					812E

SUBM	004370	684	1832L				
SUBM1	005013	1842L	1855	1862			
SUBM2	005027	1848L	1869				
SUBM3	005042	1854L	1887				
SUBM4	005046	1852	1857L				
SUBM5	005053	1860L	1890				
SUBM6	005062	1858	1864L				
SUBM7	005075	1849	1871L				
SUBM8	005077	1873L	1884				
SUBM9	005115	1883L	1897				
SWMEM	004313	681	1773L				
SWMEM2	004337	1801L					
SWMEM4	004351	1796	1806L				
TD.IN	000370	158E					
TD.OUT	000370	159E					
TER1	002220	1063L	1070				
TER3	002215	1056L	1067				
IFT	002133	82	977L	1051			
TICCNT	040033	532	534	1068	2228	2257	2534L
TOA	005313	748	1812	1842	2013L		
TOA.	005325	2018L	2271				
TOR	005343	1844	2019	2021	2036L	2089	2091 2121
TOR1	005353	2043L	2051				
TPART	002244	826	908	1080L			
TPDSP	006024	877	955	2083L			
TPRMSG	006063	1050	2117L				
TPERR	002205	857	1049L				
TPERRX	040031	827	909	1102	2533L		
TPXIT	002252	1064	1098L	1173	1254		
TS.IN	000371	160E					
TS.OUT	000371	161E					
TYFMSG	006100	643	721	737	767	1761	1773 1794 1833 2134L 2140 2223 2269
UC.2SB	000004	269E					
UC.SFW	000000	265E					
UC.GRW	000001	266E					
UC.ZBW	000002	267E					
UC.BEW	000003	268E	1594				
UC.BI	000020	288E					
UC.CTS	000020	297E					
UC.PCS	000001	293E					
UC.DDR	000002	294E					
UC.DLA	000200	274E	1577				
UC.DR	000001	284E	1475				
UC.DRL	000010	296E					
UC.DSR	000040	298E					
UC.PTR	000001	277E					
UC.EDA	000001	255E					
UC.EFS	000020	271E					
UC.FE	000010	287E					
UC.IID	000006	262E					
UC.IIF	000001	261E					
UC.LQO	000020	281E					
UC.MSI	000010	258E					
UC.OR	000002	285E					
UC.OU1	000004	279E					
UC.OU2	000010	280E					
UC.FE	000004	286E					
UC.FEN	000010	270E					



MTRBB - H8B MONITOR #09.00.00.

XREF V1.1

CROSS REFERENCE TABLE

PAGE 53

MEM	001374	907E	1813						
MNR	003024	917	921	951	1239	1253L			
MNF1	003025	1254L	1256						
MNP	003017	925	936	945	948	966	967	1238L	

24284 BYTES FREE.

## APPENDIX B

### MTR-88 DEMO

The sample program that follows shows some of the advanced features that are available to you with MTR-88. The program is not designed to be efficient or particularly useful by itself. It uses the H88 clock, console terminal, and interrupt capability to create an accurate interval timer that will time up to 377(octal) seconds. When the interval ends, the H88 audio alarm is sounded.

Use the H88 keyboard and the "Substitute" command to enter the machine code and start the program. You will also use the keyboard to enter the octal time.

The demo uses the MTR-88 firmware (program in a ROM) for most of the working routines, and you should look up the details of these routines (in Appendix A). The listing of the demo was prepared using the text editor and assembler that are available for the H88. However, the program should be loaded by hand using the "Substitute" command.

## THE SAMPLE PROGRAM

This program initially blanks the screen and then waits for you to enter an octal value. The MTR-88 routine WCC is used to send characters to the screen, and IOB is used to Input an Octal Byte.

The most subtle part of the program is the interrupt processing. First, a jump to the interrupt processor is planted in UIVC to allow processing of the clock interrupts. Then .MFLAG is set so MTR-88 will pass interrupts to the program. Finally, interrupts are enabled.

The main part of the program is a "do-nothing" loop that waits for the time to count down to zero. When the time is exhausted, the program restores the original state of .MFLAG and stops.

The interrupt processor keeps its own local TICCNT and counts it down from 500. When this count reaches zero, one second has elapsed and the new reduced time is displayed on the screen using TOB (Type Octal Byte). The local TICCNT is reset to 500. When the time is exhausted, the main program stops clock processing, so the processor is not called again.



HEATH ASM.#104,02,00.  
Page 1

\*\*\* \*\*\*\*\*

DEMO: MTR-88

SYSTEM DEFINITIONS

```

002.136 EQU 2136A MAKE NOISE
003.302 EQU 3302A WRITE CHAR TO CONSOLE
040.010 EQU 40010A USER FLAG OPTIONS
040.037 EQU 40037A USER INTERRUPT VECTOR
000.001 EQU 1A ALLOW CLOCK INTERRUPT PROCESSING
000.303 EQU 303A MACHINE INSTRUCTION JUMP
003.066 EQU 3066A INPUT OCTAL BYTE
005.343 EQU 5343A TYPE OCTAL BYTE

```

```

000.033 EQU 33A
000.015 EQU 15A
000.012 EQU 12A

```

040.100 EQU 40100A

\*\*\* \*\*\*\*\*

ERASE SCREEN

```

040.100 MTR88 MVI A,ESC ESCAPE SEQUENCE TO
040.102 CALL WCC
040.105 MVI A,'E' ERASE SCREEN
040.107 CALL WCC

```

\*\*\* \*\*\*\*\*

READ A OCTAL INTEGER FROM KEYBOARD

```

040.112 LXI H,NUMBER SET ADDRESS OF NUMBER
040.115 ANA A CLEAR CARRY (SIDE EFFECT)
040.116 CALL IOB INPUT OCTAL BYTE
040.121 CALL SETICK SETUP TICK TO 500 FOR ONE SEC

```

\*\*\* \*\*\*\*\*

INITIALIZE SERVICE INTERRUPT ROUTINE

```

040.124 MVI A,MI,JMP SET-UP JUMP INSTRUCTION
040.126 STA UIVEC STORE ,JMP, INSTRUCTION
040.131 LXI H,INTRP USER INTERRUPT ADDRESS
040.134 SHLD UIVECHI POSITIONED
040.137 MVI A,00,CLK
040.141 STA .MFLAG ENABLE CLOCK INTERRUPT

```

HEATH ASM.#104.02.00.  
Page 2

\*\*\* \*\*\*\*\*

\* \* \* \* \* WAIT FOR CLOCK TO REACH ZERO

```

040.144 072 234 040 LOOP      DO NOTHING LOOP,
040.147 376 000 CPI 0        WAIT FOR END
040.151 302 144 040 JNZ LOOP  OF COUNT DOWN.

```

\*\*\* \*\*\*\*\*

\* \* \* \* \* RETURN TO NORMAL INTERRUPT STATUS & HALT  
\* \* \* \* \* DISABLE INTERRUPT & TURN ON SPEAKER

```

040.154 076 000 MVI A,0
040.156 062 010 040 STA FLAG  DISABLE CLOCK INTERRUPT
040.161 315 136 002 CALL ALARM
040.164 166 HLT

```

\*\*\* \*\*\*\*\*

\* \* \* \* \* INTERRUPT ROUTINE  
\* \* \* \* \* CLOCK AND DISPLAY INTERRUPT

```

040.165 052 232 040 INTRP  LHLD TICK  GET COUNT (BETWEEN 0 & 500)
040.170 053 DCX H      TICK=TICK-1
040.171 042 232 040 SHLD TICK  STORE COUNT
040.174 175 MOV A,L    TEST FOR ZERO
040.175 264 ORA H      COMPARE WITH 'H'
040.176 300 RNE      EXIT IF 'NE' 0

```

\*\*\* \*\*\*\*\*

\* \* \* \* \* UPDATE DISPLAY FOR 'NEW' NUMBER.

```

040.177 076 015 MVI A,CR  DO CARRIAGE RETURN
040.201 315 302 003 CALL WCC
040.204 076 012 MVI A,LF  AND LINE FEED
040.206 315 302 003 CALL WCC
040.211 072 234 040 LDA NUMBER
040.214 075 DCR A      GET NUMBER
040.215 062 234 040 STA NUMBER-1
040.220 315 343 005 CALL TOB  NUMBER
040.223 041 364 001 SETICK LXI H,500  SAVE NUMBER
040.226 042 232 040 SHLD TICK  RESTORE COUNT
040.231 311 RET      WITH 500

```

```

040.232 DS 2      STORAGE AREA & END ASSEMBLY
040.234 DS 1
040.235 000 MTRBB  END

```

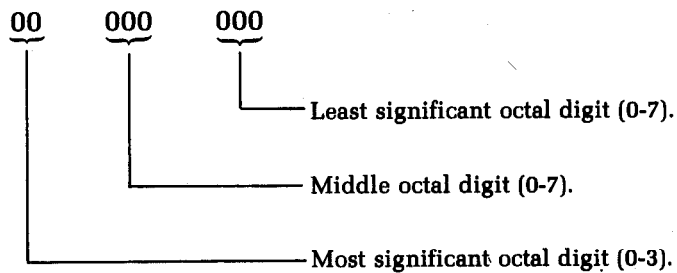
## APPENDIX C

### OCTAL DEFINITIONS

Binary numbers are converted to octal format for display. The following table shows binary to octal conversion.

<u>BINARY NUMBER</u>	<u>OCTAL DIGIT</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 representation of 16-bit numbers 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** or **split-octal**, and is used consistently for displays of 16-bit numbers.

Split-octal must not be confused with octal. For example:

$\begin{array}{ccccccc} \overline{11} & \overline{11} & \overline{11} & \overline{11} & \overline{11} & \overline{11} & \overline{11} \\   &   &   &   &   &   &   \\ 3 & 7 & 7 & 3 & 7 & 7 & \end{array}$	A 16-bit binary number
$\begin{array}{ccccccc} \overline{11} & \overline{11} & \overline{11} & \overline{11} & \overline{11} & \overline{11} & \overline{11} \\   &   &   &   &   &   &   \\ 3 & 7 & 7 & 3 & 7 & 7 & \end{array}$	Split-octal representation (377 377)

$\begin{array}{ccccccc} \overline{1111} & \overline{1111} & \overline{1111} & \overline{1111} & \overline{1111} & \overline{1111} \\   &   &   &   &   &   \\ 1 & 7 & 7 & 7 & 7 & 7 & \end{array}$	A 16-bit binary number
$\begin{array}{ccccccc} \overline{1111} & \overline{1111} & \overline{1111} & \overline{1111} & \overline{1111} & \overline{1111} \\   &   &   &   &   &   \\ 1 & 7 & 7 & 7 & 7 & 7 & \end{array}$	True Octal representation (177777)

The lower example shows true octal representation of a 16-bit binary number. True octal representation is never used in standard Heath software. Occasionally you will see split-octal numbers printed with a decimal point separating the upper and lower bytes. For example:

$$\underline{377.377}$$

Hi Byte      Lo Byte

Note that 001.000 follows 000.377.