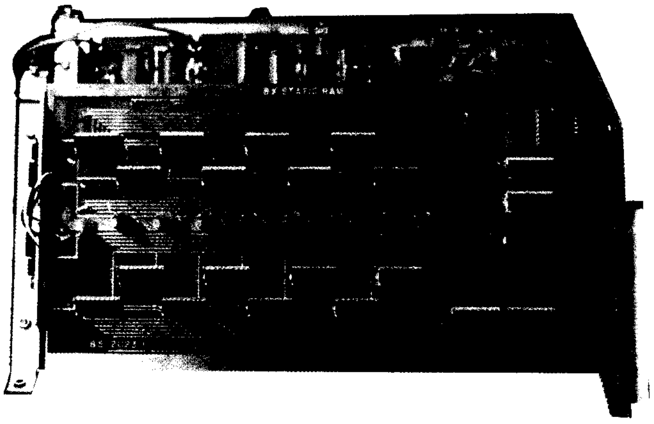


Photo 1: *The H8 programmable memory board.*

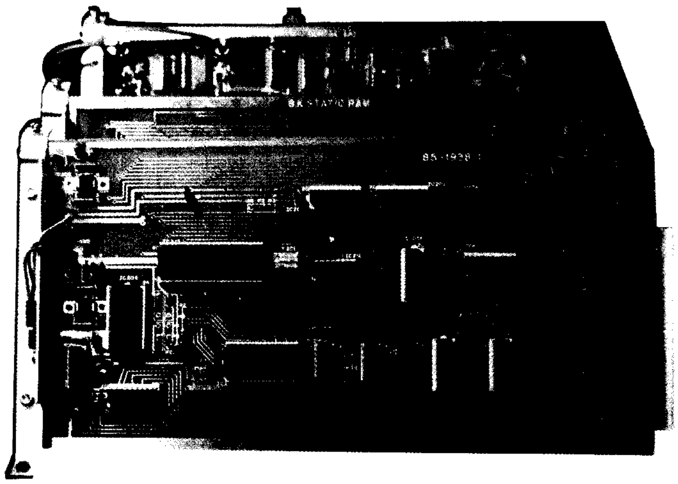


Photo 2: *The H8 processor board in position in the chassis.*

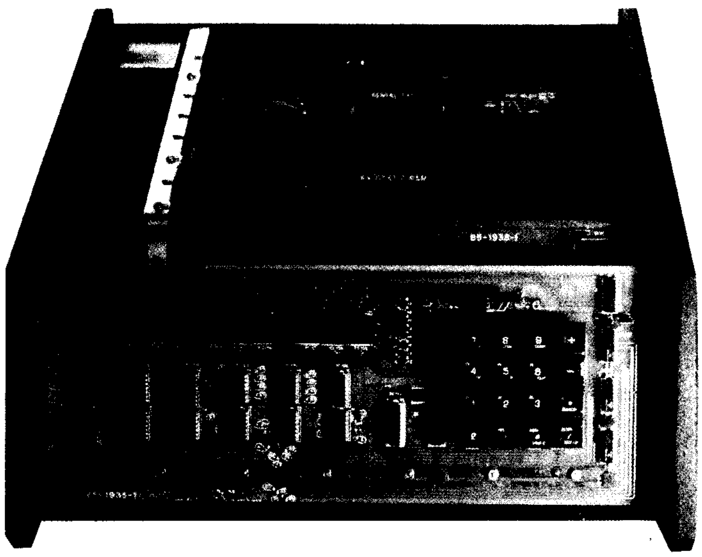


Photo 3: *Assembled H8 ready for testing.*

Building

Dr Paul R Poduska
21G Hampshire Dr
Nashua NH 03060

In response to growing public interest in microcomputing during the past several years, a number of microcomputer kits have been marketed.

The Heathkit H8 microcomputer represents a departure from the S-100 bus design mainstream. It has the full instruction set capabilities of other 8080A systems as well as an innovative, user oriented front panel control subsystem and a 10 position mother board; yet it does not use the type of large power supply found in some larger systems.

It was the kind of design I was hoping for. I was seriously considering the purchase of a computer for some time, and prior to the introduction of the H8, I almost did buy one. But I had a hunch that the Heath Company might resolve some of the design drawbacks that discouraged me from buying other 8 bit kits.

What follows are my experiences, thus far, of building, testing, and running this computer kit. By the time you've finished reading this article you should have a good feel for what the H8 is, what it can do, and how it compares with other kits on the market. In addition, I'll give you a few pointers and short programs that will enable you to take advantage of some of the H8's many features.

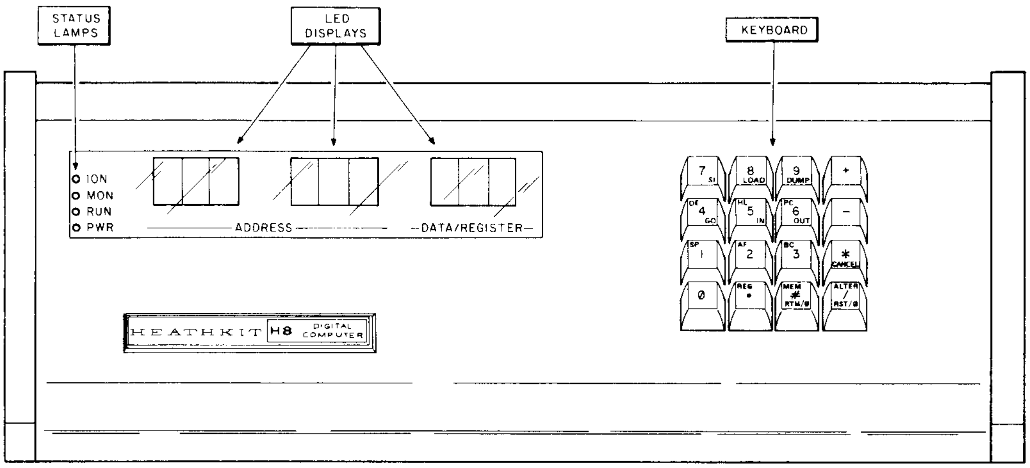


Figure 1: The Heath H8 front panel. Many of the 16 keys are multifunctional. The panel allows quick entry of data and programs into the computer. Drawn from a diagram courtesy of Heath Company.

the Heath H8 Computer

Unpacking and Building

My first few evenings with the H8 kit were leisurely spent getting acquainted with the materials and manuals. Opening the 22 lb carton, the first thing that caught my eyes was a loose slip of paper with a notice on it. It warned me to review all of the manuals supplied with the kit before putting anything together, and that if I found it to be too complicated for my knowledge and skill level, I could return it to the Heath Company for a full refund. I know of no other computer kit manufacturer who includes that kind of built-in consumer protection from the outset.

The next items were the sets of manuals and an attractive 3 ring binder to store them in. I had assembled Heathkits before and knew how thorough their documentation has always been; and I had seen some mediocre documentation provided by other companies in the computer kit industry. I found 503 pages worth of assembly and operating manuals which were far superior to any that I had seen on the market.

In all fairness, I don't feel that this point can be overemphasized. Building a microcomputer is not like building a stereo or piece of test equipment. The technology and associated hardware of a computer are unique in many ways. You may have assembled and tested electronic kits before, but unless you've also assembled digital equipment, you probably won't be an expert on this type of hardware. If you ever need to troubleshoot your computer kit, much of your previous knowledge about *conventional* analog electronics won't be of use. Therefore, *it is important that you know what you are assembling and how to do so correctly.* You don't have to be an electronics wizard to build and operate the H8. Your chances of building a computer kit that works the first time you turn on the power are nearly 100 percent with an H8, due in part to the support services that the manufacturer provides.

After the first few evenings of studying the manuals, I found a reasonably undisturbed space and, with tools in hand, began the first stage of assembly.

The Chassis

Chassis assembly consists of installing the power supply, frame, and several accessories, including a small cone speaker. The frame is made out of heavy sheet metal on the top, bottom and back panel, and structured foam on the sides. The bottom is covered with rows of small holes necessary for the H8's convection cooling system. The ample 10 A power supply is located

Continued on page 124

Our MacroFloppy™ goes twice the distance. For $695.

Introducing the Micropolis MacroFloppy™:1041 and :1042 disk drive subsystems. For the S-100/8080/Z-80 bus. Packing 100% more capacity into a 5¼-inch floppy disk than anyone else. 143K bytes, to be exact. For as little as $695.

The MacroFloppy:1041 comes with the Micropolis Mod I floppy packaged inside a protective enclosure (without power supply). And includes an S-100 controller. Interconnect cable. Micropolis BASIC User's Manual. A diskette containing Micropolis BASIC, and a compatible DOS with assembler and editor. The :1041 is even designed to be used either on your desk top, or to be integrated right into your S-100 chassis.

The MacroFloppy:1042 comes with everything the :1041 has, and more. Such as d.c. regulators, its own line voltage power supply, and, to top it off, a striking cover. Making it look right at home just about anywhere.

Both MacroFloppy systems are fully assembled, tested, burned-in, and tested again. For zero start-up pain, and long term reliability. They're also backed up by our famous Micropolis factory warranty.

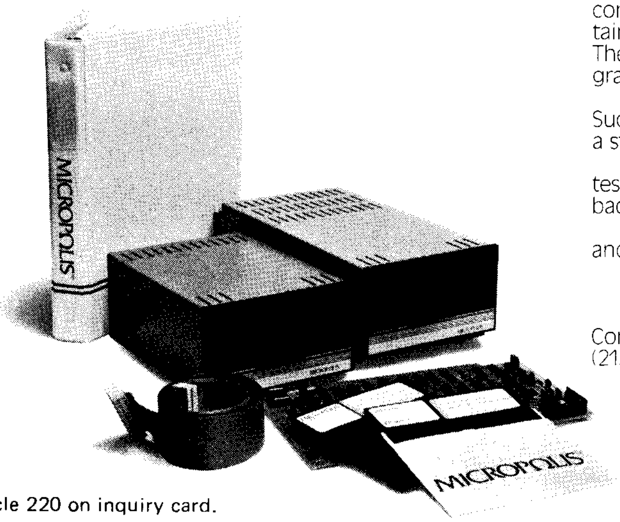
And both systems are priced just right. $695 for the MacroFloppy:1041 and $795 for the MacroFloppy:1042.

You really couldn't ask for anything more.

At Micropolis, we have more bytes in store for you.

For a descriptive brochure, in the U.S. call or write Micropolis Corporation, 7959 Deering Avenue, Canoga Park, California 91304. Phone (213) 703-1121.

Or better yet, see your local dealer.



MICROPOLIS™
More bytes in store for you.

Circle 220 on inquiry card.

Continued from page 13

along the entire back panel as well as the lower back corner of the mother board (5 VDC voltage regulators are on each circuit board). With two exceptions, everything went together easily, and exactly as presented in the assembly manual.

The first problem occurred when I attempted to install the self-retaining nuts in the side panels. For one thing, it can be very difficult to press these nuts into the very sturdy composition board unless they are gently tapped with a hammer. In addition, the part numbers for one of the nuts on the left side panel as illustrated in the assembly manual didn't correspond to the associated detail pictorial in the illustration booklet.

The only other problem I encountered during this stage of assembly was the installation of the side panels. Access to the side panel self-retaining nuts at the bottom rear corner of the chassis was obstructed by the previously installed screws and nuts that hold the rubber feet to the bottom of the chassis. My solution was to turn the chassis onto the side panel to be attached and insert and tighten the remaining screws to the appropriate panel.

The Mother Board

The mother board consists of two sections: a portion of the power supply, and ten rows of paired 25 pin plugs used later for connecting circuit boards and cables to the system bus. The installation of the plugs can be tedious. If you have not had much soldering experience, practice before you begin this board.

The most challenging part of the mother board assembly process is soldering some 500 pins which form the connectors for the individual boards of the system. The remainder of the board consists of a few capacitors, resistors, and diodes. There are, however, two instructions whose sequence should probably be reversed. Specifically, you are instructed to solder two electrolytic capacitors to the board and then secure each to the board with a self-locking cable tie. Reversing this sequence insures that no strain will be placed upon the soldered capacitor leads by the cable ties when they are tightened. The finished board is installed on the righthand side panel of the chassis after the twisted pair of 18 V leads from the power supply transformer are connected to the board. I had no trouble obtaining the proper resistance and power supply

MetaFloppy™ goes beyond.

The Micropolis MetaFloppy™ gives you more than four times the capacity of anyone else's 5¼-inch floppy. Because it uses 77 tracks instead of the usual 35.

The field-proven MetaFloppy, with thousands of units delivered, comes in a complete family of models. And, like our MacroFloppy™ family of disk drives, MetaFloppy is designed for the S-100/8080/Z-80 bus.

For maximum capacity, choose our new MetaFloppy:1054 system. Which actually provides you with more than a million bytes of reliable on-line storage. For less money than you'd believe possible.

The MetaFloppy:1054 comes complete with four drives in dual configuration. A controller. Power supply. Chassis. Enclosure. All cabling. A new BASIC software package. And a DOS with assembler and editor. There's even a built-in Autoload ROM to eliminate tiresome button pushing.

If that's more storage than you need right now, try our MetaFloppy:1053, with 630,000 bytes on-line. Or our MetaFloppy:1043, with 315,000 bytes on-line. Either way, you can expand to over a million bytes on-line in easy stages, when you need to. Or want to.

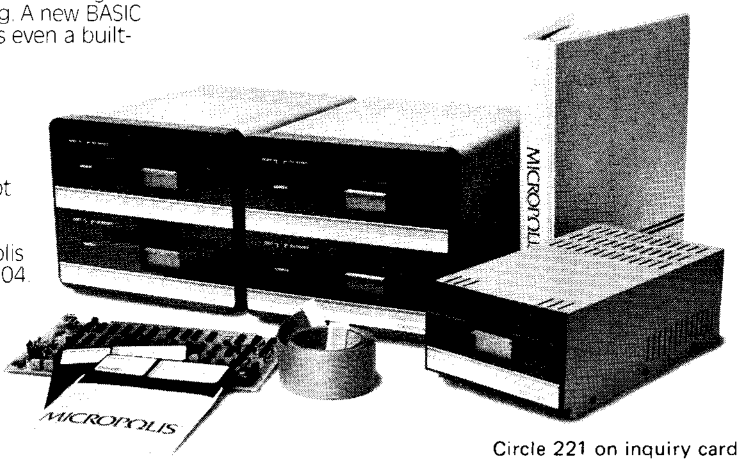
In other words, if your application keeps growing, we've got you covered. With MetaFloppy.

The system that goes beyond the floppy.

For a descriptive brochure, in the U.S. call or write Micropolis Corporation, 7959 Deering Avenue, Canoga Park, California 91304. Phone (213) 703-1121.

Or better yet, see your local dealer.

MICROPOLIS™
More bytes in store for you.



Circle 221 on inquiry card.

test readings outlined in the manual, and proceeded to the next step of assembly.

Front Panel Circuit Board

The H8 front panel control system has many features which make using the H8 a pleasure. One of these is the multifunction console keypad that provides users with direct, easy-to-use commands to operate the H8 without a terminal (see figure 1). Another feature is the 7 segment LED (light emitting diode) display system that displays a variety of system status information in an easy to read format not found on many larger 8 bit systems. We'll take a closer look at this unique system later, but first let's consider its assembly.

The entire assembly procedure is very straightforward. However, it is during the assembly of this board that a second rather tedious subassembly construction activity is encountered. In particular, it is necessary to prepare the cables that run from the board to the first row of plugs on the mother board and between the front panel and processor boards. The instructions and illustrations for this stage of construction are excellent and, if they are followed exactly, you should have no problems. The work performed on the cables is rather delicate, requiring the same amount of care and patience needed while assembling the mother board. One construction hint: you are instructed to mount four single element LEDs on the upper left side of the board. Make sure that each LED is mounted at exactly the same height as the others (0.25 inches from the board to the base of each LED). This can be easily accomplished by cutting a piece of 0.25 inch wire and using this as a spacer placed underneath each LED between the pairs of LED leads prior to soldering each to the board.

The Processor Board

Another feature of the H8 is the processor board, which comes preassembled. This is a real advantage because many of the problems that computer kit builders encounter are caused by mistakes made during assembly or installation of the processor board (see photo 2).

With assembly of the circuit boards and chassis complete, all that remains is to install the components. Of course, the H8 does not come with other boards that are necessary to have a functioning computer, such as extra memory and an

I/O (input/output) board. These are options the user purchases when buying an H8 (and most other computers, for that matter).

The Memory Board

If you want to use your H8 without external devices, the H8-1 memory board is the only other board you will need. The H8-1 is a 4 K byte programmable memory board which can be extended to a full 8 K by installing the optional H8-3 memory chip set. This board is easy to assemble, and not much more need be said about it – nothing, that is, except a short story about a mistake I made which also points to another nice feature of the H8.

Briefly, it happened like this: I was assembling the H8's circuit boards with extreme care, paying particular attention to the quality of the instructions and illustrations and making notes in my manuals I might want to use later. On the evening I started putting the memory board together, friends dropped over, and because the board looked so simple to put together, I decided to put part of it together while we relaxed and chatted. As a result of my divided attention, I installed a 14 pin IC socket in a 16 pin socket location.

One of my visitors was just beginning to get interested in computers (which was part of my reason for doing some of the assembly while company was around) and I showed her the partially completed board. She looked at it, asked a few questions, and then said, "What are those holes for by the end of that socket?"

I turned a bit red – I had blundered in the middle of a demonstration. Well, for the next 45 minutes we struggled to remove the socket, clean out the solder that was plugging the holes in the board and refit another socket. The H8 as well as its accessory kits come with sockets for all of the ICs including the 7 segment displays. If you ever need to replace an IC, you won't have to struggle with unsoldering it and cleaning off the board. And you won't have to pay extra for the socket sets when you buy the kits. The finished board is shown in photo 2.

The H8-5 SIO Board

If you want to connect your H8 to a console terminal or use the system software stored on cassette tape, you'll also want the H8-5 serial I/O and cassette interface board. It took quite a while to assemble because of its relatively high component density, but it presented no construction problems.

Testing and Alignment

After assembling the basic H8 system, you are instructed to perform three programmable test routines as well as an alignment of the cassette interface. The first test routine is a short program entered in machine code through the front panel keypad. The routine performs a general check of the H8 and at its end-of-run displays several messages on the front panel LEDs (ie: YOUR H8 . . . IS UP AND . . . RUNNING). If the test routine does not execute properly, the reader is directed to an extensive trouble-shooting flowchart that is ten pages in length and very clearly written. I had no trouble at this point and proceeded to the next test routine.

The second of the two routines is a memory test routine. It is also entered in machine code through the front panel keypad. The program performs a thorough test of every memory location on any 4 K or 8 K byte memory board by storing and retrieving consecutive octal values from 000 through 377 in every memory location. If an incorrect value is detected during a compare operation, the program halts, sounds an audio alert, and displays the expected (rather than observed) contents of the location where the test failed. The address where the test fails and the actual (or observed) contents of memory at that location can both be displayed on the LEDs by displaying the contents of the HL register pair and the accumulator, respectively. If no problems are encountered during execution of this routine, it will continue to repeat the test cycle until stopped by the user. Everything worked fine for me as I watched the memory content values go sailing by in the display.

The last task to be performed is the alignment of the cassette interface on the cassette I/O board. The procedure consists of setting the two variable resistors on the board to the correct positions as indicated by the readout on an on board single element LED test lamp. This LED also comes in handy later for troubleshooting the H8's circuitry. The adjustments were quite tricky to make, but after two tries and a number of test loads of software cassette tapes, it worked very well. Had it not worked, I could have referred to another trouble-shooting flowchart to locate the problems.

But everything did work. I proceeded to play around with the keypad commands, becoming familiar with what made my H8 "tick." Finally it was time to install the front panel cover and the louvered metal chassis cover. I connected the serial I/O cable to my H9 video terminal and was ready to do the final test routine for the serial I/O channel on the serial I/O board. It consists of setting up the USART (universal synchronous/asynchronous receive/transmit device) assigned to this channel and transmitting and receiving characters to and from the terminal. I was amazed at how easy it was to do I/O routines from the front panel. The final test was completed, so I left my system for awhile to return to the H8 operating manual and software manual set.

The Benton Harbor Bus

The Benton Harbor Bus represents a departure from the S-100 bus, which has become one of the "standards" of the microcomputer industry. The H8 system bus uses only 50 lines, compared with the 100 lines of the S-100. Since the S-100 was designed early in the history of the microcomputer industry, it incorporates bus lines which are no longer needed or which have been replaced by more recent system control hardware.

Manufacturing costs are kept down by not having to machine an edge connector tongue, not using gold plated edge connectors, and replacing the expensive 100 pin socketing system of the S-100 with much less expensive plug and socket sets which are assembled by the user.

Another feature found on the H8 is its convection cooling system. This system has been designed so that power supply voltage decreases slightly every time a circuit board is added to the mainframe. As more boards are added, the proportional amount of heat dissipation from each circuit board regulator also decreases. It is for this reason that you are instructed to "locate circuit boards in alternating positions for improving ventilation." When the time comes to add boards to the unoccupied alternating rows of plugs on the mother board, the effective heat dissipation will be low enough to position boards adjacent to one another. To improve upon this scheme, all circuit boards are installed on a slant to facilitate the convection cooling process. The end result is a quiet running machine which does not require the added cost of a fan to keep component temperatures down.

Costs are also kept down by the size of the H8's power supply. I saved a lot of money by not buying an 8 bit machine with a large power supply. The fact is that you pay dearly for every extra and often unused ampere that a power supply delivers. In this regard, the H8 delivers a full 10A, which is sufficient to operate all of the boards that the chassis can hold – up to seven in addition to the front panel and processor boards. It is also switch-selectable for operation on either 110 or 220 VAC, an advantage for European users. It also includes a switch for normal or low level line voltages, which may come in handy in case of a brownout – you'll need a separate generator for a blackout.

Split-Octal Notation

Before going on to discuss the firmware that coordinates many H8 operations, we should first describe the type of machine language code notation used by the H8. The H8 uses a number system called split octal, a modification of straight octal that is well suited to 8 bit computers like the H8. It is also well suited for the H8 display system, as you will see.

Split-octal notation is identical to octal notation except that the two most significant bits of each pair of data bytes are represented by one arabic numeral. In this scheme 377 is the highest value that can be represented by one 8 bit byte (ie: word) of data. Thus:

$$3\ 7\ 7 = \underbrace{1\ \ 1}_{3}\ \ \underbrace{1\ \ 1\ \ 1}_{7}\ \ \underbrace{1\ \ 1\ \ 1}_{7}.$$

And, the highest value that can be represented by two 8 bit bytes of data would be 377.377. The H8 defines 1 K bytes of memory as 003.377 bytes and 8 K bytes of memory as 037.377 bytes. The H8 is designed to reserve certain portions of memory for the system monitor and later system expansion as shown in the H8 memory map (see figure 2).

PAM-8, the Front Panel Monitor

The functions and features discussed above are tied together by another H8 feature – the front panel monitor – which resides in 1 K bytes of read only memory on the processor board. It also controls such activities as initializing the system during power-up, coordinating tape loads and dumps, communication with the Console Driver routine (part of every Heathkit software package), processing restart and clock interrupt vectors, and processing user defined interrupt requests.

H8 Software

Someone once said a computer system is no better than the software that comes with it. This is as true for large computers as it is for microcomputers. The H8 software packages are intended to function as an integrated system. They include TED-8, a line oriented text editor; HASL-8, a machine language assembler; BUG-8, a machine language debugger; and Benton Harbor BASIC, Heathkit's version of the popular interactive programming language created many years ago at Dartmouth College. Let's see what they're all about.

Common Elements

Each of these software packages contains a console driver routine that enables the H8 to communicate with a user's console terminal (Teletype, video display terminal, etc) and a cassette or paper tape transport. It occupies 355 decimal bytes of programmable memory in each software product from locations 040.100 through 041.144. Thus, all of the console driver routines are available to users for special purpose modifications as well as for use in user designed software which would benefit from using the same I/O routines, including I/O port assignment. It is responsible for processing console terminal interrupts, setting up and reading USART (universal synchronous-asynchronous receiver-transmitter) modes, commands and statuses, processing control characters (such as CTL-C, CTL-Q, and CTL-S), reading and writing single characters, and processing character strings in the 28 character type-ahead buffer. During interrupt processing, the console driver interacts with the PAM-8 system monitor. The entire listing of the console driver routine in assembly language is provided in an appendix to the software manuals. As in the case for the monitor, the console driver routine and all other software for the H8 were developed for Heath by Wintek. The Heath Company cannot (under contract with Wintek) provide any listings other than the partial listings presented in the software manuals, and "cannot provide consultation on user developed programs or modified versions of Heath software products." However, with a homebrew disassembler or memory dump routine and the partial listings provided, you might not have too much difficulty in generating your own unofficial versions of the source codes.

Another common element to all Heath software products is the software configuration procedure that must be performed before any software distribution tape is loaded into the H8. The procedure provides opportunities for the user to adapt each software package to his or her own needs. A list of program configuration parameters is presented to the user, who can then define items such as BKSP, which allows the user to define which key will be used to control the backspace function, HIGH MEMORY, which allows the user to specify that section of high memory that he or she does not want used by the system program, and LOWER CASE, which allows the user to select either upper and lower case or upper case only I/O to the terminal.

A third element common to all Heath software products is command completion. The feature allows the user to enter the first letter(s) of a program command, whereupon the software routine, in conjunction with the console driver, will print out the remainder of the command. For example, a text editor allows the PRINT command to be evoked by typing P, and the system responds with RINT, completing the command. The problem is that, if one is a fast

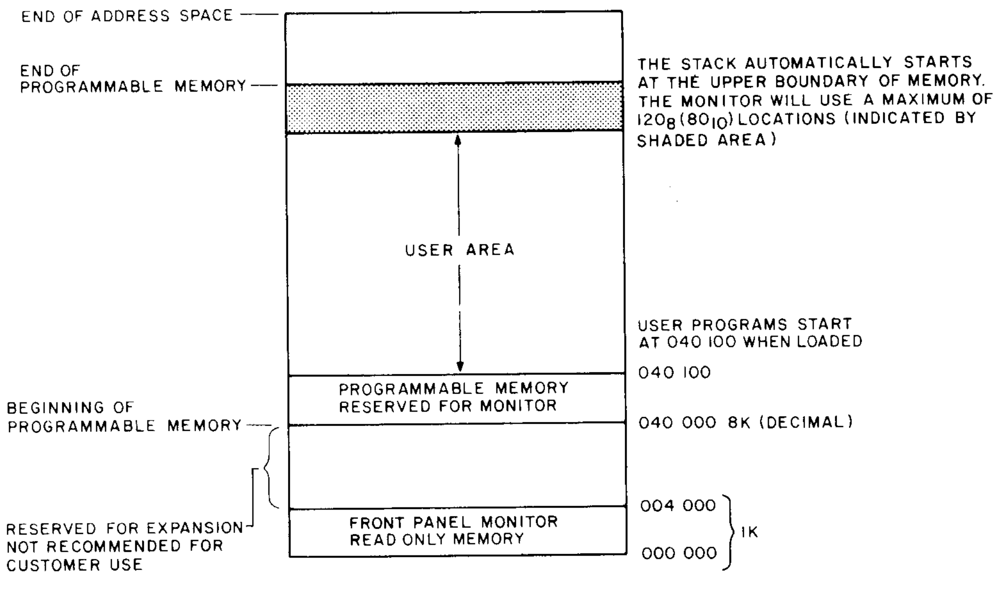

END OF ADDRESS SPACE
END OF PROGRAMMABLE MEMORY
THE STACK AUTOMATICALLY STARTS AT THE UPPER BOUNDARY OF MEMORY. THE MONITOR WILL USE A MAXIMUM OF 120_8 (80_{10}) LOCATIONS (INDICATED BY SHADED AREA)
USER AREA
USER PROGRAMS START AT 040 100 WHEN LOADED
040 100
PROGRAMMABLE MEMORY RESERVED FOR MONITOR
BEGINNING OF PROGRAMMABLE MEMORY
040 000 8K (DECIMAL)
004 000
1K
000 000
FRONT PANEL MONITOR READ ONLY MEMORY
RESERVED FOR EXPANSION NOT RECOMMENDED FOR CUSTOMER USE

Figure 2: The H8 memory map.

typist or has used computer terminals before, one tends to enter too many beginning characters and thereby cause an error. For example, I still haven't gotten used to typing just P to evoke the print command. So what happens? I type out the entire command, PRINT, and the result is PRINT RINT. The problem is even more bothersome because the command completion for some of the same commands is different for different software. Using the text editor (the above example), typing just P evokes the RINT completion; using BASIC, a PR evokes the INT completion.

TED-8, the Heath Text Editor

A text editor turns the user's terminal into a typewriter of sorts with the added capability to store what is typed into memory, and to edit the contents once stored. It is essential for writing assembly language programs and for preparing various kinds of texts, such as newsletters and reports, for personal use.

TED-8 is a line oriented text editor that occupies approximately 4.3 K of user memory. It has commands for loading and dumping both files and individual records, searching for character strings, clearing the text buffer, replacing and deleting lines, specifying range statements which control the number of lines upon which commands will operate, creating new text, and editing existing text in the buffer. It also includes a TAB command to facilitate the writing of assembly language programs, a wide variety of range commands, and a USE command that provides the user with a current number-of-lines count and memory usage information.

In my opinion, TED-8 is a reasonably good small system text editor with considerable flexibility. It is, however, somewhat inconvenient to use, especially when one needs to edit a particular line but does not know the exact line number and must therefore either count the lines (an inconvenience if there's a lot of text to cover) or write and execute an often lengthy expression to specify the character string in the line that is to be edited. Some of this could have been avoided if a line numbering system had been included, which could be stripped off at the end of editing.

Another problem is that the text editor and assembler are interdependent in some situations but must each be loaded separately into memory. This makes repeated editing and execution for program testing a real chore. I would have liked to see these programs (TED-8 and HASL-8) put together as one package, or have been provided with a dedicated linking loader as part of the PAM-8 monitor that would enable me to load both programs into memory. Aside from these two disappointments, I am pleased with the general capabilities of TED-8. For the small amount of memory it occupies, it is relatively powerful.

HASL-8, the Heath Assembly Language

HASL-8 is a 2 pass assembler which effectively means that you must load (playback) your source program (created by TED-8) twice before the assembly of the source program can be executed. (You can see how often you may have to load and unload, rewind and playback cassette tapes if you are editing and testing even a moderately sized assembly language program.) The output of HASL-8 is object code which is the machine language for the computer. In addition, a source listing of your program is generated, which includes not only the text of your original program but also the addresses which correspond to specific assembly language and machine language instructions (op codes). The Heath assembly language resides in slightly less than 8 K bytes of user memory and supports a complete set of operators, tokens, and pseudo opcodes (assembler directives), including in the latter a large set of pseudo ops for error detection and control of listings.

When a configured version of HASL-8 is started, the user must first answer a series of questions regarding the desired page size, interpage gap size, listing port, whether or not to produce a binary image of the source program in memory, and whether or not to save a binary image on tape. The program then requests input of the source program from tape, makes a second pass after the tape is rewound, assembles the object program, and dumps out a listing at the console terminal. The listing includes single character error codes in the far lefthand margin, and run summary data at the end of the listing (number of statements listed, remaining free bytes in user memory, and the number of errors detected). HASL-8 does not process macros or provide a cross-reference listing of user labels and associated addresses, both of which would be a nice improvement in a subsequent version. It is a fairly average assembler for its size, but should be adequate for most general purpose assembly language programming. Its documentation is very good, as is the documentation for all of the software products from Heath.

BUG-8, the Heath Console Debugger

BUG-8 is one of the best software packages that comes with the H8. Residing in any

3 K bytes of memory, it provides the user with powerful tools for writing, editing and debugging machine language programs from a console terminal in octal, decimal or ASCII format. It interacts with the PAM-8 monitor and uses many of the PAM-8 routines. With BUG-8 running as a full console monitor, you can do any of the following:

- examine the contents of any memory locations;
- change the contents of memory locations;
- examine the contents of any processor register;
- change the contents of any processor register;
- start program execution;
- perform single step execution of a program;
- set program breakpoints;
- clear program breakpoints;
- load and dump programs from OR to tape.

The command format used in BUG-8 is short, which makes using BUG-8 a real pleasure (the same kind of feature would be very desirable as a configuration option for TED-8).

Benton Harbor BASIC and Extended BASIC

When you purchase an H8 kit, you are supplied with Heath's 8 K byte, Benton Harbor BASIC. In addition, Heath has also marketed two versions of Extended BASIC (versions 10.01.02 and 10.02.01) which provide the user with functions for string manipulation and a number of other useful features. Version 10.02.01 of Extended BASIC also enables the user to load and dump variables, program text, and both text and variables. This latest version is a significant improvement over the earlier version of Extended BASIC in that it allows the user to pass variables from one program to the next.

There has been quite a bit of discussion about the relative execution speed of various versions and brands of BASIC that are on the market. However, the benchmark tests for Heath BASICs that have appeared in the small systems literature have been performed only under somewhat restricted circumstances. In particular, the H8 real time clock was not enabled when Extended BASIC benchmark tests were published, which by omission and implication exaggerates the speed of the 8 K byte version of BASIC. Now, speed isn't the only criterion in evaluating a higher level language's performance, but to provide you with a more balanced perspective on this issue, I give you the following benchmark comparisons between the two versions of Extended BASIC and the 8 K BASIC that comes with the kit.

A modified benchmark algorithm developed by J G Letwin (a software engineer for Heath) was used to conduct all of the tests. All timings were accomplished using the H8 real time clock via the POKE and PEEK commands. Table 1 presents the comparative data for execution time of various functions in milliseconds. The real time clock was enabled for all tests. The algorithm I used is as follows:

```
110 B=1.1:C=1.5 (numeric constants)
120 POKE 8219,0:POKE 8220,0 (set clock = 0)
130 FOR I=1 TO 2000 (2000 iterations)
140 (line for each test function)
150 NEXT I
160 Y=PEEK (8219)+(PEEK (8220)*256)
    (read run time)
170 X=(correction factor for "A=B")
180 T=((Y/500-X)/2000 (time in ms)
190 PRINT "T= ";T,"Y=";Y (output)
32767 END
```

The argument at line 180 computes the time for execution of one operation of the function supplied at line 140 by dividing total time in milliseconds (Y) by 500, subtracting the time for the identity statement A=B, and then dividing by the total number of iterations (2000). Line 140 contains the function being benchmarked across each version of BASIC (for example, 140 A=SIN(B), 140 A=SQR(B)). The POKE and PEEK commands are placed immediately before and after the test routine to be timed, respectively, thus holding any time required to execute these commands constant and to a minimum.

Table 1: Comparison of execution times for three Heath BASICs for various common functions, operators, and the FOR/NEXT statement. All times are in milliseconds, with the real time clock enabled.

FUNCTION/ OPERATOR/ STATEMENT	8 K BASIC (Version 05.01.00)	Extended BASIC (Version 10.01.02)	Extended BASIC (Version 10.02.01)
A=B	14.9	15.7	15.8
ABS	1.2	1.2	1.2
+	1.9	2.7	2.5
–	2.2	3.0	2.7
INT	2.3	2.0	2.0
RND	2.4	2.4	2.4
PEEK	3.5	3.3	3.2
*	4.3	3.3	3.1
/	5.2	5.9	5.7
FOR/NEXT	5.4	5.5	5.5
POKE	14.5	13.8	13.7
COS	18.3	15.0	14.6
SIN	19.3	15.7	15.3
EXP	22.1	17.3	17.0
LOG	27.2	21.8	22.0
ATN	29.2	24.9	24.3
SQR	47.8	15.7	15.3

What can we glean from all these timings? For one thing, the 8 K basic BASIC is generally slower than either of the Extended BASICs. The differences in execution times between the three versions is not really significant for the arithmetic functions. On the other hand, the transcendental functions run faster in Extended BASIC. The differences between the versions are most evident when the SQR function timings are compared. It appears that for arithmetic applications, such as those used in most financial accounting, the user can get by very well without using Extended versions. On the other hand, those who plan to do engineering or scientific applications software development should give serious consideration to using only the Extended versions. All of the versions will operate approximately 10 to 15 percent faster if the real time clock is disabled, which might be a further incentive for the scientific programmer to use only the latest version (10.02.01) of Extended BASIC.

Aside from timing benchmarks, there are many other criteria to consider when reviewing a higher level language like BASIC. Since I cannot possibly cover all of them in this article, I'll select the ones which are probably of most interest to many readers. Let's start with the command and function types of the three versions.

8 K Benton Harbor BASIC contains most of the commands and functions found in many commercial brands of BASIC for small systems. It has a set of editing facilities, immediate and program command modes, and a full set of relational operators. It also has additional commands, including PEEK, POKE, SEG (for controlling the H8 LED displays), PAD (for utilizing the H8 control keypad while running BASIC), and several commands for loading and dumping programs. Most 8 K BASIC functions can use variables in addition to constants within their arguments, which makes programming more flexible. This version of BASIC does not provide functions for manipulating string variables.

In contrast to 8 K BASIC, both versions of Extended BASIC have a variety of functions for handling string variables. In addition, such commands as BUILD, DELETE, LINE INPUT and CONTROL make program development easier. Each version makes use of the FREE command, which prints out memory allocation data for the user. The latest version of Extended BASIC also includes commands for differentiated loading and dumping of program text and variable as well as LOCK/UNLOCK commands for file protection. Both versions display the type of data being loaded or dumped on the H8 front panel LED displays. As the data in table 1 indicates, the latest version is quite fast. Additional refinements, such as incorporating PRINT USING, OCTS and DECS (for octal to decimal, and decimal to octal conversion), and file handling commands that could be executed in program mode (such as PUT and GET), would have been useful. In addition, a special INPUT command that suppresses the carriage return/line feed would also have been useful formatting the terminal display during INPUT, as would a more flexible RUN command to specify a file/program name for storing and executing one of several programs stored in user memory. This can be accomplished now using a combination of the GOTO and CONTINUE commands. However, this procedure prevents the use of identical variable names in different programs unless such variables are reinitialized before each execution, or the variable values of a previously executed program are to be used as *global* variables in subsequent programs in memory.

A Few Pointers

The following section includes some suggestions about using the real time clock for program run timing, displaying output using user defined patterns or character messages on the LEDs, and using the audio feedback system for any number of interesting projects. By combining and augmenting some of the following techniques, you can use your H8 as a daytime clock, alarm clock, device controller timer, and even a (rather crude) music synthesizer.

Among the applications software packages available from Heath for the H8 are a Space War program and a set of games programs. The Space War game, which requires 24 K bytes of memory is a sophisticated version of Star Trek. It has excellent displays, a full range of commands, and some surprises for those of you who dream of cruising the galaxy. Finally, the games set, which comes on one cassette tape, contains a variety of interesting games, including Craps, Nim, Hexapawn, Tic Tac Toe, Orbit, Hamrabi, and Derby, all stored in machine language and executable without first loading the BASIC interpreter into memory.

In addition to the above applications software, the Heath Company is also planning to market some business software in the not too distant future. This source of applications programs will be supplemented by the software that will become available from the Heath user's group (HUG), which is presently reviewing and cataloging hundreds of programs that are being donated to HUG by users. Software from HUG will be available to HUG members at a nominal cost and will be announced in the HUG magazine *REMark* as contributed programs become available.

The H8 Real Time Clock

Every 2 ms the H8 generates a level 1 interrupt which is then used to service the front panel and update the real time clock. The clock or "tick counter" is located in two bytes of user memory at locations 040.034 and 040.033 and is called TICCNT. Both the high order byte (040.034) and low order byte (040.033) of the counter can be displayed on the H8 LEDs using the following short assembly language program which can be executed in memory with the assembler (HASL-8) resident:

```
074.000                       ORG  074000A
074.000 041  033  040         LIX  H,040033A
                              LOW ORDER BYTE
074.003 116  DISPLAY          MOV  C,M
                              LOAD LOW BYTE
074.004 054                   INR   L
                              NEXT BYTE
074.005 106                   MOV  B,M
                              LOAD HIGH BYTE
074.006 055                   DCR   L
                              RETURN POINTER
074.007 303  003  074         JMP DISPLAY
                              NEXT BYTES
074.012                       END  074000A
```

Be sure to set the front panel to display the BC register pair before running the program.

The BASIC timings described earlier accessed the TICCNT memory locations by using the PEEK function and POKE command. Using these commands, it is possible to create a program run timer for programs that run up to approximately two minutes. Unfortunately, after this time TICCNT will start all over again instead of continuing to increment third and fourth memory locations, which would have provided a very convenient built-in extended range timer. You will need to allocate additional bytes in memory for a higher order counter in order to extend the clock range.

H8 Display Control

As discussed earlier, the H8's front panel is an integral part of the entire system and can be used for many operations even while using a console terminal. One of the advantages of this design is that the LED display can be accessed and controlled by the user either through the front panel or through a program being run from a terminal. Much of this is made possible by accessing the monitor control cells and flags stored in memory.

As an example, the assembly language program in listing 1 provides you with complete control of the display updating. First the display update control bit in the user definable memory flag .MFLAG is set to 002 to disable display update. The displays are then loaded with the desired bit pattern, in this case the pattern "XrunningX" (where X = blank). The user accessible memory cells for each LED bit pattern are in memory locations 040.013 through 040.023. The temporary bytes for storing the pattern for this display are in locations 071.000 through 071.010. Each byte in the temporary area is, in progression, loaded into the corresponding byte in the monitor LED memory bytes. To modify the display, change the bit patterns in the temporary area. Add additional temporary bit pattern tables and a table reference and transfer routine to display more than one front panel message.

The H8 Horn

The audio feedback system that is part of the H8 has other practical uses. The audio tone can be turned on from a running program that stores the octal value 160 in the front panel hardware control cell CTLFLG located at 040.011. If an attempt is made to do this from the front panel without going through a program, the front panel will be disabled before the operation is completed. The tone can be turned off by restoring CTLFLG to its normal octal value of 360.

Here is an interesting BASIC routine to generate an audio beep sequence with timing that follows a sine curve. Change the value of X to vary the timing curve. (Note: line number 3 is written in multiple statement form to structure the program to run as fast as possible.)

```
1   X=1
2   A=8201: B=120: C=208: D=20
3   J=J+X: POKE A,B: POKE A,C:
    FOR I=X TO SIN(J)*D+D: NEXTI: GOTO 3
4   END
```

Split-Octal to Decimal Conversion

It is often useful to be able to quickly convert split octal bytes to their decimal equivalents. The following routine does this. It can be located as a temporary program in BASIC while other programs are being developed or executed. Be sure to use line numbers that do not conflict with those in the program you are using. (Note: input N if done.)

```
010   REM OCTAL TO DECIMAL CONVERSION
020   PRINT: INPUT "A=?";N
```

```
071.000                          ORG   071000A
071.000  377                     DB    377A       LED PATTERN 1
071.001  235                     DB    235A       LED PATTERN 2
071.002  203                     DB    203A       LED PATTERN 3
071.003  221                     DB    221A       LED PATTERN 4
071.004  221                     DB    221A       LED PATTERN 5
071.005  363                     DB    363A       LED PATTERN 6
071.006  221                     DB    221A       LED PATTERN 7
071.007  240                     DB    240A       LED PATTERN 8
071.010  377                     DB    377A       LED PATTERN 9
                          *
                          *
071.011  041 010 040             LXI   H,040010A  POINTER TO .MFLAG
071.014  076 002                 MVI   A,002A     (A)=OFF CONSTANT
071.016  167                     MOV   M,A        DISABLE LED UPDATE
071.017  041 000 071 DISPLAY     LXI   H,071000A  PATTERN TBL POINTER
071.022  021 013 040             LXI   D,040013A  LED TABLE POINTER
071.025  176         LOADIT      MOV   A,M        (A)=BIT PATTERN
071.026  353                     XCHG             (H)=LEDPT, (D)=DISPT
071.027  167                     MOV   M,A        DISPLAY IT
071.030  353                     XCHG             (H)=DISPT, (D)=LEDPT
071.031  043                     INX   H          NEXT PATTERN BYTE
071.032  023                     INX   D          NEXT LED
071.033  067                     STC              SET CARRY=1
071.034  077                     CMC              SET CARRY=0
071.035  076 024                 MVI   A,024A     (A)=COMPARE CONSTANT
071.037  273                     CMP   E          (A)<(E)
071.040  322 025 071             JNC   LOADIT     NEXT PATTERN/LED
071.043  303 011 071             JMP   DISPLAY    REINITIALIZE
071.046                          END   071011A
```

Listing 1: An 8080 assembly language program to allow the user to control the LED (light emitting diode) readouts on the front panel of the H8 computer.

```
030   IF N > 377 GOTO 100
040   IF N=0 GOTO 110
050   A=INT(N/100): IF A >3 GOTO 90
060   B=INT ((N/10)-(A*10)). IF B > 7 GOTO 100
070   C=N-((A*100)+(B*10)): IF C > 7 GOTO 100
080   Q=C+(8*((8*A)+B))
090   PRINT: PRINT "A="N;"D="Q: GOTO 20
100   PRINT: PRINT "ERROR! USE ONE
      SPLIT-OCTAL INPUT BYTE!"
110   GOTO 20
120   END
```

Heath Support Services

Regarding the servicing of digital systems like the H8, Heath's present approach is to do most repairs at regional centers or at the factory in Benton Harbor MI until such time as each local store can either provide additional in service training for existing technicians or add on additional technical staff. Some stores that are not regional centers, such as the Heathkit Electronic Center in Peabody MA, have service technicians who are capable of performing most of the necessary digital equipment repair jobs. All Heath products are warranted for a 90 day period from date of delivery. If replacement parts are needed, some parts can be obtained through local stores; the remainder must be ordered from the factory. I have used the replacement parts service and found it to be most adequate.

Another area of support services is the Heath Users Group (HUG). Owners of Heath computers can pay a yearly membership fee to join HUG which is administered in St Joseph MI. HUG supplies users with a variety of services including contributed software, membership information, and newsletters. HUG is an important component of the overall services provided by Heath and is growing rapidly. HUG members who are interested in forming local or regional users groups should feel free to contact HUG manager Robert Furtaw for additional information and assistance.

A third area of support services is new system products. The design of the H8 lends itself to upgrading and expansion in many ways, which Heath is already taking advantage of. I have also heard they're considering a number of designs for interfacing the H8 to their excellent line of ham radio gear, which should be of considerable interest to ham operators who are also interested in RTTY (radio teletypewriter). Likewise, I wouldn't be surprised if Heath introduces sophisticated video game boards to interface to their television sets, and household controllers to interface with some of their home oriented equipment.

Final Thoughts

Aside from these items, the possibilities are many. Digital to analog/analog to digital converters, video drivers, speech recognition systems, EROM (erasable read only memory) boards, upgraded firmware monitors, firmware versions of existing and future software, printers, modems, intruder entry detection devices, graphics boards, and perhaps even bubble or CCD (charge coupled device) memory boards are just a few of the many possible products that Heath might market.■