

\$2.50

IN THIS ISSUE THE FIRST IN A SERIES
MODEM TUTORIAL 83
by **Walt Jung**
see page 11

REMark[®]

Issue 40 • May 1983



Official magazine for users of **HEATH**  computer equipment.



THE H-100 SERIES: THE WORLD'S FIRST 16-BIT COMPUTER KITS.

with authorized sales and service
only through the Heath Company and
Heathkit® Electronic Centers.*

The world-famous H-100 Series Heath/Zenith computers are the first to give you advanced 16-bit computing at a kit price. Many who have never considered kitbuilding are now interested because most circuit boards are prewired, making the H-100 Series our simplest computer kits. Dual microprocessors deliver 16-bit speed and 8-bit compatibility. The industry standard S-100 card slots allow a host of peripherals and memory expansion to 768K RAM. And our stores and catalogs have the software to help you take full advantage of the faster 16-bit operating speed.

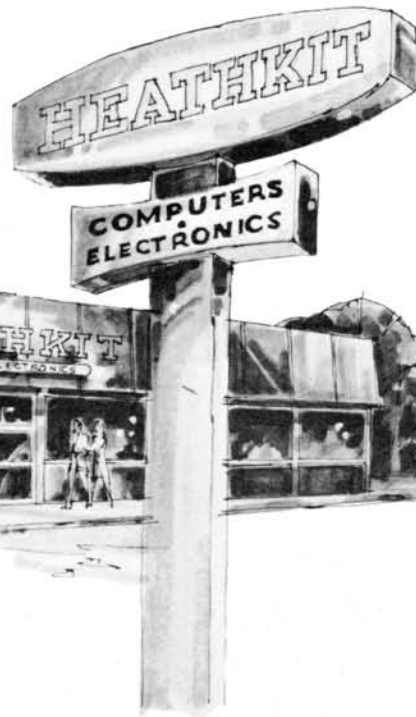
When you're ready to move up to H-100 Series,

remember: these computers are available *only* through the companies that give you solid service and a trustworthy warranty...Heath and Heathkit® Electronic Centers.* The companies that stand by their promise of support to kit-builders. The companies with the pledge: "We won't let you fail."

Many companies would like to sell our product but no other dealer or vendor can resell the H-100 Series without voiding the original factory warranty. When buying the world's first 16-bit computer kit, buy only from the world leader in electronic kits.

**Authorized sales and service available
only through the Heath Company
and your...**

Heathkit®
ELECTRONIC CENTER*



*Heathkit Electronic Centers
are units of Veritechnology
Electronics Corporation.

VEC-770

HUG Manager Bob Ellerton
 Software Engineer Pat Swayne
 HUG Bulletin Board
 and Software Developer Terry Jensen
 Software Coordinator Nancy Strunk
 HUG Secretary Margaret Bacon
 REMark Editor Walt Gillespie
 Assistant Editor Donna Melland

Printers Imperial Printing
 St. Joseph, MI

REMark is a HUG membership magazine published 12 times yearly. A subscription cannot be purchased separately without membership. The following rates apply.

	U.S. Domestic	Canada & Mexico	International
Initial	\$18	\$20*	\$28*
Renewal	\$15	\$17*	\$22*

*U.S. Funds.

Membership in England, France, Germany, Belgium, Holland, Sweden and Switzerland is acquired through the local distributor at the prevailing rate.

Limited back issues are available at \$2.50 plus 10% handling and shipping. Check HUG Product List for availability of bound volumes of past issues. Requests for magazines mailed to foreign countries should specify mailing method and appropriate added cost.

Send Payment to: Heath Users' Group
 Hilltop Road
 St. Joseph, MI 49085

Although it is a policy to check material placed in REMark for accuracy, HUG offers no warranty, either expressed or implied, and is not responsible for any losses due to the use of any material in this magazine.

Articles submitted by users and published in REMark, which describe hardware modifications, are not supported by Heathkit Electronic Centers or Heath Technical Consultation.

HUG is provided as a service to its members for the purpose of fostering the exchange of ideas to enhance their usage of Heath equipment. As such, little or no evaluation of the programs or products advertised in REMark, the Software Catalog or other HUG publications is performed by Heath Company, in general and HUG in particular. The prospective user is hereby put on notice that the programs may contain faults the consequence of which Heath Company in general and HUG in particular cannot be held responsible. The prospective user is, by virtue of obtaining and using these programs, assuming full risk for all consequences.

REMark is a registered trademark of the Heath Users' Group, St. Joseph, Michigan

Copyright © 1983, Heath Users' Group



on the stack

Second National HUG Conference <i>Bob Ellerton</i>	7
Conference Registration	7
Buggin' HUG	8
Questions & Answers	9
Modem Tutorial 83 <i>Walt Jung</i>	11
The WCCF in HUGger Perspective <i>Terry Jensen</i>	15
Teaching HERO A Thing Or Two <i>Pat Swayne</i>	17
Introduction to Z-BASIC Part VI <i>Gerry Kabelman</i>	21
H/Z-37 Controller on Standard 8" Drives <i>E. B. Blayer</i>	23
New HUG Products	26
HUG Price List	27
Computer Aided Instruction Part 2 <i>Walt Gillespie</i>	29
Datetime for ZDOS <i>Frank T. Clark</i>	31
A Review of RT-11, Version 5 <i>Ed Judge</i>	34
Getting Started With Assembly Language (#2) <i>Pat Swayne</i>	36
CheapCalc Revisited <i>Clifford C. Lundberg</i>	39
Putting The H-8 LED's To Work <i>Glenn F. Roberts</i>	41
"ESCAPE" with FORTRAN and PASCAL <i>John F. Sams</i>	44
Heath Related Products <i>Tom Huber</i>	50
Pilot Corrections	27
Raiders Bug	27

ON THE COVER: A montage of modems (photo by Jon Falkner) to help kick-off the first in a series of articles by Walt Jung, "Modem Tutorial 83", on page 11.

The Heath/Zenith 88-89 CPU with a Future

SUPER 89

The Super 89 replaces the central processor board in the Heath/Zenith 88-89 series computers to bring your 88-89 to current state-of-the-art technology. The Super 89 gives you features that are useful today and allow expansion of your capabilities. The Super 89 is fully compatible with all Heath/Zenith products and also supports many peripheral devices from other manufacturers. New software and hardware are enhanced with the Super 89 by using all the features of the Z80 technology.

The Highlights of the Super 89:

- Twice the operating speed (4MHz +)
- Memory Capacity to 256 K in Software Bank Selectable 64K blocks
- CP/M and HDOS Compatible without modification
- Twice the number of expansion slots (Six)
- Real time clock on-board
- Two serial I/O Ports
- Designed for multi-user capability
- Parity checking for RAM assures integrity of memory transfer operations
- Arithmetic processor provision facilitates mathematic operations

These features, along with an enhanced monitor to access to all the Z80 CPU, give you power from your 88-89 that only large computers can claim.

High Speed Processing

The Super 89 runs twice as fast as the standard H/Z CPU board. Time savings on running programs are significant.

Expanded Memory Capacity

This feature allows you to use the advantages of the more sophisticated programming languages; enables you to use enhanced memory software such as print spoolers and electronic disks to increase speed; allows the use of "scratch pad" memory to increase efficiency; the bank select features give you high speed data handling and manipulation; and provide for multi user capabilities.

Super 89 Electronic Disk

This optional software package for the Heath/Zenith CP/M 2.2.02 and 2.2.03 allows the Super 89 user to access auxiliary RAM as a very fast mass storage device. Provides up to 180K bytes of storage area that is accessible without the slowness of disk drives. The Electronic Disk also includes display capability for the Real Time Clock.

Peripheral Expansion

This important feature lets you use your Super 89 in more ways with peripherals from DG, Heath and many other manufacturers. Some of these important enhancements are: Additional floppy disk controllers; Modem or Printer serial interfaces; Color video controllers; IEEE 488 BUS for test equipment and measuring interface; Analog/Digital interface; Parallel interface for high speed printers; Hard disk system controllers; Bread-board development cards; Computer game controllers; and Production process controllers.

Enhanced Super 89 Monitor

Gives you all the features of Heath's MTR-89 monitor plus the ability to display all the Z80 register contents; Single-step through a program and set up break-points; Supports H/Z and other manufacturers of disk systems; Improved system diagnostic routines; and Supports the Super 89 Real Time Clock.

Real Time Clock

The Real Time Clock allows you to program activities and control functions according to time; allows the use of interactive time functions with an electronic disk; and is very useful in accounting functions.

Parity Checking

This feature ensures the integrity of memory transfer operations. The Super 89 alerts you if a parity error occurs.

Full CP/M and HDOS Compatibility

The Super 89 has full compatibility with either the HDOS or CP/M disk operating systems that does not require hardware modifications. This feature gives you the best of both worlds in the amount of existing software you may use.

Arithmetic Processor Provision

The Super 89 has on-board provisions for the optional AM9511A. This is a separate processor that features basic arithmetic as well as exponential, logarithmic, trigonometric and binary functions. Calculations are high speed and can be accomplished as a "hardware subroutine". This device is a must for anyone using any amount of mathematical computation whether complex functions or arithmetic calculations.

Ease of Installation

The Super 89 is simple to install and takes only minutes. No soldering required. Simply remove the old CPU board, configure and install the Super 89 to multiply the capabilities of your H/Z 88-89.

D·G ELECTRONIC DEVELOPMENTS CO.

Ordering Information: Products listed available from DG Electronic Developments Co., 700 South Armstrong, Denison, Tx. 75020. Check, Money Order, VISA or MasterCard accepted. Phone orders (charge only) call (214) 465-7805. Freight prepaid. Allow 3 weeks for personal checks to clear. Texas residents add 5%. Foreign orders add 30%. Prices subject to change without notice.



Fine Tuning the Future.....

Have you sat down lately and tried to comprehend the strides that have been made in the electronics industry recently? The advancements of the past 5 years, let alone the last decade have been mind boggling. Who thought that Sears and K-Mart would be selling computer systems twenty years ago?

I happened to be browsing through a surplus electronics warehouse the other day and the owner asked if I knew of anyone that might be interested in a used IBM-360 system! There it was, all twenty tons, sitting under a tarp. It was a little dusty but had been operating when taken out of service. "What price?", I asked, expecting an astronomical figure. The answer, "twenty-five hundred." Boy, has this industry changed!

With the introduction of the 16 bit computers such as the H/Z-100 a tremendous amount of computing power has been put into the consumer's hands. The possibilities of 32 bit units gaining wide spread acceptance look slim, at least until full exploitation of the 8 and 16 bit computers has been made.

The next few years should be the "years of the peripherals and software" in the computer industry. The idea of an LCD panel about a 1/4" thick replacing the CRT tube is thought possible. Forty megabyte Winchester type 5" disk drives are being experimented with. Low cost, \$600 color dot matrix printers are entering the marketplace and 64K memory chips are getting into the price range of the average computer hobbyist.

It's almost impossible to predict just what strides will be made in software offerings. Everyday it seems like something new comes on to the market. It would take at least a 150 page issue of REMark to just list the offerings for the Heath/Zenith computers let alone try to review them.

As to the future, that will be decided by you the computer owner, or potential owner! No one will plan on developing or marketing that new fast memory board or that "Wizz Bang" software if that isn't what you are buying.

You can help "fine tune" this future. Let the manufacturers know what you need and are willing to buy, not just what you would like to see.

Walt Gillespie
REMark Editor

Introducing -

THE NEW H8* COMPUTER

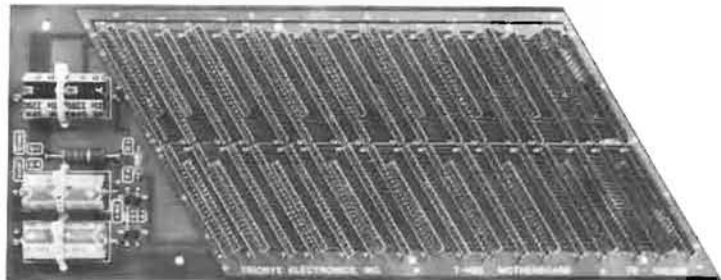
* H8 is a Registered Trademark of the Heath Company



PC Board Mounts Additional Pair of Connectors to Plug Into 90 Pin Bus



Optional 40 Pin Connector Assembly Converts H8 Boards to 90 Pin Bus



Fully Assembled Motherboard - \$250.00

Featuring the TRIONYX T-H90 MOTHERBOARD A Professional-Quality Bus for the H8 Computer

- 3-Layer Board Has Center Ground Plane

- Designed For 4 MHZ Bus Operation

- 7 Auxillary Positions For Port Addressable Bus Interface Cards

- Completely Compatable With Original H8 Computer

- 90 Pin Bus Has:

- Additional Ground Connections For Reliable Operation
- 8 Additional Data Bits For 16 Bit CPU Board
- Additional Address Lines For Expanded Memory Capacity
- Bank Select Lines For Memory Management
- Additional Lines For Special Control Signals and Future Defined Functions

PC Board \$ 75.00

25 Pin Gold Connectors - Set of 20 \$45.00
- Build Motherboard to Original H8 Standard

20 Pin Gold Connectors - Set of 18 \$34.00
- Add Additional 40 Pins for 90 Pin Bus

25 Pin Gold Connectors - Set of 14 \$35.00
- Add 7 Auxillary Card Positions

Power Supply Parts for Motherboard \$16.00
- Includes Extra +18 Volt Filter Capacitor

Bus Termination Card - Complete Kit \$29.50

PC Board Connector Expansion Kit \$15.00
- Adapts Original H8 PC Boards to 90 Pin Bus:
Contains PC Board, Two 20 Pin Connectors,
Mounting Blocks and Hardware - One Kit
Required Per PC Board

The new T-H90 Motherboard has been designed to provide completely reliable operation of the H8 computer through the use of gold-plated connectors and a well grounded bus. The bus has been expanded from 50 to 90 lines and 7 additional card slots have been added. Full implementation of the Motherboard functions will transform the H8 into a commercial grade computer.

Check • Money Order • VISA • MASTERCARD • C.O.D.

Phone Orders Welcome (714) 830-2092 - Send For Free Brochure

TRIONYX ELECTRONICS, INC.

P.O. BOX 5131, SANTA ANA, CA 92704

The Framework for the Second National HUG Conference



Bob Ellerton
HUG Manager

With this article and several updates to follow, HUG will begin the process of keeping you informed as to the progress being made toward the Second National HUG Conference. The information provided here is a rough description of the events planned for the Conference which will again be held at the O'Hare Hyatt Regency in Chicago, Illinois over the weekend of August 19, 20, and 21, 1983.

Many of you who attended the First National HUG Conference will know how exciting it is to meet with your fellow users at an event such as this. Last year, we were privileged to hear from individuals within the Heath/Zenith organization that have made our computer hobby grow and expand. This year we will be concentrating on the user community almost exclusively. This means that we will be able to hear from fellow users on subjects ranging from "Interfacing to the S-100 Buss in the Z100 Computer" to "Modem Communications".

To accomplish the task of a wide variety of subjects that may be of interest to a varying number of users, HUG has reserved four meeting rooms along with a fifth room to be used as a lounge for general discussion or a little relaxation. The speakers selected for discussions in these rooms will be available

Vectored to 8



NATIONAL HUG CONFERENCE II

Official Conference Registration Form
O'Hare Hyatt Regency Hotel, Chicago, Illinois
August 19, 20 and 21

Name(s) _____

Address _____

Company _____

City _____ State _____ Zip _____

Enclosed is \$20.00 per individual to attend The Second National HUG Conference to be held the weekend of August 19, 20 and 21 1983. Please send ticket(s) and information regarding hotel reservations.

AMOUNT ENCLOSED _____ NUMBER ATTENDING _____

For our information:

Which Heath/Zenith computer do you now operate? _____

Are you a Non-User-Attendee? YES NO

Are you a Heath/Zenith related vendor? YES NO

For your information:

Space limitations for the dinner to be held Saturday August 20, 1983, will restrict the number of attendees for that dinner to 1000. Therefore, it is important that you register as soon as possible. Visitor tickets for those of you simply attending and not planning to stay for the dinner and prize drawings will be available at the registration booth for \$10.00. Send your registration form or a suitable copy to:

Heath Users' Group
Attention: National HUG Conference Registration
Hilltop Road
Saint Joseph, Michigan 49085

Special Note to Vendors:

Vendor Information Packages will be made available to Heath/Zenith Related Vendors who are planning to exhibit their products while at the conference. Three times more space is available this year for the purpose of showing those products of interest to owners of Heath/Zenith computer products.

for talks at least three times during the weekend thus allowing you to attend most of the meetings should you desire. It was realized last year that many of the attendees were interested in a narrow spectrum of topics. Therefore, on suggestions from users, HUG selected the a method of rotating the meetings so that each of you can choose those subjects you wish to hear most about. As the speakers are selected in the next few months, we will be updating the program with examples of the meeting schedules for the Second National HUG Conference. We will keep you informed via REMark.

Last year, the Vendor Exhibit Area contained about 25 manufacturers of products or services related to Heath/Zenith computer equipment. The area was crowded most of the time and in some cases, it was difficult to get a good look at the products you were interested in. This year, HUG has reserved the entire bottom floor of the Hyatt to contain the Vendor Exhibit Area. Roughly, this means that we will have more than three times the space for manufacturers and users. Couple the increased area with the rotating meeting schedule, and we feel that you will have a good opportunity to see all of the goodies in the Vendor Exhibit Area. We currently have forty vendors of Heath/Zenith related hardware, software and support information signed up for the August event. The Vendor Exhibit Area will be open from 3:00PM on Friday afternoon allowing an extra half-day to examine products of interest to you (see details in the rough schedule provided with this article).

How does the schedule look to date?

Let's take a look at the tentative plan of events for the Second National HUG conference:

Friday, August 19, 1983

1:00PM

Registration Booth open for sign-in until 9:00PM

3:00PM

Vendor Exhibit Area open to users until 7:00PM

8:00PM

Grand Opening Warm-up (United A-B)

9:00PM

Second National HUG Conference — Official Opening (Rosemont)

Saturday, August 20, 1983

8:00AM

Registration Booth Opens

8:30AM

Vendor Exhibit Area Opens

9:00AM

Morning talks begin every fifteen minutes (speakers to be announced).

1:30PM

Afternoon talks begin every fifteen minutes.

5:30PM

Vendor Area and Registration close in preparation for dinner.

6:30 PM

Dinner warm-up with cash bar (United A-B).

7:30PM

Dinner, Keynote Address and Prize Drawings.

Sunday, August 21, 1983

8:00AM

Registration Booth opens for the duration of the Conference.

8:30AM

Vendor Exhibit Area opens.

9:00AM

Morning Talks begin at fifteen minute intervals.

1:30PM

Local HUG Club Gathering with presentation by Local Groups.

2:30PM

Questions and Answers at Local HUG Gathering

3:00PM

Closing Comments

4:00PM

Vendor Area and remaining activities close.

Next month we will provide a "Conference Thermometer" to let you know how close we are coming to the "full" mark for dinner attendees. Remember, for those of you only attending for a single day, special tickets will be available at the Registration Booth. These tickets do not include dinner and are valid for the date shown only.



Dear HUG,

A small bit of information. Commercial software bought to run on CP/M, requiring C run, limits you to the number of data files you can create if you use C run 207. On using double-sided double-density CDC drives and CBASIC, as provided by Heath/Zenith, it was found necessary to use C run 237 in order to utilize both sides of that 8-inch drive.

Thank You

Gerald R. Myers, MD
10267 N. Scottsdale Rd.
Scottsdale, AZ 85253

Dear HUG:

Though much of the information (REMark) is much too advanced for me, I do find a lot that is very useful and informative. I have benefitted a lot from the articles on CP/M. Between them, some help from Benton Harbor, some help from Dick French of the Seattle Heath Electronic Center, and a lot of my own sweat and skull-scratching, I've finally licked CP/M and have my H-89 with hard/soft sector disk drives talking nice to me.

Since I started into computing virtually cold last Christmas with my H-89 kit, I've got an awful lot of catching up to do to even get to be an "intermediate" (to borrow a term from skiing) in computing. However, I'm working on it. I'm even able to help my wife with her VIC20, when she has problems. Boy, I wish Heath had a "beginners" computer along that line. Even the simplest H-8 is much too advanced for a lot of folks getting into computing these days, and many don't have the persistence I had to have to solve my problems. Really, there is a place for such a computer in kit form. For the rank beginners, it must have the operating system and language resident in ROM. There will be a lot more computer operators, programmers, engineers, and service techs in the future, and they have to start somewhere. Heath did that in the past with high fidelity, stereo, ham radio, and electronic servicing, why not microcomputers?

I'm now 61 and will be retiring within a few years, but still I'm excited by what I see happening with microprocessors and microcomputers in the offices, drafting rooms, and shops.

Join with other
HUGgies
at the
**1983 NATIONAL
HUG CONFERENCE II**

Keep up the good work with HUG. I enjoy it a lot, and hope some day to be a contributing member.

W. P. Howell
1507 Putnam St.
Richland, WA

Dear HUG,

For the owners of the H/Z-89 using CP/M with two drives there are no instructions for testing the second drive that I can find in my meager supply of material. The test that is listed in the MTR-89 is great for the installed drive, but leaves something to be desired when a test for number two is required. It turns out that the problem is not insurmountable, however.

After examining the listing for the MTR-89, the turn-on word for the installed drive was located at 006254. The byte, 022, sets the bit at pin 14 of P-803 for testing this drive. This signal is called "SY 0 L" on my print.

In order to test the speed of the second drive we must get it turned on. This can be accomplished by setting the signal "SY 1 L". The turn-on byte for the second drive is 024 and must be substituted for the byte at location 006254. (Ed: The byte for the third drive is 030.)

One way to do this is to pick a spot of RAM that isn't being used and modify the program. A very nice spot for this is provided at location 040100.

Turn on your computer and the second drive, install a disk in each, and follow the procedure given in your operation manual, page 3-4, under "Computing Test". Substitute the following numbers for the ones given in the book.

At 040100 enter 041 instead of 076, at 040101 enter 371, at 040102 enter 006. Continue the program with 315, 100, 006, 076, 000, 062. 002, 040, 076, 024, 303, 255, 006. Your last entry should be at memory location 040117. After pressing the "return" key and getting the "H:" prompt, type "P(ROGRAM COUNTER)" and enter 040100. Press "RETURN" and type "G(O)", then "RETURN".

The second drive should now be running and the speed displayed on the screen.

There are, no doubt, many other ways to do the job but this one is handy, is easy, requires no other equipment, and works. Happy testing!!!

R.C. Perkins
P. O. Box 114
Debarry, FL 32713

Vectored to 47 ☞

Questions & Answers

(EDITOR'S NOTE: If you need answers to specific questions on software or hardware problems that would be beneficial to other users, please drop us a note, Questions & Answers, Heath Users' Group, Hilltop Road, St. Joseph, MI 49085. Please keep your questions brief and to the point. We will do our best to answer your question here in this column in future issues. Some Questions & Answers are contributed by Zenith Data Systems Software Consultation.)

Q. How do I access the ports using CP/M MBASIC (BASIC-80) and how do I switch the printout from the terminal to the printer?

A. CP/M only provides two logical devices for normal usage and MBASIC supports each with a specific command. The LST: device is output only and is supported with the LPRINT command. The CON: device is supported by the PRINT and INPUT commands. If you wish to have output that goes to either the printer or the console under program control, the most straightforward way of doing this is with a status flag and two separate print statements. The following example uses the flag LP which is set to 1 if the output is to be the printer and 0 if the output is to be to the console:

```
100 IF LP THEN LPRINT "Date ";D$ ELSE PRINT "Date ";D$
```

Q. Is it possible to have both hard sector (H-17) and soft sector (H-37) disks and associated controller boards on my H-89 at the same time and still be able to run under HDOS 1.6 and 2.0 as well as CP/M 2.2.03 using either type drive at any time?

A. Both HDOS and CP/M will support the hard and soft sectored format at the same time, however, the operating systems must be "configured" to recognize the two environments. An HDOS update, P/N HOS-5-UP, is an HDOS 2.0 update which contains a DK device driver and software for soft-sectored. The BIOS of CP/M must be reassembled using MAKEBIOS, which comes on the distribution disk of CP/M version 2.2.03.

Q. I have just completed the building of my H89A with HDOS 2.0 through a microcomputer course. Would it be to my advantage to buy the CP/M operating system? If so do I need to modify my computer?

A. HDOS and CP/M are two different operating systems, with each having strong points. You should evaluate your needs and review available software that will meet your needs. If HDOS has the programs to satisfy your software demands then only the HDOS operating system is needed. Generally, this is not the case. There is abundant software under CP/M. The most objective view is to use both operating systems and their subsequent software to your advantage. The bottom line is, this is a question that only the user can answer.

Q. How does a holder of Heath/Zenith applications software learn of new versions that become available, e.g. SUPERCALC, MAGIC WAND, etc.?

A. Zenith policy for updates is currently being evaluated and developed. In the meantime, all REGISTERED purchasers of a particular piece of software will be notified of any updates. If the registered holder has purchased the product less than one year from the announcement date of the update, the purchaser will receive the update. If the product was purchased over a year before the announcement date, the purchaser will be notified and asked whether he would like to purchase the update at a nominal fee.

The important thing to note is that the purchaser must be registered. When you purchase a product, it takes only a few minutes to register, and it is worth it!



Modem Tutorial 83

Part I

Walt Jung
1946 Pleasantville Rd.
Forest Hill, MD 21050

(Copyright 1983 by Walter G. Jung)

Getting Started

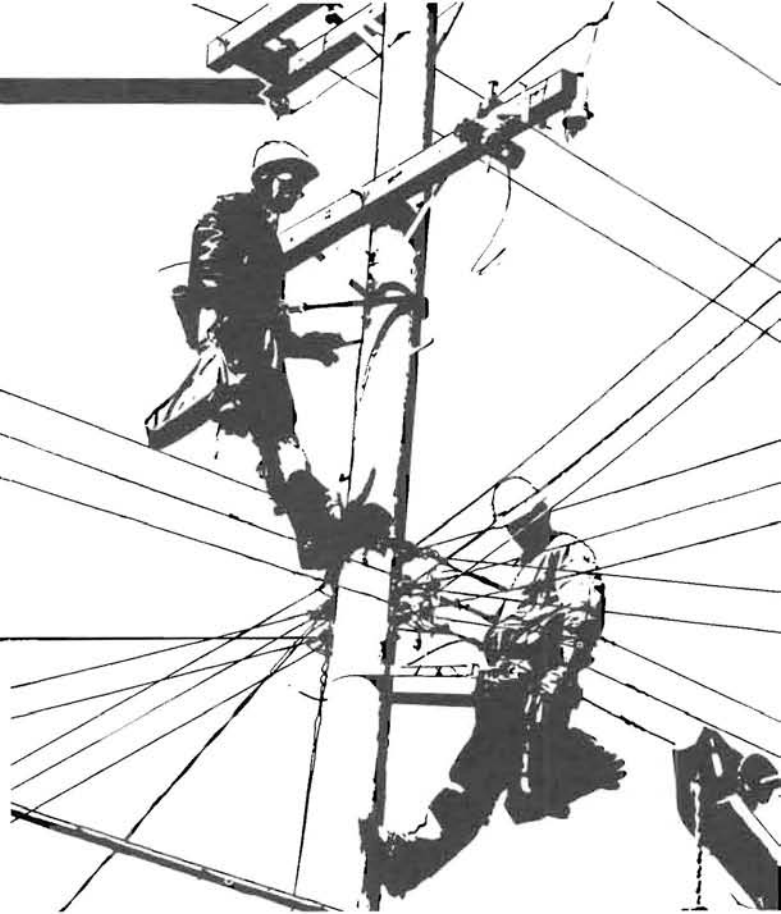
In virtually every computer magazine you pick up these days, you see a great deal of talk on the subject of modems, modem programs, and bulletin board systems. "Why all the fuss?" is probably a question which has occurred to many who haven't yet taken the plunge into the world of modems. I can honestly say that a thought or two along that line went through my head, before I got a modem and got involved WITH telecommunications. But, once that was done, I learned for myself the many useful things the right modem can do when used effectively. In this series of articles, I hope to share many of those ideas with REMark readers.

Before jumping into anything at all in the way of technical background, probably the most important thing to understand about modem communication is simply the *context* of what it does for you. Here, context means simply how it can change the way(s) you use your computer. No matter what Heath or Zenith machine you are using right now, regardless of what operating system you use on it, there exists a modem and software package which you can have up and running within 15 minutes of time. Further, you can then be ready to communicate with not only any other Heath/Zenith machine running any of the available operating systems, but other machines and systems, as well. To put it simply, the two main points I am making here is that modem communication is both immediate in its accessibility, and it also has the inherent power to reach across barriers. These barriers include not only the different disk formats, but even totally different operating systems. And, with the proper hardware/software combination, it can be just as reliable as you would ever desire...as reliable as swapping disks.

Whymodems?

Given the fact that a modem can deliver those two main performance features, let's break it down further to see what this can buy for you, and what things you can specifically do with a modem attached to your computer.

One of the most basic and easily understood things you can do is to simply dial up the local BBS, and send or receive messages to/from other computer users. If you are within a local call of an HEC, there is most likely already a BBS operating. On this system you can then talk to other Heath users and exchange program ideas, discuss hardware problems and enter a technical forum where you are exposed to multiple lines of thought at once, all relevant to your own uses. Even if you are not near a Heath oriented BBS, you will undoubtedly be near some type of BBS system, and you will still be able to communicate, since the common denominator used is ASCII.



A second thing you can do, if the BBS system supports it (and many of the HEC systems do, to some extent), is to send yourself (download) programs and text files, from the system. Or, if you are a prolific public spirit, you can send the system (upload) material you have written, and contribute to the data pool which keeps such systems alive. If you haven't been tuned into such activities before, the quality and amount of free software (for the price of a call and your time) you can obtain this way has to rank as one of the better bargains of all time. Of course, not all BBS systems have software available, but many of the more innovative HUG local groups often do (although sometimes by password access, for members only).

Thirdly, it is common today for many electronic firms to use large computer systems, which can be accessed by telephone, via a modem link. This allows employees to work at home, yet still have the power of the large computer at their finger tips to perform tasks beyond micro capability.

A fourth type of use is electronic manuscript submission. Today, many writers are increasingly going to electronic media for the submission of their work to the publisher. Obviously, this saves the editor the job of re-keying the entire article, and with only minor edits, a manuscript can be typeset. REMark uses this process now for disk and modem transfers of manuscripts. Many typesetters have already installed modem links, and can now accept 300 and 1200 baud transmissions. You can write it in the morning, and have it typeset in the afternoon!

A fifth modem use, one accessible to virtually everyone, is to access one or more of large mainframe time sharing systems. For HUG members, the HUG bulletin board on the Compuserve system is the best known example, but if you take a look at a Compuserve index you'll find that the HUG SIG (Special Interest Group) on Compuserve is just one of many SIG's, and SIG's are just one of the many Compuserve services. Compuserve, if you have some how managed to miss past references to it here, is a nationally based time sharing system. Although headquartered in Ohio, it is accessible throughout

the US and Canada, via local dialup nodes in major cities. We will cover Compuserve and the HUG SIG in a latter installment, but for now the important thing is that this "big BBS" is a *national* one, and you can go on line to send/receive messages to the thousands of HUG members who use it from time to time, and even talk key-key, in real time. Or, you can send or receive programs to the HUG SIG database, one of the largest of the various personal computing SIG software databases on Compuserve.

Using a time sharing system such as the HUG SIG example really personifies those points made initially. In doing so, you have immediate access to timely relevant information, and you can also send/receive programs and other files, over the barrier of your micro, to the giant DEC main frame.

If this is not sufficiently compelling, there is a sixth type of use you could put a modem to, with your computer. You can access other personal computers, such as those of your friends and local HUG members, to exchange programs. This allows beating the time lag of mailing disks, and most local HUGs already use such means to receive newsletter articles. A variation on this is to set up your own computer for remote dial up, by yourself, or others. You can even build your own BBS system on your Heath/Zenith machine (a later article will go into this, in some detail), and go "on line" for use. When you get to the point that you are doing this, you are then a bonafide modem freak, and will need no more coaxing from here or any where else...you are on your way!

All of the above is simply to set the stage for what is to follow, which goes into modems and what they are, modem programs and how to select them, a detailed look at the HUG SIG in particular and Compuserve in general, and a guide to getting your own BBS up and running. In this first installment, I'll cover what a modem is, both functionally, and as it is typified by current hardware. We will also see how the characteristics of the Heath/Zenith machines influence the selection of the modem and software.

Whatsamodem?

To answer the question of what a modem is and how it functions, I will purposely limit the discussion to those types which are classified as "external" modems, which means that such a modem connects to the computer via an RS-232 port, like a serial driven printer. This has the obvious advantage that *any* such modem is potentially useful, on *any* Heath/Zenith computer. By contrast, an "internal" (or buss) modem such as the PMMI MM-103 will work only with S-100 buss computers, for example as in the H/Z100 series. While this is a modem highly useful for that series of machines, it is not applicable at all to the H8 or H/Z-89, so it will not be covered (interested Z/H100 users can contact PMMI direct, at the address below).

The classic discussion of what comprises a modem starts out with the derivation of the MODulate-DEModulate MODEM acronym, as that's what it does, modulate on one end, demodulate on the other. With that obligatory explanation aside, a more important concept is simply that the modem is a tool which marries the digital data levels of the computer to the analog signal domain of the telephone. To do so, the modem must encode a digital bit pattern into a corresponding telephone system compatible signal, going into the line (modulate). At the other (receiving) end, it must reconvert the telephone signal back, into a facsimile of the original data (demodulate). But this is but one half the picture, as it only covers transmission to, but not from the distant computer. Such a one way system is "half-duplex" as information is transmitted in only one direction at a time.

Effective real time data transmission requires data flow in both directions essentially simultaneously, which is called "full duplex". This

is almost always used, and is the type of modem you will need for the various examples of modem communication, as described above.

Obviously, to transmit data in both ways at once, two signal channels must be used. These channels must somehow be made to live peacefully within the relatively restricted 300-3000Hz voice bandwidth of the standard telephone line. Further, acceptably low data error rates must be possible, through any of the typically used dial up facilities, anywhere in North America...across town, or across the nation.

The most common variety of modem communication is covered by the Bell system 103 standard, which defines an FSK (frequency shift keyed) full duplex system. Two channels are provided for, termed Originate and Answer. Each of these channels transmits data in a serial fashion, that is a given byte is defined by a series of (up to) 8 bits. In this manner the full ASCII character set, as well as binary files,

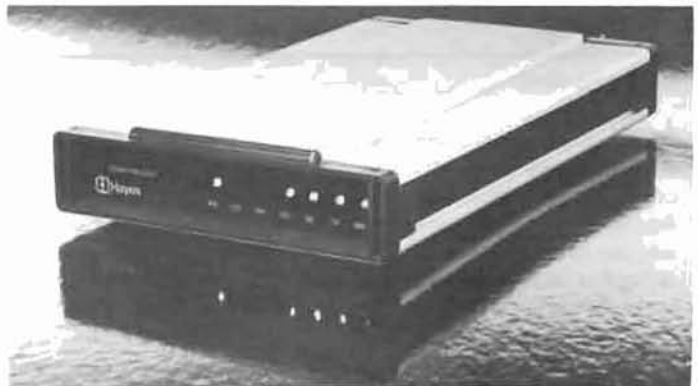


Figure One: The Hayes Smartmodem operates at baud rates of up to 300 baud, in accordance with Bell 103, with auto-answer, auto-dial direct connect features. (Courtesy Hayes Microcomputer Products, Inc.)

can be transmitted. The state of a given bit is represented by either of two tones, for the mark (1) and space (0) conditions.

Since the link always has a modem on each end, there is logically a duality of send/receive and originate/answer, between the calling and the answering modem. The calling modem is (usually) in the Originate mode for transmission, and uses for *modulation* a 1070/1270Hz tone pair. To receive this transmission channel, the answering modem is in it's Answer mode, and uses the same pair of frequencies, for its *demodulation*. The answering modem transmits back to the calling modem, in the Answer mode, by modulating on a 2025/2225Hz tone pair. The calling modem receives the return path data, demodulating using the same tone pair. In this manner, a full duplex data communication link is established. You should note that when one modem is in the Originate mode, the other must necessarily be in the Answer mode, for the data to be transferred. By convention, the calling modem is usually in the Originate mode, and the answering modem is in the Answer mode.

The tones associated with the Originate and Answer modes are also called *carriers*, in fact this is a more common usage. In establishing the initial contact, part of the scheme of operation is that the Originate modem goes on line, and awaits the Answer mode carrier from the other end. When the called modem answers the phone, it goes on line, and makes it's (Answer mode) carrier. When the Originate mode (calling) modem senses this carrier, it then turns on it's own carrier, and data communication is ready to begin.

Because of the bandwidth limits of the telephone system, there is an inherent upper limit to how fast data can be sent with this FSK system, before intolerable errors are seen. The standard baud rate is considered to be 300 baud, roughly equivalent to 30 characters per

second. Some Bell 103 type modems can exceed this rate with good results (such as the PMMI mentioned...600 baud). However, as a rule 103 style modems operate at 300 baud, and most BBS systems as well as CompuServe expect to see 300 baud, when you sign on. Anything higher should really be considered a special case, and not universal.

Bell 103 can generally be considered a low speed modem standard, and is limited to the rates as described (although with some effort this can be extended higher). To go appreciably above this on standard dialup lines requires a modulation technique basically different than that of FSK. Bell 212A is a higher speed standard, and uses a four level PSK (phase shift keyed) modulation method. This standard allows 1200 baud, full duplex rates, on standard lines. Bell 212A uses Originate mode carriers of 1200 and 2400 Hz for transmit and receive (respectively), while in the Answer mode it uses 2400 and 1200 Hz for transmit and receive (respectively).

Obviously, 1200 baud transmission is basically four times as effective as 300 baud, with everything else equal. Currently, more than a few 1200 baud BBS systems exist, with more coming on line all the time. CompuServe has 1200 baud access lines, as well as 300 baud. Still however, 1200 baud is right now without the universality of 300 baud. Most 1200 baud capability modems currently available are actually *dual* 300/1200 baud models, which makes them inherently twice as complex (since the two distinct standards require different circuit types). As a result, the 300/1200 baud modems tend to cost more than twice as much as a 300 baud only model. However, since the market for modems, particularly 1200 baud types, is growing by leaps and bounds, 1200 baud could easily become the standard that 300 is today, as prices drop. This article will cover both types, in terms of hardware, software and usage.



Figure Two: The Hayes Smartmodem 1200 operates at baud rates of up to 300 baud as per Bell 103, and at a baud rate of 1200, per Bell 212, with auto-answer, auto-dial direct connect features.
(Courtesy Hayes Microcomputer Products, Inc.)

Modem Hardware

As stated at the outset, a glance at any current computer periodical will show you a bewildering array of modems of all types. However, from all of these some trends can be seen, particularly in the RS-232 types, as are useful on Heath/Zenith computers. Breaking it down even further, modems can be categorized into two general classes, acoustic and direct coupled. As their names imply, the acoustic coupled unit uses your standard phone's handset, and talks/listens through acoustical couplers, at either end. The direct connect unit ties directly into your telephone line wiring, usually through the standard plug-in jacks and plugs.

What are the pros and cons of these two types? First of all, let's take the acoustic modem. It has the disadvantages of being noise sensitive, possesses a limited dynamic range, has no auto-answer or

auto-dial capability, operates at 300 baud only, can be used only with standard handsets, and it must be manually operated (it cannot be programmed). However, on its plus side, it is lowest in cost, it is simple, and it requires no FCC registration (since there is no connection to the phone lines).

Direct connect modems have a host of desirable features going for them, and their only real disadvantage is simply that they cost more. Their pluses are insensitivity to acoustic noise, greater dynamic range, higher baud rates, programmability, auto-dial as well as auto-answer capability, and in some units, even built-in intelligence.

In choosing between these two types, if the primary criteria is in terms of flexibility and performance, the direct connect type is the way to go. However, if you are on a budget or just starting out, an acoustic modem can be had for a very low tariff; see for example "Build the ECM-103, an Originate/Answer Modem", by Steve Ciarica, *Byte*, March 1983, p. 26.

Some basic things you should set as minimum performance in a modem, depending upon how you intend to use it with your computer. For example, if you plan to set up a BBS, a direct connect auto-answer modem is a must. Or, if you would want the modem to dial out at a specified time, a programmable auto-dial unit would be required.

A most interesting general class of modem within the category of the direct connect types is the *intelligent* modem, which has on board circuit smarts. This allows such features as a program command mode, re-dialing, command echo to console, diagnostics, loop back and so forth. The original Hayes Smartmodem as well as the more recent Smartmodem 1200 (shown in figures 1 and 2) are examples of this type of modem, and either unit works very well with any of the Heath/Zenith equipment. ("Smartmodem" and "Smartmodem 1200" are trademarks of Hayes Microcomputer Products, Inc.)

Both units are programmed quite simply, with ASCII command strings, via your RS-232 serial port. An abundance of software is available for the Smartmodem, and since the command sets of both the low speed and dual speed models are virtually identical, programs work on either, as long as some means of setting the baud rate is available. Both Smartmodem units are highly popular, with "Smartmodem compatible" look-alike units already showing up. The original Smartmodem operates over a range of 110-300 baud (Bell 103), while the Smartmodem 1200 works over this range, plus 1200 baud (Bell 212). The two models are available either through Heath/Zenith, or other suppliers, and those interested in further technical data can write to Hayes at the address below (mentioning this article).

In the next installment, I will get into the features necessary and desirable in a modem program, cover the considerations involved with driving the two Smartmodems effectively, and discuss hands-on some modem programs useful on Heath/Zenith computers. In just a short time, you will literally be able to talk to the world, via your modem. Questions and/or comments are welcome, and can be sent C/O REMark, to CompuServe ID 70001,756 on the HUG SIG, or left on the BHEC RCPM, at (301) 661-2175. Happy modeming!

Further information:

Smartmodem and Smartmodem 1200

Hayes Microcomputer Products, Inc.

5835 Peachtree Corners East
Norcross GA, 30092

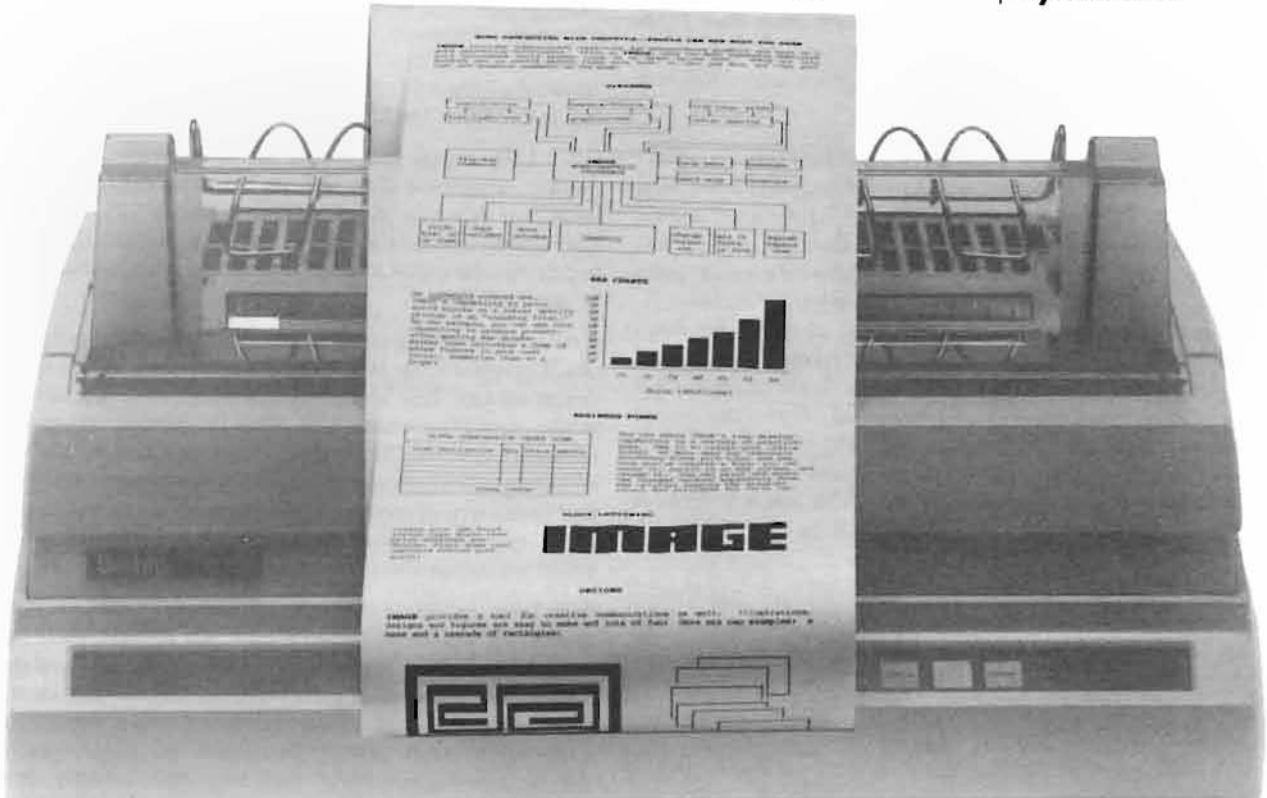
MM-103

Potomac Micro-Magic Inc.
Three Skyline Place, Suite 604
5201 Leesburg Pike
Falls Church VA, 22041



WORD PROCESSING IS NOW MORE THAN PROCESSING WORDS.

ANNOUNCING **IMAGE™** FOR **ZENITH** data systems



WORD PROCESSING WITH GRAPHICS... PEOPLE CAN SEE WHAT YOU MEAN.

IMAGE adds a new dimension to word processing... graphics! Now, you can combine text and linear graphics to create *BAR CHARTS, FORMS, ORGANIZATIONAL CHARTS, FLOW CHARTS, BLOCK LETTERS*, and much more. The result? More powerful and more effective communication. People can *SEE* what you mean.

TOP MARKS FROM INFOWORLD

IMAGE won top marks from *INFOWORLD* for its innovation, quality, reliability, and ease of use:

"*IMAGE* certainly deserves accolades in the *performance* category."

"The *documentation* is simply superb. It is professionally done from cover to cover."

"Without a doubt, this program is *easy to use*."

"The program is *bombproofed* so well that I had trouble finding any errors."

IMAGE is a trademark of MicroArt Corporation.

InfoWorld
Software Report Card

Image

	Poor	Fair	Good	Excellent
Performance	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Documentation	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Ease of Use	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Error Handling	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>

All quotes are from InfoWorld's *IMAGE* software review, by Marty Petersen, June 14, 1982.

Copyright 1982 by Popular Computing, Inc. a subsidiary of CW Communications, Inc., Framingham, MA—Reprinted from InfoWorld.

EXTRAORDINARY VALUE...

"The modest \$295 Price Tag is a Bargain."

IMAGE runs on any Zenith Z/89 or Z/90 computer, on any Heath H/89 or H/90 computer, or on any Z-80-based CP/M system linked to a Z/19 or H/19 terminal.

TO ORDER CALL TOLL FREE:
1-800-MICROART (1-800-642-7627)
in Oregon, call: **1-692-3950**

OR WRITE: MicroArt Corporation
200 Market Bldg.
Suite 961
Portland, OR 97201

Mastercard, Visa and COD orders accepted.

The WCCF in HUGger <TLJ> Perspective

*Terry Jensen
Software Developer*



The West Coast Computer Faire (WCCF) is one of the largest and most prestigious personal computer shows in the world. This year the show covered the San Francisco Civic Auditorium and adjacent Brooks Hall with every nook and cranny filled with booths. Next year promises to be even larger, including a new hall located a few blocks from the Civic Auditorium. The attendance for the show was up about 10,000 over the previous year to about 47,000.

This was my first year to attend the WCCF and as it happens, I was there working the Heathkit booth, along with several Heath/Zenith employees. I had the opportunity of representing Heath Users' Group (HUG) at the show. Most of my time was spent working the booth or preparing for the HUG meeting which was held on Sunday. However, I did get around a little to take pictures and see what was going on around the show.

Part of my responsibility in going to San Francisco was to write a short expose' of my experience at the WCCF. I will probably have a different view of the show than most everyone else but then that's me!

I found, bright and early Friday morning before the show was even open, Ray Livingston of Livingston Logic Labs walking around checking out the booths (and he wasn't even a vendor). A little while later, Steve Bard, of Micro-Widget Works, and his wife introduced themselves to me. I have "known" Steve for a long time on the HUGBB and finally got to meet him face to chest. (Steve is little fella, about 6'-4".)

Ah, Dean Gibson was around there and of course, he was present at the HUG meeting trying to get Barry Watzman off his guard. Rick Bates (of the HUGBB) was there helping me out with questions about the H/Z100. (Let's see I was the one working the booth.) Oh boy, I could go on and on about all the great people that came up to chat with me.

The above is an example of the typical great time that I have each time I meet and talk with new and old Heath/Zenith users. In addition to talking with them, the highlight of my time in San Francisco was leading the HUG meeting which I will explain later. However, from the moment I arrived at the show, I questioned the high regard for, and the mystic about, the WCCF.

My impression of the WCCF is that there were actually three shows going on at the same time. There are the arcade bonanza's, which

get the most attention and publicity. Then, there are the serious vendors showing what their particular computer can do or the support they offer for a particular computer. (Heath and its vendors fell into this category.) The last "show" was simply stated, the IBM illusion; nothing more needs to be said about that.

John Matlock of MPI called it right when he said "Brooks Hall is a zoo". Brooks Hall seemed to have more spectators than the Civic Auditorium. It was Brooks Hall that received the attention of the local television stations (per the evening news as I crashed after the show each night). It was clogged with arcade type booths, which of course appealed to the game freaks visiting the show. The booth space in Brooks Hall is generally helter-skelter, because the vendors are scattered around wherever they can get a booth. It was difficult, I am sure, for the serious vendors to deal with the noise and confusion in Brooks Hall.

The Civic Auditorium was far from quiet, but it did lend to a more "relaxed" and (if I may) a more professional atmosphere than Brooks Hall. The Heath Company and related support vendors molded very nicely to help create this supportive atmosphere by being placed together in a common area of the Auditorium.

CompuServe (the time-share system of which the HUG Bulletin Board is a part) used a ZT-1 terminal within the Heathkit booth. Bill Loudon was there to show the large CompuServe data base including the Heathkit Catalog, which is now on-line.

Magnolia Microsystems was located next to the Heathkit booth with all their neat hardware and software including the Corvus hard disk network. Patient Kay Gjerding, as she should be called, hadn't received one of her crates from storage by the time we were already done. (And I was running out of patience with the Union for their mixed up procedure of storing our 13 crates in five separate truck-trailers.)

Right across from the Heathkit booth was Sextant magazine and the Buss Newsletter showing their support for Heath/Zenith. I was elated that Charlie remembered who I was. (I am not exactly the most well known HUGgie around.) It was great to see Al Dallas again and meet Fred Zimmerman for the first time.

Trionyx and Microtran shared a booth displaying their support for the H-8 computer. Bill Perry and Myron Seibold along with Rick Lutowski showed their selection of H-8 boards.

Joe Garguillo and Dave and Barry Murray of Evryware and Janet Hoyle of Hoyle and Hoyle showed off their software products, while sharing a booth. I think it is great that vendors can share booth space, even though they may have products that are in "competition".

Behind them, Howard Nurse, and Kirk Bowersock of Commsoft were displaying the software they have available, including their amateur radio goodies. Herb Drake, the author of the program ROOTS, worked the booth and attended the HUG meeting for Commsoft.

The other island across from the Heathkit booth, known as the HEX Island, had several more vendors with their excellent support. Remember that all of the Heath related booths are grouped fairly close together in one area, rather than being scattered all over the show arena.

John Matlock of MPI was there showing off the MPI 150 highspeed printer, while sharing a booth with Studio Computers. You may be interested in knowing that some of the vendors, for example Ray Massa of Studio Computers, worked the show on their vacation time. Ray's wife, Nancy, and their two children were there in San Francisco enjoying the rain along with everybody else.

Next to them, TMSI was showing off the powerful H1000. Tom Snoblen and Lee Hart of Technical Micro System Inc., are quite proud of their new dual processor board for the H89.

Around the corner, MicroMaster displayed their hardware including the two narrow five inch drives inside the H89 cabinet. John Kennan had a busy few days with the location of his booth right at one of the main entrances into the Civic Auditorium.

On the back side of the Hex Island, Mark and Laura Brooks of CDR were there with their hardware, along with Dick Fox and Betty Chamberlain of Professional Systems and Applications (PSA) with their support.

Kres Engineering was there displaying their many hardware products. Ken Smith, Bob Koepke, Pat Diehl, and Kreg Smith seem cool and calm when it comes to working computer shows. Pat even had her beautiful Samoyed there after the show.

Of course, Walt Bilofsky of The Software Toolworks was there selling software products and displaying some computer equipment. Susan Haze, Gal Halverson, and Jim and Maritta Gillogly were there working the booth also.

I am sure for these vendors the WCCF was a success!

For little ole me the biggest event of the WCCF was the HUG meeting! Why should I be nervous, as I have been in front of groups many times before? Was it because Boss Bob told me that if the meeting goes great, the HUGgies won't ask many questions, but if the meeting goes bad ... look out? Could it be because nobody would know who I am, and perhaps they would not respond to my "position"? Or was it because we had asked for a room that would seat 350 to 500 people and received instead a room of 144 seating capacity? Well, whatever the cause, Saturday night I wore out the carpeting on the floor in my hotel room. Let's see ... everyone has seen the slide show before, but the "slide show must go on" as they say. So, once more over the notes for each slide. What can I say that will keep their interest for two hours? One more time across the floor.

Finally, the big moment arrived!! Not even standing room only! The meeting began. First, I had to introduce myself because after all nobody (but Charlie) knew who I was. Oops, I was mistaken, there were five people there who knew who I was (bless their little hearts!).

Next came the important announcements. First, I was leading the

HUG meeting! (I think the laughs were from a private joke.) The main announcement was, of course, that we had two new slides for the show this year!! Everyone applauded!! They even applauded more when I told them I would save the slides for last, just in case we didn't have enough time to see them!!

On we went with the introduction of vendors. We always make mention of them, you know? After all, that is what separates Heath/Zenith from other computer companies. (I must confess though and apologize publicly to Kay, that I forgot to introduce Magnolia Microsystems.)

What about the local HUG Clubs? There were the typical California groups that always come. Charles Bennett was there from Mission, Kansas. Oh, as I recall there was a group from Seattle, a couple from back east and a few from Tim Buck Two. Jim Branchaud was there modeling the snazzy new T-shirts of Hawaii HUG.

Next, we had to find out which computer everyone had. Of course, there were H/Z89's and H/Z100's. But there were few machines of which I had never heard of before. Let's see...there were several users there with H-8's (I think they're called). And Kurt Schultz was bold enough to announce he was still using cassette (did I spell that correctly?). Amazingly, the three people with H11's didn't throw rocks and tomatoes. There may even have been a few ET-3400 type individuals.

After all the exciting questions regarding what HUG is doing, I was able to rest my case, while Wayne Wilson, Product Line Manager for Heathkit General Products, did a HERO-1 demonstration. Wayne and HERO did a superb job entertaining the HUG mob, (who were patiently waiting for their chance at Barry). That little guy (HERO-1) is so cute. He even laughed at my crummy jokes.

At that time, I turned the meeting over to Barry Watzman, ZDS Product Line Director. Barry informed the group that the H-100's, Z-29's, and Z-100 Winchesters are ready to ship. He also announced that the Z-100 prices have been reduced. All in all, to everyone's dismay, there was no real earth shattering news.

After cutting off Barry, we drew for the three winners for this years show. Each winner would have his/her choice of any one software product from the Heath/Zenith Catalog. Dot da daaa ... HERO-1 announced the first number; 113. The second number; 047. And the third; 102.

At that time, the meeting was excused and everyone was adjourned, because those riotous Atari freaks were trying to say that their meeting (ha, I bet) was to start five minutes ago. The gall of such accusations!

It turned out the three winners were Jim Jensen of San Carlos, California (no relation), Maureen Hackley of Aloha, Oregon, and Jack Van Zandt of Menlo Park, California. All three winners chose Condor, with Jack choosing the 16 bit version for ZDOS. We quickly took a mug shot and departed for another year. (Typically as would happen, out of the four camera's taking pictures at the meeting, the roll of film that had the picture of the three winners was lost at development. Really! My apologizes to Jim, Maureen, and Jack.)

In conclusion, the WCCF can easily be a show where computer users of all brands square off to display their "great" stuff and cut down the rotten junk produced by their competition. Or the WCCF can be the joining of forces of a particular computer brand to show the support for each other through the charisma of the people that make up the support group. The WCCF can be either of these or whatever you want to make it out to be. I guess that is the reason for the popularity and mystic of the West Coast Computer Faire.





Pat Swayne
Software Engineer

Teaching HERO A Thing Or Two

(A Cross Assembler for the ET-18 and ET-3400)

If your friends are no longer impressed that your HERO 1 robot can say "Freddy" (or something like that) when you press his reset button, or repeat a few simple moves that you programmed in with the teaching pendent, maybe it is time that you taught him a thing or two. So you get out your Robot Technical Manual and start working up a program using "Robot Language" codes.

Suppose you want to move the robot's head to position 30 (hex), and perform some other task while the head is moving. So you first decide that Robot Language code CC (hex) is the one to use, and you punch that number in. Now, you have to figure out the two byte argument that goes after this code. Since it is the head you want to move, the most significant bits of the first byte should be 010, and you want to move at slow speed, so the next two bits should be 01. So the first of the two numbers is 01001000 in binary, so all you have to do is convert that to hex and punch it into your robot.

After a few bytes of that process, you start thinking that maybe programming this robot is not going to be as much fun as you thought it would be. It would be much easier if the robot had a keyboard, and you could type into it "Move Head to 30". Well, now it is almost that easy with the HUG XMET cross assembler. It allows you to write your programs on a CP/M or HDOS compatible computer using easy to remember mnemonic representations of the robots commands along with the entire 6800 instruction set. The end result will be the correct hex numbers figured out for you so you can just punch them in, or you can use a downloading technique such as the one discussed in last month's REMark, or one that will be presented in this article (using the ETA-3400 as an interface), to load the program directly into the robot. For example, to move the head to position 30 in the continuous mode at slow speed, you would enter

```
MCA    #HEAD+SLOW+30H
```

MCA is a mnemonic which stands for "Motor move, Continous mode, Absolute positioning". HEAD is pre-defined by XMET to the correct bits representing the robot's head, and SLOW is defined to the bits meaning slow speed. The only number you have to figure out is the one indicating the position of the head. XMET also includes definitions of ROM subroutines such as INCH and OUTBYT, and definitions of the most important ports and RAM areas, such as the motor position bytes.

A Few Preliminaries

Most HUG members use computers that have 8080 or Z80 proces-



sors, and may not be familiar with the 6808 used in the robot, or with 6800-type assemblers. Because of this, I have included this brief discussion.

The 6808 Processor

The 6808 (a newer version of the 6800) is a memory oriented processor, not register oriented like the 8080 family processors. To illustrate this, suppose you want to increment the contents of a memory location using an 8080. To do it, you either have to get the contents of the location into a register, or the address of the location into a register, as follows.

```
LDA    ADDRESS    ;GET VALUE FROM MEMORY
INC    A          ;INCREMENT IT
STA    ADDRESS    ;REPLACE IT
```

or

```
LXI    H,ADDRESS ;POINT TO VALUE
INC    M          ;INCREMENT IT
```

With the 6808, however, you can work directly on the memory cell.

```
INC    ADDRESS    ;INCREMENT VALUE AT ADDRESS
```

On the other hand, suppose you want to OR the contents of two registers. With the 8080, you can do it directly, as in this instruction.

```
ORA    B          ;OR A AND B REGISTERS
```

With the 6800, however, you must place the contents of one of the registers into a temporary memory location, because you can only do this kind of operation on memory locations, not registers.

```
STA    E    TEMP ;STORE E REGISTER CONTENTS
ORA    A    TEMP ;OR A CONTENTS WITH TEMP CONTENTS
```

I think that you can see from this that both processor types have their advantages and disadvantages. Just remember to orient your thinking to the kind of processor you are using, and you can write efficient code for either type. I have seen 8080 programs written by 6800 programmers (and vice versa) that were not done too well because the

programmer was still thinking in terms of the other processor.

If none of what I have said makes any sense at all to you, you should study the microprocessor sections of the Heath Robotics Course (EE-1800) before you attempt to program your robot.

6800 Assembler Conventions

Not only are 6800 family processors based on a different philosophy from 8080 family processors, 6800 assemblers also follow different conventions. The basic definition of a line of assembly code is the same for either type:

```
LABEL  OPCODE  OPERAND          ;COMMENTS
```

The main difference is in the way the various addressing modes of the 6800 are specified. If there are two or more instructions that do the same thing but use different addressing modes, they will have the same mnemonic representing them, with some kind of special character or characters in the operand area indicating the addressing mode. Since you may not be familiar with 6800 addressing modes, I will discuss them briefly in addition to pointing out how they are specified in assembly code.

The 6800 has 6 addressing modes: inherent, immediate, extended, direct, index, and relative. The inherent mode is like the 8080 implied mode. In this mode, no actual addressing of memory is involved, but a register may be altered, as with INC A (increment the A register). These instructions do not take an operand. The "A" in INC A is not an operand, but part of the opcode, and with some 6800 assemblers, it would be written as "INCA". With the XMET cross assembler, all opcodes involving the accumulators (A and B registers) are written with at least one space separating the main body of the opcode and the register designation.

The immediate addressing mode is where the operand is itself the data to be operated on by the opcode. The normal convention for specifying immediate addressing is to place a "number sign" (#) before the operand. Remember our robot example?

```
MCA    #HEAD+SLOW+30H
```

This is an example of immediate addressing. The expression "HEAD+SLOW+30H" represents the actual data to be operated on by the Motor command MCA.

The extended addressing mode is where the operand is an address of a memory location containing the data to be operated on. This mode is assumed by the assembler when no special characters are used. If we leave off the "#" sign in the example above, the assembler will evaluate the expression "HEAD+SLOW+30H" as an address, and if the resulting program is run, the robot will attempt to move a motor using the data at that address, with unpredictable results! A correct example of extended addressing is

```
MCA    MOTDAT
JMP    ANYWHERE
MOTDAT FDB    HEAD+SLOW+30H
```

Here, MOTDAT is the address of a location in memory that contains information for moving the head motor. As with the 8080, extended addressing is also used for control transfer (jumps) in a program. In this example, ANYWHERE is the address that control is transferred to by the JMP instruction. FDB in this example is equivalent to the DW pseudo opcode used in 8080 assemblers, and with XMET you can use either FDB or DW.

The direct addressing mode is a special version of extended addressing for memory locations in "page 0" of memory, that is, memory address less than 100H. With it, the address can be specified using one byte instead of the two normally required. Some 6800 as-

semblers will automatically use direct addressing if the address evaluates to less than 100H, but with XMET, it must be specified using ",D" after the operand. This allows the programmer the option of using either direct or extended addressing in page zero of memory.

```
ADDRESS EQU    10H
LDA A    ADDRESS,D
LDA B    ADDRESS
```

In this example, the contents of ADDRESS are loaded into the A and B registers, using direct addressing for A (two bytes of code generated), and extended addressing for B (three bytes of code generated).

The index addressing mode is where the value of the operand is added to the contents of the X register in the processor to arrive at an address. It is specified by placing ",X" after the operand. This mode is most often used to extract data from tables, as in this example

```
LDX    #TABLE    ;POINT X REGISTER TO TABLE
MCA    4,X        ;MOVE HEAD ACCORDING TO 3RD ENTRY
JMP    THERE

TABLE  FDB    HEAD+SLOW+20H ;MOVE HEAD TO POS. 20
      FDB    HEAD+SLOW+40H ;MOVE HEAD TO POS. 40
      FDB    HEAD+SLOW+60H ;MOVE HEAD TO POS. 60
```

Note that an offset of 4 gets us to the third table entry. This is because each entry is two bytes, and so the second entry is two bytes from the label TABLE, and the third entry is 4 bytes from the label. It is important that you take into consideration the size of each table entry when you calculate an offset for index addressing.

The relative addressing mode is used exclusively for program control transfers. The operand is an offset added to the program counter to arrive at an address to transfer control to. To distinguish between transfers using extended addressing and those using relative addressing, extended transfers are called "jumps", and the opcodes involved start with J, while relative transfers are called "branches", and their opcodes start with B.

The XMET Cross Assembler

The XMET cross assembler will assemble all of the 6808 processor's instructions using standard 6800 mnemonics, and has additional mnemonics to represent all of the codes in ET-18 Robot Language. Here is a list of the Robot Language mnemonics.

Op-Code	Operand	Hex Code	Meaning
ADM		02	Abort Drive Motor
ASM		03	Abort Steering Motor
AAM		04	Abort Arm Motors
BIBB	byte	10	Branch If Base Busy
BISB	byte	1D	Branch If Steering Busy
BIAB	byte	1E	Branch If Arm Busy
BISP	byte	1F	Branch If SPeach busy
ZERO		21	Zero motors
REX		3A	Return to EXecutive mode
ORL		3F	Change to Robot Language (6800 SWI)
ELD		41	Enable Light Detector
ESD		42	Enable Sound Detector
EUF		45	Enable Ultrasonic Ranging
EMD		48	Enable Motion Detector
EDIS		4E	Enable DISplay

DL0	51	Disable Light Detector
DS0	52	Disable Sound Detector
DUR	55	Disable Ultrasonic Ranging
DMD	58	Disable Motion Detector
DDIS	5E	Disable DISplay
SFC	byte,X	61 SPEak, Continuous mode (index)
SPW	byte,X	62 SPEak, Wait mode (index)
SFC	word	71 SPEak, Continuous mode (ext.)
SPW	word	72 SPEak, Wait mode (ext.)
CML	83	Change to Machine Language
SLEEP	#word	87 Sleep
PAUSE	#word	8F Pause
JISF	word	BF Jump If SPEaking
MWA	#word	D3 Motor move, Wait mode, Absolute
MCA	#word	DC Motor move, Continuous mode, Absolute
MWR	#word	D3 * Motor move, Wait mode, Relative
MCR	#word	DC * Motor move, Continuous mode, Relative
MWA	byte,X	E3 Motor move, Wait mode, Absolute
MCA	byte,X	EC Motor move, Continuous mode, Absolute
MWA	word	F3 Motor move, Wait mode, Absolute
MCA	word	FC Motor move, Continuous mode, Absolute
MAA	#17 bytes	FD Motor move, All Absolute

Notes: The term "byte" refers to any 8-bit number, and "word" refers to any 16-bit number. The motor relative commands (marked with "**") cause motor movements relative to the motor's current position. They do not use relative addressing. The MAA command takes as its operand 7 bytes separated by commas and enclosed in brackets.

In addition to mnemonic representations of Robot Language instructions, the XMET cross assembler also has three special pseudo opcodes that cause the inclusion of special definitions. The opcode RMOT causes motor definitions to be included, such as HEAD, which equals the bits required to select the head motor. Speed and direction definitions are also included. The opcode RRAM adds definitions of special RAM locations and ports used by the robot, including the motor position counters and the clock ports. The opcode RROM adds the definitions of the user available ROM subroutines.

Below is a sample program for the robot as written for the XMET cross assembler. This program is from page 27 of the ET-18 Robot Technical Manual.

```

; SAMPLE ROBOT PROGRAM
; THIS PROGRAM CAUSES THE ROBOT TO MOVE FORWARD
; WHEN IT DETECTS A SHARP SOUND, AND TO STOP
; IF IT DETECTS ANOTHER SHARP SOUND WHILE MOVING.

RMOT ;INCLUDE MOTOR DEFINITIONS
RRAM ;INCLUDE RAM DEFINITIONS
ORG 450H
START ESD ;ENABLE SOUND DETECTOR
PAUSE #2*16 ;WAIT 2 SECONDS
LOOP BSR LISTEN ;LISTEN FOR SOUND
MCA #DRIVE+SLOW+OFFH ;DRIVE FORWARD OFFH UNITS
PAUSE #1*16 ;WAIT 1 SECOND
BSR LISTEN ;LISTEN FOR SOUND
ADM ;ABORT DRIVE MOTOR
PAUSE #1*16 ;WAIT 1 SECOND
BRA LOOP ;KEEP LOOPING

LISTEN LDA A SENSE ;READ SENSE PORT

```

```

CLC ;CLEAR CARRY
CMP A #30H ;TEST SOUND LEVEL
BCS LISTEN ;LOOP UNTIL LOUD NOISE
RTS ;RETURN ON LOUD NOISE
END START

```

The XMET cross assembler not only makes it easier to write robot programs, but it makes it easier to modify them. For example, suppose you want the robot to move at medium speed instead of slow in our sample program. Instead of trying to figure out the new number representing medium speed, you just change "DRIVE+SLOW+OFFH" to "DRIVE+MED+OFFH" and re-assemble.

How XMET Works

The XMET program is actually a translator, not a true cross assembler. It translates the source file into a list of DB statements that can be assembled by either the CP/M or HDOS 8080 assembler. Utility programs are provided to convert the result into MIKBUG(tm) format hex files (for downloading into the robot using the method described below) or an easy-to-read hex format if you want to manually key in the program.

If there is anything in the source program that XMET cannot recognize, the line containing it is passed through unchanged. That means that you can include special directives for the 8080 assembler that will be used to assemble the translated program.

Downloading with the ET-3400

If you have an ET-3400 or ET-3400A Microprocessor trainer and the ETA-3400 accessory, you can use it to load programs developed with XMET into your robot. First, convert your assembled robot program into MIKBUG format using the utility supplied with XMET. Now, connect the ETA-3400 serial input to the modem output of your H8, H89, etc. computer, and use a terminal program such as ZTERM or HTERM to make your computer the terminal for the ETA-3400. Start the ETA-3400 monitor program, enter L0 to load a MIKBUG file, and use the transmit command of ZTERM or HTERM to send your robot MIKBUG file to the ETA-3400. Once the program is in the ETA-3400, you can either use it to make a cassette tape of the program, which can then be read by the robot, or you can connect the ETA-3400 tape output directly to the robot tape input. Then start a tape load on the robot, and a tape save on the ETA-3400, and the program will be transferred directly.

Note: See product listing on page 26 for pricing and further information.



Missing Photos Found: Terry congratulates HUG's West Coast Computer Faire winners Maureen, Jack and Jim.

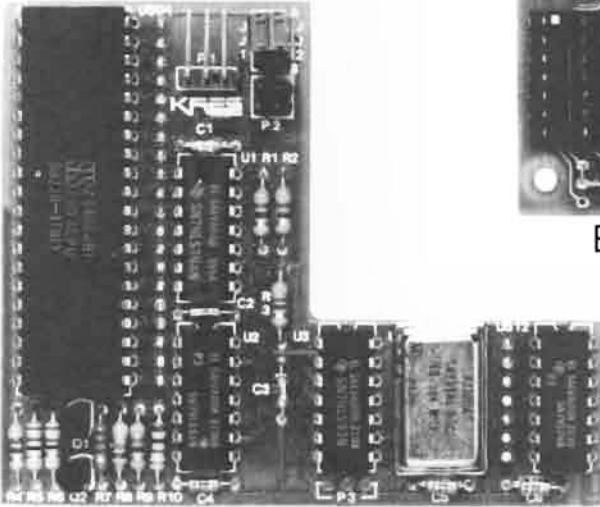
MORE SOLUTIONS

from KRES

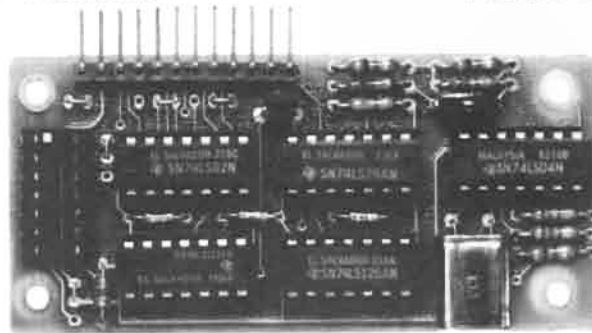
the DUAL SPEED MODULE

a faster solution

Now Available in Two Versions



DSM-240 for the H/Z 89-90



ESM-240 for the Expansion

Why run half speed when you can run full speed? Install one of these KRES Modules in your H/Z 89-90 and cut computation time in half. The supplied HDOS or CP/M software will allow you to operate at regular or double speed (2 or 4 MHz).

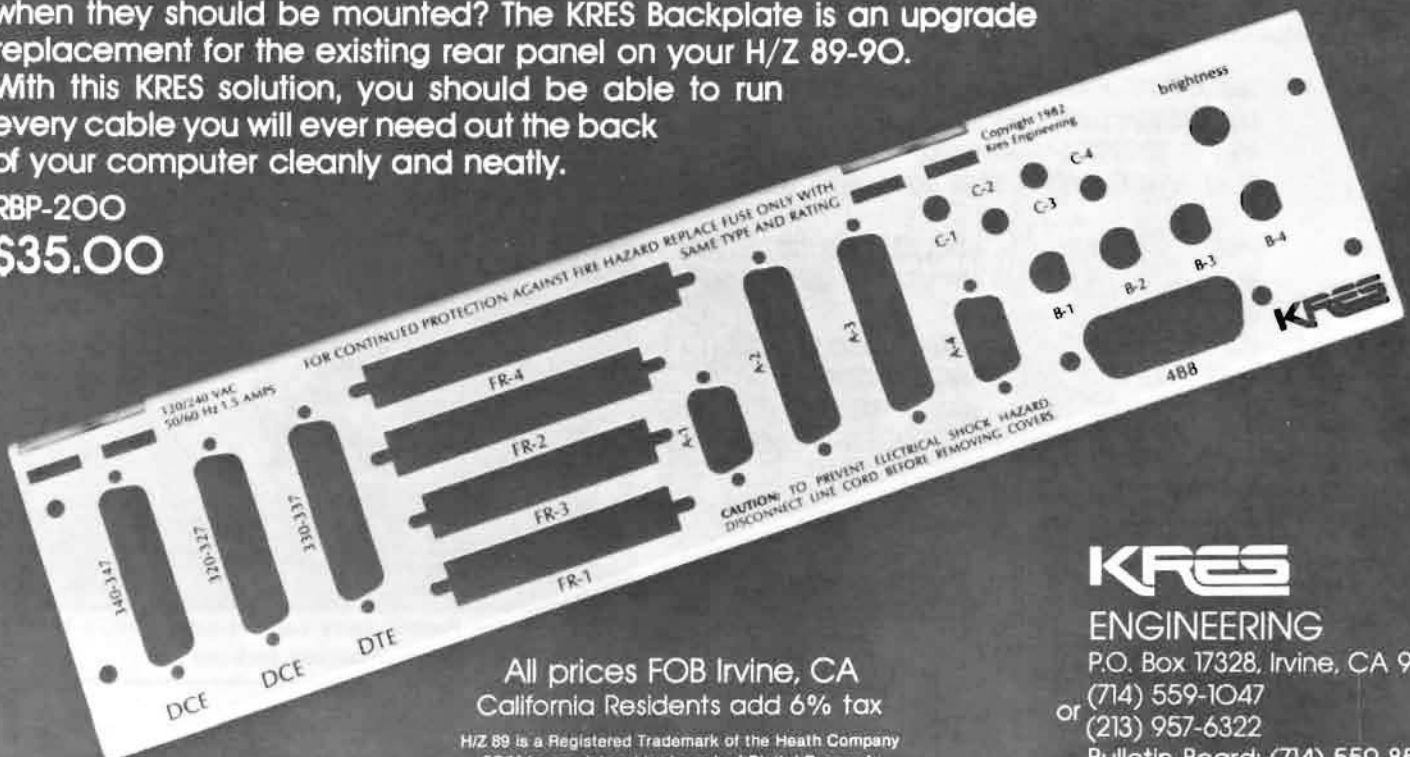
Either version with software **\$79.95**

the BACKPLATE

a neater solution

Are you tired of having wires and cables stuffed out the holes of your computer when they should be mounted? The KRES Backplate is an upgrade replacement for the existing rear panel on your H/Z 89-90. With this KRES solution, you should be able to run every cable you will ever need out the back of your computer cleanly and neatly.

RBP-200
\$35.00



All prices FOB Irvine, CA
California Residents add 6% tax

H/Z 89 is a Registered Trademark of the Heath Company
CP/M is a registered trademark of Digital Research

KRES

ENGINEERING

P.O. Box 17328, Irvine, CA 92713

(714) 559-1047

(213) 957-6322

Bulletin Board: (714) 559-8579



Introduction To Z-BASIC

Part VI

Gerry Kabelman, C.E.T.
Zenith Data Systems

This is the sixth article in a series of articles dealing with the new commands of the Z-100's Z-BASIC over BASIC-80. Previous articles dealt with several graphic commands and this article is the finish of the flag we began in Issue 37.

This series started with Issue 34 of REMark (November 1982). As a reminder, limited back issues of REMark are available and for those issues that may not be available, a REMark volume for each year is available from the Heath Users' Group. Check with the HUG parts list in this issue for information on the REMark volumes.

Last month we added a blue background to the thirteen stars and a flag pole. As indicated at the end of the last article we will be finishing the flag this month.

To finish the flag we need to add the red and white stripes. As stated last month we can do this by adding three lines of code, however doing some additional checking shows that only two lines of code are required for the stripes of the flags.

Using the last listing in last month's article add the following two lines.

```
64 FOR I=0 TO 12:CL=4:IF I/2<>I\2 THEN CL=7:' Red & White
65 LINE(40,I*12+20)-(600,I*12+32),CL,BF:NEXT I:' Stripes
:
```

The above two lines look very complex but if you look at them one at a time they make a lot of sense.

Line 64 starts our FOR-NEXT for the thirteen (0 to 12) stripes of the flag. The variable CL is the color of the stripe. (CL=4 is Red and CL=7 is White.) We first set the variable CL to four (Red) and then using the IF-THEN statement we check to see if the value of the variable 'I' is odd or even. If the IF-THEN statement is true (odd) then CL is set to seven (White). For those that have not used the backslash (\), it is used for integer division. In this case we are first dividing the variable 'I' by two, using floating point division (/) and then we divide it by two, using integer division checking to see if both are equal. If they are not equal we set the value of CL to seven (White) and if they are equal the value of CL is left at four (Red).

Line 65 actually draws the stripes using the LINE command for drawing a box that is filled. The variable is multiplied by 12 for stripe size and an offset of 20 is added for the top of the stripe and an offset of 32 is added for the bottom of the stripe.

Try changing line 65 to the following line and note difference on the screen as the stripes are displayed on the screen.

```
65 LINE(40,I*12+20)-(600,176),CL,BF:NEXT I:' Stripes
```

Let's add another line to connect the flag to the flag pole so the flag doesn't fly away.

```
63 PSET(30,15),7:DRAW*M39,20D155M37,233:' Rope
```

Line 63 first sets the dot pointer to the top of the flag pole, drawing a line to the top of the flag and then draws a line straight down the left side of the flag angling it back towards the pole at the bottom.

This completes the flag with thirteen stars and stripes, however, American flags for the last twenty plus years have had fifty stars not

thirteen. To change the flag to fifty stars several lines of code will need to be added as follows.

```
69 IF R<>0 THEN 101
101 FOR A=36 TO 220 STEP 18:' 50 Stars
102 FOR B=1 TO 9
103 IF (B/2<>B\2 AND A/36<>A\36) OR
(B/2=B\2 AND A/36=A\36) THEN 105
104 PUT(A+10,B*10+13),A#,XOR
105 NEXT B,A:R=0
:
```

Also lines 100 and 110 must be changed as follows:

```
100 NEXT I:GOTO 110
:
110 A$=INKEY$:IF A$="" THEN 68 ELSE CLS:LIST
```

When making a change to a line of a code you may retype the line and hope you have not introduced any new errors or unwanted changes. A better way is to edit the line.

To edit a line, first list the line, then, using the arrow keys, move the cursor to the location that is to be changed. Now type the new characters on top of the old characters pressing the RETURN key when finished. Use the I-CHR (Insert Character) key to go in and out of the insert character mode. Also the D-CHR (Delete Character) key allows deleting of one character at a time. No changes are recorded into memory until the RETURN key has been pressed. *NOTE: If there are any unwanted characters to the right of the last character on the line, be sure that they are removed, before pressing the RETURN key, as they will become part of the line and may cause real problems later on.*

Try running the program after the above lines have been added and lines 100 and 110 changed.

The additional lines 101 to 105 add the fifty stars to the flag which are done by using two FOR-NEXT loops and a couple of formulas to determine where the stars are to be drawn. The value of the variable 'R' is set to zero as it is to be used in the check in line 69.

Here is the complete listing as modified by adding the above lines and making the required changes.

```
10 CLS:' FLAG.BAS Version 04.04.83 GK:
:
20 C6=6:PSET(26,3),C6:' Create Star
30 DRAW"F3R3G3D3H3G3U3H3R3E3"
40 PAINT(26,4),C6
50 DIM A$(100):GET(20,3)-(32,12),A#
:
60 CLS:LOCATE 25,16:PRINT"Press any key to stop"
:
61 CIRCLE(25,7),15,6:PAINT(30,9),6:' Top Of Pole
:
```

```

62 LINE(20,15)-(30,225),6,BF:' Pole
:
63 PSET(30,15),7:DRAW"M39,20D155M37,233":' Rope
:
64 FOR I=0 TO 12:CL=4:IF I/2<>I\2 THEN CL=7:' Red & White
65 LINE(40,I*12+20)-(600,176),CL,BF:NEXT I:' Stripes
:
68 LINE(40,20)-(250,115),1,BF:' Blue Field
:
69 IF R<>0 THEN 101
70 R=35:FOR I=1 TO 13:' 13 Stars
80 C=(I*(360/13))*3.14159/180
90 PUT(135+(INT(COS(C)*R))*2,63+INT(SIN(C)*R)),A#
100 NEXT I:GOTO 110
:
101 FOR A=36 TO 220 STEP 18:' 50 Stars
102 FOR B=1 TO 9
103 IF (B/2<>B\2 AND A/36<>A\36) OR
(B/2=B\2 AND A/36=A\36) THEN 105
104 PUT(A+10,B*10+13),A#,XOR
105 NEXT B,A:R=0
:
110 A$=INKEY$:IF A$="" THEN 70 ELSE CLS:LIST

```

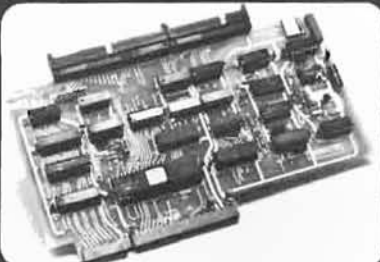
The American flag was an easy one to create, now put yourself to the real test and create your own state flag and send it to HUG. As flags are received I will make them available to those that are interested in them. Some of you may also try drawing your state on the Z-100. Rhode Island has already been done, whose next? Maybe we can come up with the best Name the States educational game.

This completes the introduction of most of the new graphic commands for Z-BASIC. The article for next month will explain some of the other new commands available under Z-BASIC. As the introduction of all new commands will be finished in the next couple of articles, take some time and write a letter to me in care of the Heath Users' Group with any suggestions for future articles on Z-BASIC commands.



1 CONTROLLER

FOR 8" & 5.25" DRIVES




Now be able to run standard 8" Shugart compatible drives and 5.25" drives (including the H37 type) in double and single density, automatically with one controller.

Your hard sector 5.25" disks can be reformatted and used as soft sector double density disks. The FDC-880H operates with or without the Heath hard sector controller.

NEW PRICE \$495

Includes controller board CP/M boot prom, I/O decoder prom, hardware/software manuals BIOS source listing. HSOD driver now available for \$40.00.

5-20 day delivery—pay by check, C.O.D., Visa, or M/C.



Contact:
C.D.R. Systems Inc.
7210 Clairemont Mesa Blvd.
San Diego, CA 92111
Tel. (619) 560-1272

Newline Software

P.O. Box 402 • Littleton, MA 01460
617-486-8535 (evenings and weekends)

SOFTWARE FOR HEATH/ZENITH COMPUTERS

**SATISFACTION GUARANTEED...
OR YOUR MONEY BACK!***

NEW VERSION...TEXT PROCESSOR VERSION 4.1

Text Processor (Txtpro) is a full-featured, screen-oriented text processor designed especially for the H8/H19, H/Z89, and H/Z-100 computers. Over 2,000 satisfied users have proven that TxtPro is an extraordinary value. Many users compare TxtPro to other word processors costing hundreds of dollars! TxtPro is ideal for writing reports, articles, letters, documents, programs, or any other text you may need to create or edit on your computer.

We are so confident that you will be totally satisfied with the quality and value of TxtPro that, for a limited time*, we will give you your money back if, for any reason, you are not delighted with the power and ease of operation of TxtPro! What other software producer would dare to make this offer?

Features include: word-wrap, margin settings, left and right justification, selectable hyphenation, files larger than memory, complete cursor control, macro-command buffer, global or range repeat, search with replace or delete, OOPS key (undoes your last command), lines up to 255 characters, insert control characters, split and merge line, and many, many more.

The HDOS and CP/M 2.x versions support the Super-19/ FONT-19 ROM set character roms for the H19 and H/Z89 terminals.

For HDOS, CP/M 2.x, Z-DOS, CP/M-85, **\$59.95 each**
Special: CP/M-85 and Z-DOS versions,
same order **\$99.95 both**

*money-back offer good through July 31, 1983.

FREE CATALOG.....WRITE FOR YOURS TODAY!.....FREE CATALOG

H/Z-37 Controller on Standard Eight Inch Drives

E. B. Blayer
1008 West Bay Ave.
Barnegat, NJ 08005

Sooner or later the need for more storage overcomes all keyboard pounders. At the present time there are various solutions to this storage problem offered by the many vendors and suppliers of accessories for the Heath/Zenith line.

This article describes a way to add 8" drives using the lowest priced double-density controller.

Before I outline the details, however, let me explain how I came to choose this particular solution. You may have come to the same conclusions and it may, therefore, meet your needs as it is certainly meeting mine.

Background

My first consideration was cost. A quick survey of the available equipment showed the 8" drives to be the most cost effective as a mass storage device. The H/Z-47 seemed to be an excellent unit but it used drives too sophisticated for my needs and was too costly for my budget. I had noticed that standard Shugart compatible drives could be found from four to six hundred dollars; surplus drives from commercial users are showing up at computer flea markets in quantity and at affordable prices.

My first move was to purchase a Shugart 850 double-density double-sided drive. I then began to think about designing a suitable controller.

A survey of all the available controller boards for the H/Z-89 shows a common factor: the same controller chip, Western Digital's WD1797, was used in all of them. A call to Western Digital brought all of the technical data for their various disk controller chips. After reviewing their material, I realized the project would take considerable time and effort. A more practical approach appeared to be purchasing, rather than building a controller.

The hardware changes were the simplest part of the task. The technical data on the control chip showed that there are only minor changes in the circuitry for the 5" drives to that for the 8" drives. Basically, the changes are as follows:

CIRCUIT	5 IN.	8 IN.
WD1797 CLOCK	1 MHZ	2 MHZ
VCO IN-DATA SEPARATOR	2 MHZ	4 MHZ
PLL SERIES R/C FILTER	.68 mfd & 68 Ohms	.33mfd & 33 Ohms

Calls to several manufacturers of controller cards for the H/Z-89 soliciting their specifications yielded little in technical documentation. I knew that on any controller I bought, the changes would be fairly simple. A visit to my local Heath Electronic Center for a look at the H/Z-37 schematic showed that the circuitry for the WD1797 was exactly as the Western Digital manual recommended and in addition all the interrupt controller circuit for the Z-80 was expanded, the entire circuit had been moved to the H/Z-37 card. The decision about which controller to buy for modification wasn't hard after this.

The use of a Heath product probably carried the most weight after I considered what had to be done. Rather simple modifications to the hardware would make the card (originally intended for 5" drives) operate for 8" drives. The modifications in this article are intended only for use with 8" drives. Using dual 5" and 8" is just not possible with the simple changes listed here.

Results

I am currently running a pair of Shugart 850 drives. The capacity of a double-sided double-density 8" drive is 4000 sectors or 1 megabyte; thus, I have 2 megabytes on line! A single-sided drive offers 2000 sectors.

Several members of the South Jersey Heath Users' Group (SHUG) have upgraded to 4MHz and are running various combinations of 8" drives.

The drives are daisy-chained. Four drives off one connector have worked perfectly. Only three drive-select lines are used (DS0- DS2). If more than 3 drives are desired, a binary select scheme has to be implemented. DS3 is changed to a WG43 signal to lower the drive write current on the inner tracks.

Requirements for Getting Started

The first requirement is a 4 MHz CPU. There have been a number of articles about how this modification is done on HBBS and in REMark.

The other requirements are made to the H/Z-37 controller board. These modifications will support only one system: 8" drives, single and double density. Once the modifications are made, 5" drives on the H/Z-37 are no longer possible, though your present H-17 board will still be useable.

An adapter board must be made to change the 34 pin cable from the H/Z-37 to the 50 pin cable used by 8" drives.

Hardware Modifications

Read Data Conditioning

The first modification that needs to be completed involves a change to the raw data conditioning monostable. This is used to set the width of the read data pulses from the drive to the width required by the WD1797 read input. The time needed is 150ns plus or minus 50ns. To accomplish this, refer to the H/Z-37 schematic at U-14A. Capacitor C27 has to be changed to 15pfd.

WD1797 Clock

For 8" drives the controller chip requires a 2MHz clock. The clock generator is U-9 which is a 16 MHz oscillator module. This drives U-10 which is a Divide By 16 to produce the necessary 1MHz for the 5" drives. To obtain 2MHz required by the 8" drives, cut the path at U-10, pin 11 and install a jumper from the cut path to U-10, pin 12. At pin 12 is the 2MHz output of the divider.

VCO Frequency

The voltage controlled oscillator is U-17. This normally operates at

2MHz; it must be changed to 4MHz. The frequency is determined by capacitor C32 which is 82pfd and must be changed to a value of 39pfd.

PLL Filter

The PLL filter is a network used to reduce ripple on the control voltage to the VCO and consists of components C29 (.68mfd) and R68 (68 ohms). C29 is changed to 0.33mfd capacitor and R68 is changed to a 33 ohms resistor.

TG43

This signal is used on some drives to lower the write current when the density on the media's inner tracks is too high to write reliably with the normal value. This signal is output by the controller chip (WD1797) when track 43 or higher is addressed by the controller. This signal is not available on the H/Z-37 but is easily added. Another output line is needed from the card so the DS3 line was selected to use existing output buffering.

The modifications for the TG43 are straightforward:

1. The path at U-11 pin 19 is cut to disable use of the DS3.
2. A jumper is added from U-12 pin 29 to U-16 pin 9. This ties the WG43 pins together.
3. A jumper is added from U-12 pin 29 to U-15 pin 10. This ties the new WG43 signal to the input of an unused drive buffer.
4. A jumper is added from U-15 pin 9 to U-22 pin 1. This takes the output of the previously unused drive buffer to drive the DS3 line.

Note: The schematic shows U-15's unused drive input on the output side of the chip lines and the unused output on the input side.

RFI Filters

The RFI filters on the output lines must be removed for the read and write data lines. The constants used attenuate double density signals. The easiest way to do this is to cut capacitors C39 and C44 from the board. They are 470pfd disk. The series inductors L13 and L18 must be removed and replaced with jumpers. The inductors are the ferrite beads with the wire run through twice. This will take care of the top connector P3. If the bottom connector P4 is to be used, then parts C53, C58, L27, and L32 must also be changed as above.

1. L13 and L18 are shorted or removed and jumpered.
2. C39 and C44 are removed.

This prevents the RFI filter from degrading read/write data.

This completes the hardware modifications on the H/Z-37 board. The modified board now needs to be aligned.

Alignment of Controller

At this point the controller card alignment should be done. Refer to the section in the H/Z-37 operation manual entitled "Recalibration" (p.33).

The following changes need to be made in the section entitled "VCO CENTER FREQUENCY ADJUSTMENT":

1. With a frequency counter the frequency will now be 3950 to 4050KHz. (See p. 34 in the H/Z-37 Manual.)
2. With an oscilloscope the period will be 247ns to 253ns. (See p.36 in the H/Z-37 Manual.)

Precompensation

("Precomposition" in the Manual; p.36)

1. Change to comply with drive manufacturer's recommendations.

I found for the drives tested, minimum precompensation worked best.

All other steps in the recalibration section should be followed as written.

If all of these changes have been accomplished, then the project is almost completed. Next is to describe the adapter card for pinout conversion.

Adapter Card

Next an adapter card is fabricated. A perf board like the Radio Shack (276-162) with 0.100 inch spaced holes can be used. Two headers for the ribbon connectors are also needed. The 34 pin and the 50 pin headers are then mounted on the perf board and all the ground pins are connected together first. Next the signal leads must be cross wired to suit your particular drive. The following list has been prepared for use with Shugart 800-1, 801, 850, and 851 type drives.

Note: Be sure to check orientation of the cable to this adapter card before wiring and verify pin connections at the controller card with an ohmmeter or you may find a reverse wired connector.

34 pin connector	H-37 Signal	8" Drive signal	50-pin
6	DS3 (NEW WG43)	WRITE CURRENT SW	2
8	INDEX	*	20
10	DS0	DS1	26
12	DS1	DS2	28
14	DS2	DS3	30
16	MOTOR	To external Relay (see MOTOR below)	No Connect
18	DIR	DIRECTION SELECT	34
20	STEP	STEP	36
22	WR DATA	WRITE DATA	38
24	WR GATE	WRITE GATE	40
26	TK00	TRACK 0	42
28	WRPT	WRITE PROTECT	44
30	READ DATA	READ DATA	46
32	SIDE SELECT	SIDE SELECT	14

Motor

The driver software has a timed motor routine for those who prefer not to run the drive motors continuously. This allows the motor lead +5V from the controller to operate a solid state relay or equivalent for switching the 115V to the drive motors. The motor(s) will then switch on as required and then the drive motor after a suitable delay (say, a 30 second period of inactivity) will switch off.

Software

My primary operating mode is HDOS so this was the first driver to create for the 8" system. A copy of the H/Z-37 device driver was disassembled. The analysis of the code took about four weeks, rewrite about two more, debugging took another 3 weeks.

The resultant driver has the following characteristics:

1. Set option for number of drives (1-7), allowing minimum use of GRT space taken from HDOS user memory .
 - Line select for up to 3 drives.
 - Binary select for 4-7 drives on lines DS0-2.
2. Separate set option of step time for each drive.
 - Step times are 3,6,10,and 15 milliseconds.
3. Display of presently set options for all enabled devices.
4. Use of the driver with HDOS Init.
 - Will give selection of density and number of sides.
 - Sector skew is variable. Track 0 has no skew allowing the block read boot code in MTR-90 to load in less than a second.
 - All other tracks have the sector sequence skewed 5 sectors. This skew was optimized by trial, MBASIC loads in less than 3 seconds.

Media8 - A media check program that writes a test pattern over an entire disk and then reads back all sectors verifying the data and reporting bad sector groups.

DUPDK8 - A disk duplicating program to format and copy is currently being tested.

Shugart 800-1 & 801

Internal Selectable Jumper or option		Comment
X	X	
DS	X	
HL	0	
A	X	
B	X	
800-801	X	WHICHEVER TYPE OF DRIVE
T2	X	
T1	0	
DC	X	
D	0	
C	0	
I	0	
R	X	
S	0	
DS1-4	X	WHICHEVER DR. SELECTED
T 3-6	ALL	TERMINATE LAST IN CHAIN ONLY
Z	X	

Automount - A utility intended for use with any DVD that has mountable units. All command line specified devices will have any units with disks in the drives mounted .

CPM

BIOS.SYS modifications are currently being worked out. As CPM is not my favorite system, its lowest on the list of things to do!

Configuring Drives

The following tables can be used as a guide for configuring a specific drive to the system. The information here should be sufficient to allow an "oddball" drive to be properly configured.

X=connected or jumper in; 0=no connection

Note: Persons interested in making these modifications should contact Ed Blayer, 1008 West Bay Ave., Barnegat, NJ 08005, for software listings, etc. Ed will also make modifications to your H/Z-37 board upon request. Contact Ed concerning price and delivery.

Shugart 850 & 851

Internal Selectable Jumper or option		Comment
dip shunt		
R	X	
I	X	
S	X	
HL	0	HEAD LOAD ON SELECT
A	X	
B	X	
X	X	
Z	X	
JUMPED OPTIONS		
DS1,DS2,DS3,DS4 :ONE ONLY FOR SY0:,1,2		
RP9	X	LAST DRIVE IN CHAIN ONLY !
850-851	X	850 POSITION
DS	X	STEPPER PWR ON SELECT
S2	X	STD. SIDE SELECT
TS-FS	X	JUMPER FS POSITION
RS-RM	X	JUMPER RS POSITION
IT	X	
-12,-5	X	POSITION TO SUIT PWR SUPPLY!
ALL OTHER OPTION PLUG POSITIONS OPEN		

Note: For more than 3 drives a binary select option must be fitted to the drives.



HUG NEW PRODUCTS



885-1123 (HDOS)
885-1229[-37] (CP/M)

XMET Robot and ET-3400 Cross Assembler \$20.00

Introduction: XMET is a 6800 cross assembler that includes mnemonic representations of the ET-18 Robot's "Robot Language" codes, to make writing programs for the HERO 1 robot or the ET-3400 easier. It is provided in both HDOS and CP/M versions, and is supplied with printed documentation.

A detailed look at the cross assembler is described in this issue under the article "Teaching HERO a Thing or Two" by Pat Swayne.

Requirements: CP/M or HDOS and at least 32k of memory.

HDOS: This version requires the HDOS operating system, version 2.0, on an H8/H17/H19 or H/Z89 and one disk drive.

CP/M: This version requires the CP/M operating system, version 2.2 or later on an H8/H17/H19, H/Z89 or Z90 with one disk drive.

The source code files are included.

The following files are contained on the XMET disks.

885-1123	885-1229
README.DOC	README.DOC
XMET.ABS	XMET.COM
XMET.ASM	XMET.ASM
ABSHEX.ABS	HEXMIK.COM
ABSHEX.ASM	HEXMIK.ASM
HEXMIK.ABS	EZHEX.COM
HEXMIK.ASM	EZHEX.ASM
EZHEX.ABS	RSAMPLE.XET
EZHEX.ASM	ESAMPLE.XET
RSAMPLE.XET	
ESAMPLE.XET	

Author: Patrick Swayne

XMET.ABS, XMET.COM — This is the XMET cross assembler. For a description of its capabilities, see the article "Teaching HERO a Thing or Two" in this issue.

XMET.ASM — The source code for XMET.

ABSHEX.ABS — This program is supplied with the HDOS version to convert an assembled .ABS file to Intel HEX format.

ABSHEX.ASM — The source for ABSHEX.

HEXMIK.ABS, HEXMIK.COM — This program converts Intel HEX files to Motorola MIKBUG(tm) format files.

HEXMIK.ASM — The source for HEXMIK

EZHEX.ABS, EZHEX.COM — This program converts Intel HEX files to a format that is easy to read (for manual entry of programs into the robot or ET-3400).

EZHEX.ASM — The source for EZHEX.

RSAMPLE.XET — A sample robot program in XMET source format.

ESAMPLE.XET — A sample ET-3400 program in XMET source format.

Comments: What can you say!! This program is a must for any HERO/Heath computer user.

Software Announcements:

Purchasers of HUG P/N 885-1095:

If you would like the updated HSY.DVD and supporting programs which come on the new HUG P/N 885-1121 Hard Sector Support Package, you may send the original P/N 885-1095, a blank disk and \$15 to HUG, Hilltop Rd., St. Joseph, MI 49085, in care of Nancy Strunk. Nancy will send you the HUG P/N 885-1121 package.

Users of CP/M-85 on the H/Z100:

HUG will be releasing HDOS disks, five per month, on soft-sectored format diskettes. This will allow user of CP/M-85 to utilize Pat Swayne's HRUN, HUG P/N 885-1223[-37] (an HDOS emulator). The utility HTOC (on P/N 885-1223[-37]) is required to "transfer" the programs from the HDOS disk format to CP/M-85 disk format.

The HDOS soft-sectored part numbers are treated the same as the CP/M counterparts; simply add a -37 to the part number to specify that soft-sectored is desired, e.g. P/N 885-1062-37.

The following five HDOS disks are now available in soft-sectored format:

	885-1067[-37]
885-1062[-37]	885-1029[-37]
885-1086[-37]	885-1060[-37]

Remember the -37 must be included to indicate soft-sectored.

HUG Price List

The following HUG Price List contains a list of all products not included in the HUG Software Catalog. For a detailed abstract of these products refer to the issue of REMark specified.

Part Number	Description of Product	Selling Price	REMark Issue
HDOS			
885-1121	Hard Sected Support Package	\$ 30.00	37
885-1122	MicroNET Connection	\$ 16.00	37
CP/M			
885-1211 [-37]	Sea Battle	\$ 20.00	36
885-1222 [-37]	Adventure	\$ 10.00	36
885-1223 [-37]	HRUN HDOS Emulator	\$ 40.00	37
885-1224 [-37]	MicroNET Connection	\$ 16.00	37
885-1225 [-37]	Disk Dump and Edit Utility (DDEU)	\$ 30.00	38
885-1226 [-37]	CP/M Utilities by PS:	\$ 20.00	38
885-1227 [-37]	CP/M Cassio Graphic Games	\$ 20.00	38
885-1228 [-37]	CP/M Fast Action Games	\$ 20.00	39
885-3003 [-37]	ZTERM Modem Package	\$ 20.00	36
885-8012 [-37]	Modem Appl. Effector (MAPLE)	\$ 35.00	36
ZDOS			
885-3004-37	ZBASIC Graphic Games Disk	\$ 20.00	37
885-3005-37	ZDOS ETCHDUMP	\$ 20.00	39
MISCELLANEOUS			
885-0004	HUG 3-Ring Binder	\$ 5.75	
885-4001	REMark VOLUME 1, issues 1-13	\$ 20.00	
885-4002	REMark VOLUME 2, issues 14-23	\$ 20.00	
885-4003	REMark VOLUME 3, issues 24-35	\$ 20.00	

NOTE: The [-37] means the product is available in hard-sectored or soft-sectored. Remember, when ordering the soft-sectored format, you must include the "-37" after the part number; e.g. 885-1223-37.

Raiders Bug

The RAIDERS game from HUG disk 885-1114 has a "bug" in it that causes the keypad on your H19 to be taken out of the shifted mode so that you cannot control the direction of your "space ship" after the first game. To correct the problem, enter the following patch using the program PATCH.ABS (what you type is shown in **bold print**).

```

>PATCH

PATCH Issue #50.06.00.

File Name? SY1:RAIDERS.ABS

Address? 77231
077231 = 033/0
077232 = 365/200
077233 = 315/^D          (Control-D)
Address? 77326
077326 = 305/372
077327 = 257/^D
Address? ^D

PATCH Issue #50.06.00.

File Name? ^D
    
```

The above example assumes that PATCH.ABS is on SY0: and RAIDERS.ABS is on SY1:. If the old data (the numbers before the slashes) does not match what is shown above, do not make the patch. Our stock is being updated with a correct version of the game, which has also been modified to allow operation on the HA-89-3 color board from New Orleans General Data Services for H89 computers. See REMark issue #36 (Cross Reference issue), page 50, for details on the HA-89-3.

PILOT Correction

In the PILOT program that appeared in Issue 39, on page 31, line 1140, all of the variables A\$ should be changed to B\$. This concerns line 1140 only.

About the PILOT Author: Kurt Albrecht is a junior in High School and has been programming since eighth grade. He hopes to be entering Drexel University soon to major in Computer Science. Kurt feels the computer is the students best friend and has a couple years worth of reports and term papers on file. He is an assistant in the computer science lab at his school and also writes software for profit.



BE A HAPPY HUGGER.

Get to know Floppy Disk Services.

We are contracted dealers for Siemens, Shugart and Tandon disk drives. You may do a double take when you see our prices, but there's no catch—we simply buy in large quantities and sell at reasonable prices.

We've sold thousands of drives to our Heath/Zenith friends. And with our good service and fair guarantee, we keep our friends friendly.

Check our prices on these Heath/Zenith compatible drives.

FDD-100-5b 'flippy' (Just like your Heath, but 'flippy')	\$215.00
FDD-221-5 DS/DD 96tpi (80 track) same as TM-100-4	330.00
FDD-200-5 same as 100-5 but double sided (2 heads)	250.00
FDD-111-5 5 milisecond SS/DD 48tpi same as TM-100-1	210.00
FDD-211-5 5 milisecond DS/DD same as TM-100-2	265.00
FDD-100-8d5 SS/DD 8 inch 3 milisecond step!	275.00
FDD-200-8p5 DS/DD 8 inch 3 milisecond step!	395.00

Look for our Half Height Drive Specials Coming Soon.

PAYMENT POLICY We accept Mastercard, Visa, personal checks & M.O. We reserve the right to wait 10 working days for personal checks to clear your bank. All shipping standard UPS rates unless otherwise requested. New Jersey residents *must* add 5% sales tax.

Due to production deadlines, prices in this ad could be as old as 2 months. If in doubt, call!

Prices and specs subject to change without notice.

Need an enclosure for 5¼ drives?

Single vertical case w/ps A&T styled to match Heath color	\$ 50.00
Dual vertical case with power supply A&T	75.00
Dual horizontal case with dual floppy supply A&T	125.00

... or an enclosure for 8 inch drives?

Single 8 inch horizontal case w/ps A&T for 1 standard or 2 half height	Call
Dual horizontal case with commercial grade supply A&T	\$275.00
Dual horizontal case metal only, w/power switch, and hardware	115.00

Magnolia 8" Controller

Magnolia Microsystems controller for the Heath H88, 89, 90 enables you to use any combination of 8 or 5¼ single or double sided, single or double density—up to 4 of each size drive! You even get CP/M in 8 and 5¼ format serialized for you. This controller is available at a special price of \$525.00.

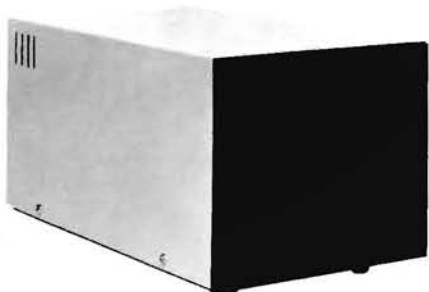
Our pledge to you . . .

All Floppy Disk products are brand new and 100% warranted. If you are unhappy with our systems or components for any reason, we will refund your money promptly. There are two requirements: 1. the equipment must be in as good shape as you received it; and 2. you must call or write for a return authorization. This offer is good for 30 days, beginning with your invoice date. Feel free to call us with any questions.

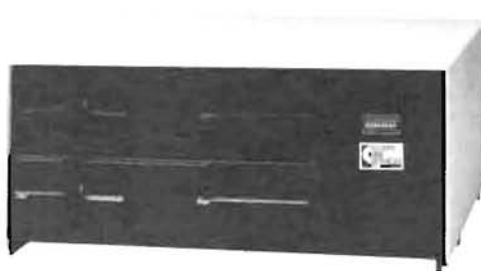
Repairs, too.

We repair drives of all types. Call for details.

**We make Huggers happy.
Call today! (609) 799-4440**



**Dual Heath Add-on
FDD-100-5**



**Dual Half Height
or Single Full Size 8"**



741 Alexander Rd.
Princeton, NJ 08540
(609) 799-4440

CP/M and H88-89-90 are registered trademarks of Digital Research and Heath Co., respectively.



Computer Aided Instruction, Part 2

Walt Gillespie
REMark Editor

In Part 1 you were introduced to CAI (Computer Aided Instruction) and given preliminary help in choosing a subject. Now, we will begin the Design Task.

The design task is a very important part of any CAI project, because, it is at this point that decisions are made that will effect the overall visual appeal of your presentation. "The first step in designing instructional computing materials is the organization of the material. Lay out your material by frames on paper first. A frame is one display on the screen.", so says the MECC (Minnesota Educational Computing Consortium). You should use some kind of a screen layout form that is suited to the particular terminal being used, or graph paper marked to your needs.

Plan your "frames" so that the student can know where to expect certain types of materials to appear, that is, always place the questions in one particular part of the screen, etc. If you make it a practice to move similar information to different parts of the screen from frame to frame the student will become confused if he/she has to search the screen for a certain piece of information. Be consistent with your frame layout is the first rule.

Limit the use of graphics and audio. Flashing animation or the beeping of the computer can be distracting not only to the operator but also to others in the room. When graphics are inserted sparingly they become a novelty, with too much use they distract and become a pain. Audio should only be used to alert the instructor that a student is having problems after the student has been given maybe three chances but still can't find the correct answer, or, in the case of home study, to alert the student as to the correct answer, etc.

Use such items as reverse video and light graphics, such as borders, to enhance your frames. Reverse video could possibly always be used to prompt for input, thus the students eyes will be drawn to that point, but if too much is used it will be hard to focus the eyes. If you always enclose your questions within a border, again, the student will automatically know its meaning and the eyes will be drawn to it. Placing both these items in the same general location on all frames, such as questions in the upper left corner, will reinforce the students learning and comprehension when moving from frame to frame.

Some helpful notes from the MSCC are:

"Move from LEFT to RIGHT on the screen to reinforce eye- training."

"Avoid over use of a FLASHING mode. Many times this tends to put pressure on the user to 'hurry up'."

"DO NOT allow SCROLLING of text on the screen."

Be careful of the overuse of any special screen techniques, Reverse Video, Graphics, etc., as a reinforcement, this can become boring. The first time through these items might be a novelty but if the student has reason to repeat the materials many times over "extra added attractions" can become tedious after a while.

"A VERY EFFECTIVE reinforcement technique is not simply to say 'OK', but to give the student some additional information concerning the question. This allows extensions of what is being discussed

while providing needed re-reinforcement."

A provision must be made for the movement between frames, here again be consistent. Arguments can be made as to which is the best way to accomplish this movement, the most common being the 'Space Bar' and the 'Return Key'. Another method of frame movement is the 'Type Any Key' format, this can be confusing if mixed with frames using multiple-choice questions. Remember, if you use an automatic function such as 'INKEY\$', the chance of multiple entries is present. You should allow the student to back up a frame should a multiple entry cause them to jump ahead an extra frame or two.

You might find it best to use an answer type mode to move the frames ahead, that way you limit the possible selection on input and can return automatically if the incorrect answer is given.

Avoid timing loops. "Nothing is more frustrating to a user than having to wait for a timing loop to expire when ready to go on, or to have it expire before the text has been read."

The user should not have to guess how to enter answers. Give examples if necessary, such as double letters or commas between numbers if needed. Be consistent in your method of asking questions from frame to frame, don't use YES or NO in one frame and Y or N in another. Never rely on abbreviations, spell out all words. Try to ask your questions while information necessary is on the screen. If options are needed on more than one frame repeat them again in the same position on the screen, if necessary use a 'HELP' option to refresh the students memory.

At times the multiple choice method will provide the best method of answer selection, use either number or letter designation and allow for upper and lower case input if necessary. When a direct answer method is called for, use a 'Key Word' match so that if you request, "WHAT IS OUR PLANET CALLED?"; a student's response of, "ITS CALLED EARTH"; would be counted correct. Set a maximum number of tries on any question, so that after, say, three incorrect answers you give the correct answer and move on to the next question. You might retry the missed questions later on to see if the student has retained the information.

If you have reason to present multiple choice type questions, use a "menu" type format, again, be consistent. Don't use alpha letters (A,B,C,etc.) in one frame and numerals (1,2,3,etc.) in another to designate possible choices. The menu format is one of the easiest for the beginner to use. Menus are best used for program selection and execution. Allow for single key response to such questions as; "Run This Program Again?", "Return to Main Menu?" or "End Program?", thus, error trapping is made easy. As for "hard copy", remember not all installations have this available. Your project should allow this as an option only. If your program is "hard copy" dependent or dependent on one particular kind of printer you are limiting the number of possible end users.

Throughout this section I have stressed the importance on consistency in frame layout, question & answer positions, use of multiple choice and menus. A CAI project can only be effective when this consistency is coupled with good programming practices and good

error trapping. But, error trapping can be taken just so far. If a person sets out to "crash" a program they will generally find a way, just don't make it too easy.

Choosing a Language.

I won't try to expound the merits of any particular language, we don't have the room here to present all the pros and cons, nor do I want to start any arguments as to which is best. What I do want to do is help you make a choice based on which language is both easy for you to use and will fill your projects needs. MBASIC (interpreter) is probably the most widely used language and is generally an easy language to program in. You might be more familiar with another language but unless it can be compiled into a .COM or .ABS file for direct execution you might not find it acceptable to the general user. The speed of your particular program is not necessarily an important factor, unless there is a lot of machine processing, the execution time will be set by the operator's response time. The language chosen should be one that is easily loaded and run. Your program should include all necessary subroutines and should not require the purchase of any additional languages, compilers or device drivers. Your program should be able to run on any standard configuration of Heath/Zenith Equipment, example: a program written in MBASIC for CP/M using standard Heath graphics would normally run on an H8, H/Z-89 and H/Z-100. But programs written for FORTRAN on the H/Z-89 under HDOS might not run on any other configuration thus limiting its use. Forget your personal prejudices concerning a language and choose one that will give you the most general acceptability, then write your program to best utilize that language.

I hope that this material is helping you in preparing your CAI project. In the next installment I will address Part 4, Keeping a Clean House and Part 5, Fine Tuning. ✖



BEFORE

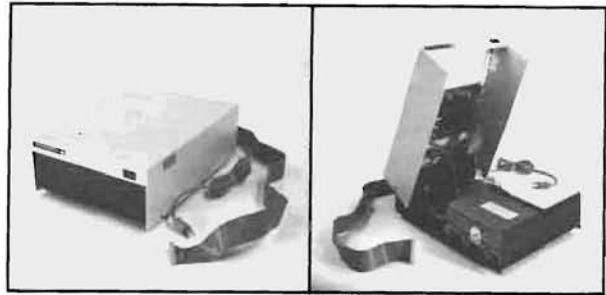


AFTER

Paper Holder for H-25 Printer holds 40 lbs. 11 1/2 x 14 inch paper. If enough are interested, can make it for Diablo Printer and stand. For prices and information write to:

KINGS MFG. COMPANY

1621 2nd Avenue North
Estherville, Iowa 51335



NOW 12 MEGABYTE (CDR-10M) \$3195

and 6 MEGABYTE (CDR-5M) \$2495

WINCHESTER SYSTEM For the Heath/Zenith Computer

Systems complete with software case, power supply & signal cable.

Runs with CP/M, on the H/Z89, Z90 & H8 (with Z80 card).

- Switching power supply
- Expansion for backup installations
- Auto attach BIOS
- Hard disk utilities
- Formatting program
- 1 year parts & workmanship warranty

CP/M is a trademark of Digital Research. Heath, H8, H89 are trademarks of Heath Corporation. Zenith, Z89, Z90 are trademarks of Zenith Data Systems.

5-20 day delivery-pay by check, C.O.D., Visa, or M/C.

Contact:



C. D. R. Systems Inc.
7210 Clairemont Mesa Blvd.
San Diego, CA 92111
Tel. (619) 560-1272

PRODUCTIVITY TOOLS for Heath CP/M users

ZSpool-Plus \$59.95

eliminates your printer bottleneck. It's the only Mainframe-quality queue manager and spooler for Z80-based Heath/Zenith systems. It will keep your printer busy while doing almost any other task. Extends your CP/M to improve type-ahead, reduce warm-boot time, improve keyboard operation, stop a runaway SUBMIT and much more.

ZSD-89 \$19.95

Z80 screen-dump program to print your H/Z89 display upon demand. Assign keys to print the screen, advance one line or a page. Its features are also available as CP/M calls.

Super-19 from Accusonics \$49.95

The H/Z19 ROM upgrade everybody is raving about. It makes ZSD-89 work like lightning. A must for operation at 19.2 KB. Includes time-of-day clock, more keyboard functions, light-pen support and most VT100 features. Permits use of extended character ROMs too.

Free shipping anywhere in the USA.

VISA and MasterCard orders call: (800) 343-7775



SYSTEMS, INC.
8 Crescent Street
Wellesley Hills, MA 02181
(617) 431-7870

Datetime

ZDOS Automatic Time Update

Frank T. Clark, M.S.C.S.
402 W. Ferry
Berrien Springs, MI 49103



Many users probably used HDOS at one time and developed an appreciation for the HDOS date function. It is a useful feature and I for one am glad to see it in ZDOS (MDOS). There is one nagging little problem. Even primitive HDOS had a way of remembering the date you gave it and would allow it to be selected as a default if you didn't want to bother to enter a new date. Just pressing RETURN at the HDOS DATE prompt was very common for me as I am sure it was for many.

ZDOS is just like HDOS in some ways in that it requests the date and time every time you boot up. It also allows you to accept a default by just pressing RETURN. The problem is that this default is always back to the release date of the BIOS and cannot be easily changed. I personally found this intolerable and felt that the whole idea was somewhat wasted. I decided that if ZDOS could not be any smarter than that, I would write a program to improve the situation.

Solution

The solution I decided upon was a program that could be run with the AUTOEXEC.BAT start-up and any time prior to powering down the computer that would keep the time updated the best way it could. It turned out to be relatively simple to get some kind of approximate date and time. Every file, when it is written to the disk, has the date and time recorded in it. A search through the directory could be done very quickly and the date of the newest file checked against the current time. If there was a file on the disk newer than the current time, the date and time were adjusted to match this time. After this was done and an empty file was created on the disk, ZDOS would then nicely record the date and time in the directory.

Operation

The DATETIME program is run by entering:

DATETIME [?:]

at the ZDOS prompt. The optional argument indicates which disk will have its directory scanned and the file DATETIME.TMP written to. If no argument is given, it defaults to the current disk. The DATETIME.TMP file has no data written in it so it takes no free disk space and uses only one directory entry of which there are usually plenty. The directory scan and file write take only a couple of seconds. The DATETIME program is itself a COM type file for minimum size and fastest operation.

The DATETIME program would normally be copied onto every bootable ZDOS disk. Along with this would be an AUTO-

Program listing

```

        title  datetime - set date and time to newest file
        page   ,132
;
; DATETIME.COM - set the date and time
;
;copyright March 20, 1983 by Frank T. Clark
;
;This program is designed to be run from an AUTOEXEC.BAT file. It
;will automatically scan the entire directory selecting the date
;from the newest file. If this is more recent than the current date
;it will set the date and time. In any event it will create a file
;DATETIME.TMP which of course will have the date and time just set.
;This program can also be run at any time without any deleterious
;effects. An argument can be given specifying the drive to be searched
;and written to.
;
;Suggested AUTOEXEC.BAT file contents:
;
;DATETIME A:
;DATETIME B:
;
        .xlist
        include defns.asm
        .list
pgmseg segment
        assume cs:pgmseg,ss:pgmseg,ds:pgmseg,es:nothing
        org 100h
;
;start with ???????.?? and search for first directory entry
;
start:
        cld
        mov     di,phd_fcb1+1
        mov     cx,11
        mov     al,'?'
        rep     stosb
        mov     dx,phd_fcb1
        mov     ah,dosf_srhfi
        int     dosi_func
;
;get the date and time from this and successive files
;
loop:
        mov     dx,ds:[phd_dioa+25]    ;fetch date bits

```

EXEC.BAT file to run the program at boot time. There are several different command sequences that could appear in the AUTO-EXEC.BAT file depending on how comprehensive the user wants to be. The simplest would be just 'DATETIME'. More than one command could be included, with one for each drive that will usually be active, so that all the drives will be checked and updated at boot time. The maximum AUTO-EXEC.BAT file would contain a DATETIME call for every drive that would normally be mounted. If a drive happened not to be mounted, it could easily be skipped. Then there would be a call to the normal DATE and TIME to allow new values to be entered, if desired, or returned for the default. A second DATETIME command for every drive would be necessary to insure that the cumulative greatest date is on all disks. This would appear as follows in the AUTO-EXEC.BAT file.

DATETIME A:

DATETIME B:

DATE

TIME

DATETIME A:

DATETIME B:

This procedure would automatically guarantee that every file written to the disk will have a date newer than other files already on the disk. The difference in the date will be a relatively accurate proportion, though not absolutely exact, unless the user always enters the correct time at bootup.

About The Author:

Frank is a software consultant for Zenith Data Systems in St. Joseph, Michigan. He is also vice-president of the Blossomland Heath Users' Group. His background includes Xerox Sigma systems APL, COBOL, FORTRAN, Meta-Assembler, BASIC, word processing, IBM VS/APL, Ohio Scientific OS65D, 6502 assembler, Commodore DOS, Heath/Zenith CP/M, MBASIC, COBOL-80, and CP/M MAC 8080 assembler. He graduated from Andrews University, Michigan in August 1979 with a B.S.C.S degree and in August 1982 with an M.S.C.S. degree. His interests center on anything even remotely related to computer software including music, graphics and education as well as reading, mostly computer books and science fiction.

If you wish to contact the author please write to the above address, do not telephone, and include a self addressed and stamped envelope.

```

mov     di,offset buffer
mov     cl,7
rol     dx,cl
mov     ax,dx
and     ax,01111111b
add     ax,1980
xchg   al,ah
stosw
mov     cl,4
rol     dx,cl
mov     ax,dx
and     ax,011111b
stosb
mov     cl,5
rol     dx,cl
mov     ax,dx
and     ax,0111111b
stosb
mov     dx,ds:[phd_dioa+23]    ;fetch time bits
mov     cl,5
rol     dx,cl
mov     ax,dx
and     ax,0111111b
stosb
mov     cl,6
rol     dx,cl
mov     ax,dx
and     ax,01111111b
stosb
mov     si,offset buffer2
mov     di,offset buffer
mov     cx,6
repe   cmpsb
jae    older
;
;remember the date of this most recent file
;
mov     si,offset buffer
mov     di,offset buffer2
mov     cx,6
rep   movsb
;
;ignore old file and search for the next
;
older:
mov     dx,phd_fcb1
mov     ah,dosf_srhnx    ;search the rest
int     dosi_func
cmp     al,0ffh
jne    loop
;
;is the current date greater?
;
mov     ah,dosf_gdate
int     dosi_func
xchg   ch,cl
mov     word ptr buffer,cx

```



```

xchg dh,d1
mov word ptr buffer+2,dx
mov ah,dosf_gtime
int dosi_func
xchg ch,c1
mov word ptr buffer+4,cx
mov si,offset buffer2
mov di,offset buffer
mov cx,6
repe cmpsb
jbe check
;
;set to the new date
;
mov cx,word ptr buffer2
xchg ch,c1
mov dx,word ptr buffer2+2
xchg dh,d1
mov ah,dosf_sdate
int dosi_func
mov cx,word ptr buffer2+4
xchg ch,c1
mov dx,word ptr buffer2+6
mov ah,dosf_stime
int dosi_func
;
;check for any other modifications?
;
check:
;
;create file datetime.tmp with current time
mov si,offset filename
mov di,phd_fcbl+1
mov cx,11
rep movsb
mov dx,phd_fcbl
mov ah,dosf_crfile
int dosi_func
mov dx,phd_fcbl
mov ah,dosf_clfile
int dosi_func
stop:
int dosi_term
buffer db 6 dup(0)
buffer2 db 6 dup(0)
db 99,59 ;jump to next minute immediately
filename db 'datetime.tmp'
pgmseg ends
end start

```

The following commands will create the DATETIME program after it has been entered into the file DATETIME.ASM using EDLIN.

```

masm DATETIME;
link DATETIME;
erase DATETIME.obj
exe2bin DATETIME.exe.com
erase DATETIME.exe

```



SJULSTAD ENGINEERING

BRAND NEW 256 K RAM — \$699.00 (Software Included)

Expand your computer memory beyond its present 64 K RAM limit. Inexpensive, easily installed, and completely compatible with Heath/Zenith 88, 89, and 90 Microcomputers. Nothing on your present system is wasted or thrown away as with other 256 K RAM's. Write for more information on these products:

256 K RAM	\$699.00
128 K RAM	\$469.00

Both Memory Expansions complete with Instructions and Software.

REMOTE VIDEO OUTPUT — \$59.95

Provides NTSC Industry standard composite video to any monitor equipment including:

- Color or Black/White Television Monitors
 - Large Screen Monitors
 - Beta/VHS Recording Equipment
- Especially suited for Educational and Business Applications!*

16 K ADD-ON RAM — \$59.95

Increase your present 48 K system to its full capacity of 64 K. Required in order to run D-base II.

DISCOUNTS AVAILABLE FOR LARGER ORDERS

For more information write or phone:

SJULSTAD ENGINEERING
503 East Fremont • Northfield, MN 55057
(507) 663-3422

All boards are completely assembled and tested and are guaranteed for ninety days to be free from all defects.



We repair and service most microcomputers.



HOME FINANCE SYSTEM VERSION 2

—An extensive Home Finance System that keeps track of checking, asset accounts (cash, savings, IRAs, CDs), and regular bill payments. Let your printer write your checks for you on **any** business-sized check (design your own check format).

—Checks have user defined codes and a separate flag for tax deductible items.

—Many reports, including listing all checks, or checks by codes or tax flag.

—System consists of 100 page users manual with 5 program disks (5-1/4") and a sample data disk.

Hardware: H8/H19 or H-Z89/90 with 64K RAM and two disk drives. Printer strongly recommended (any Heath[®], Zenith[®] or other printer).

Software: HDOS 2.0 and Microsoft MBASIC 4.82 for HDOS.

—Complete system: \$89† (specify hard-soft sector 5-1/4", or 8"). Manual alone \$21†.

Master Card/Visa accepted, please include your phone number.

Jay H. Gold, M.D.
Jay Gold Software
Box 2024, Des Moines, IA 50310
(515) 279-9821

†Prices include shipping



A Review of RT-11 Version 5

Ed Judge
Fairbrother Assoc
Box 685
Northampton, MA 01061

Editor's note: Fairbrother Associates have become an excellent source of information and software for H11 users. They have public domain software and other software available that cannot be sold through HUG and other Heath sources. See the last paragraph of this article for details on getting a list of the software they have available.

RT-11 Version 5

The following is an excerpt from a report included in the latest Mini-Tasker from the New Mexico DECUS meeting held in Albuquerque on Sept 30 - Oct 1, 1982, written by Rally Barnard, which included highlights of new RT11 V5. The code for V5 was frozen on Sept 30, 1982, so the comments presented here represent the "most likely" configuration for V5.

1. LD - The Logical Device Handler is essentially a supported implementation of XD or SD, etc. It is a runnable handler implementation (issuing the MOUNT DVn:FILE LN: command runs the attaching program). It allows logical to logical assignments (ASS DK: DEV:). The SHOW SUBSETS command will indicate logical assignments; mounts are preserved over boots; the handler doesn't have to be loaded - it can be FETCHED, and will fetch the target physical device. [I don't know how many assignments it will take, but there is a new version called "VD" that allows you to rename the device to V*, where "*" is any letter. You only need one *DEF file for all of them. This may prove to be more flexible than the RT11 V5 version.]

2. IND - A RSX-style indirect command file processor; fully compatible with VMS and RXS V4.0. Supports DCL-style commands, parameters, etc. Thus, command files can now have variable arguments. [This is one of the most important additions.]

3. CCL/UCL - What we have all been waiting for (if we didn't have TSX+)! Issuing a command such as "RUNOFF File LP:" will cause a "RUN SY:" to be placed in front of it, thus giving you a simple command syntax. If there is no file of that name on SY:, the system will go into User Command Language (a SYSGEN option). This gets the unparsed command for doing with it what you want.

4. CO - Console Logger; another SYSGEN option. Uses a GT-style handler. Since it is big, it may be suspended to save space. Now output may be directed to the LP: in case it is necessary to try to decipher crashes.

(NOTE: Console Logger did not make V5 on the first pass but may make subsequent releases.)

5. BUP - Backup utility. It can backup a RL02 to TSV05 (the new 25 ips/100 ips streamer) in less than 1.5 min. It supports multiple volumes necessary to backup a big device onto a small one (i.e., floppies).

6. SL - Single Line Editor. A subset of KED. KED will be CHANGED slightly so that SL and KED are identical. It uses the arrow keys, PF1, PF2, BS, W; and is really slick. When you type "CPOY DEV:File.EXT/QUE DV2:*/LOG...", you can bomb back to the command with an up arrow, change only the offending characters and hit a return. The default for this feature is to have it in KMON only, but if you load SL, then ANY program which uses normal terminal input (ACCEPT, etc) can use it.

7. New Handlers - DU. For the RA80 (121 Mb), RC25 (50 Mb, the AZTEC), RD50, RX51 (the 5-1/4 INCH winnie and floppy). Now it is possible to have Gigabytes on your system under RT-11. For disks over 65k blocks, there are 65k partitions. This handler represents the new MSCP (mass storage Communications protocol). New devices need only emulate a MCSP device, and DU will handle it (i.e., no longer necessary to emulate a RL02). Legal MCSP devices are in the range 0-65k, while RT-11 devices are in the range 0-7. Thus unit translation is done (DU3:=UNIT 4047, etc).

8. VM - Virtual Memory handler. The handler is a cleanup and extension of the one we know and love. It has automatic mode-22, and XM support. You can set the base of VM, so that FORTRAN virtual arrays can live below VM:. Now with the SJ monitor and 4Mb of memory, you can have about 5000 blocks of VM disk. The Extended Memory monitor has been thoroughly reworked to feature "quality", "pleasantness", and "functionality" (DEC's words). The rework features a major internal cleanup, fetchable handlers, 22-bit addressing support, and a "SHOW MEMORY" display. With the XM monitor, it is now possible to have 65k words of double precision in a virtual array. You can get at the extended memory with virtual overlays, virtual .SETTOP, and the .VM programmed request.

RT-11 Version 5 will probably be going to field test in the next few weeks, and should be available for distribution in about May, 1983. It is hoped that FORTRAN 77 and an extremely extended BASIC will be available this spring for V5.

Other Software

Anticipating these new developments, Fairbrother Associates is offering a few programs for sale that either allow some of these features now in V4, or help with a problem we see coming up and can be used with V4.

The first, SETDTM is an enhancement over many of the programs available in the public domain for setting the date and time when the system is first started up. This program allows the entering of the minimum amount of data required for any input. Entering "1" causes the date to set itself to the first date of the next month, rolling over the year if necessary. An "A" would uniquely identify August, "JA" for January, etc. You could enter 23O82 for 23-OCT-82. "O" would set it to 1-OCT-82. If you boot the system again, it bypasses the input routine and prints out the time and date. These are saved by the program writing into itself and are printed out asking if they are correct, and if not, enter the correct data. You can also enable the "System" command, which asks "System?:" and will execute an indirect command file of the name entered in response to the prompt.

The second, LOCATE, allows you to enter physical or Virtual devices names and search for a file specification on all devices listed in their order as entered in the table. There are complete editing switches to alter the table when things change and a report, which gives the device and the names of any files found that match the selection criteria. This can be optionally directed to the printer. If the switch "/F" is used in the command line (which is the standard CSI - Command String Interpreter), the file is looked at as a Virtual device, and its directory is displayed. This is very handy when

you can't remember what exactly was on that floppy image.

The third is CLEAN, a program that looks at files with non-binary extensions and allows you to see what is inside and decide if you want to delete it. You can ask to see the first 23 lines, and can request another 23 lines, with the directory string for the file on the 24th line. You can then either bypass it or delete it, and then the program goes to the next file. If you do a lot of writing, this allows you to find a certain file, or to thin the disk of non-essential or non-useful files, freeing up disk space. I use it quite a bit.

The fourth program SPLMRG, for SPLit and MeRGe, is a program that allows you to back up one large file onto a number of smaller devices, usually floppy discs. This program is unique in that it uses a bit-map to restore the file, so entering the same disc twice does no harm, you can replace the discs in any order which will eliminate the most common failure. The program will also tell you if all the discs were used, and if not, which section(s) is missing. This allows a floppy to become a convenient method of Winchester backup, not a messy kludge.

The fifth program is COPVOL. This program copies all files from the input device to as many output volumes as required. The output volumes can be empty or contain other files. Normal operation preserves existing files with the same names. This program is just as fast as PIP (just as good as a Xerox!!!). The output volumes can be empty or contain other files.

Full wild card support is provided for both the file name and its extension. The default input is *.* (all files). This program will only work with input devices that have a standard directory starting at block 6. This includes RX01, RX02, RX03(DS/DD), RL01, RL02, RK05, RK06, and RK07. The normal mode of copying preserves the original file's date and does not process files that exist on the output device. A large assortment of switches allows most backup needs to be met.

The sixth program is FCHECK. This program will format and check a single or double density disk unit and report any errors found. One re-read on errors is performed. The error report shows the track, sector, and contents of the various registers for later analysis, as well as an indication of whether the error is hard (occurs twice) or not.

The program asks if you want to FORMAT a disk. A 'Y' response will initiate the formatting process. If a double-density controller is in use, the program asks if you want to format in SINGLE or DOUBLE density format. Typing a 'D' formats in Double Density; anything else becomes Single Density. If a double sided disk is detected then it will be processed accordingly. The user may also specify that a special VOLUME ID and OWNER be placed on the disk when it is initialized after it has been formatted successfully. If formatting has not been selected, a prompt for CONTINUOUS checking is printed.

Finally, there is SCAN, which has been described as a cross between PATCH and DUMP. This program searches a file for a match to a single word or byte, along with any mask you specify. When a match is found, a line with the following components is output; first, the block number, then the offset, the two words before the match, the matched section, and the two following words, along with RAD50 and ASCII translation of the five words. Very useful for debugging assembly language programs, etc.

We are selling the set at \$150.00, documentation on the media. The heavily commented sources are available for an additional \$300.00. We feel that people who need the source should be able to get them, but people who do not need them should not

have to pay for them, and that the price of the SAV disc is very modest considering the cost of other software for this market. The closest bargain we know of is SPELL-11, whose .SAV file and dictionary cost \$100.00, and whose sources cost an additional \$200.00. Incidentally, we also sell SPELL-11.

Several products are in the making, such as a FORTRAN callable library of RT11 utility subroutines that implement system functions in a straight forward manner. Included are routines to set and reset individual bits (useful in setting the JSW for "special mode", enabling lower case, checking if a file exists, its length, etc.), CRT functions, change lower case to upper case and the reverse, and others. We are also working on a list oriented database featuring all the conveniences that we can think of to make entering lists as easy as possible, with all sorts of compile-time and run-time options and defaults. Keystroke abbreviations are supported, along with range checking, formatting, and use of pictorial screens whenever they seem more useful. All defaults are resettable, and any option set at compile time can be over-ridden at the keyboard. The key structure will be a B* tree that will find any record in 1,000,000 in 1 or 2 seeks. We hope to have it completed just around the time V5 is released.

Others are on the way as we discover new needs to answer. If you have any ideas, write to us about them. The public domain programs are still available, and you can still get a list of them, along with notices on other stuff we sell. Send \$1.00 and a SASE with 40 cents postage on it to the address above and we will send back the literature.

Editor's note: Fairbrothers Associates also sells hardware to upgrade the H-11 to a full 11-23 with hard disk. Contact them at the address noted above.



If you want to become more productive with your computer. . .

DON'T

buy this game.

If you're like a lot of people, you bought your Heath computer to do home finance, word processing, or perhaps to learn how to program. You've got PerfectStar, DataPlan, MergeWriter, and all the latest Pascal compilers. Fine. Just don't buy GRAVITRON.

We're telling you this because we've noticed that people who play GRAVITRON get hooked. As they pass through the warp gates to the first planet, they realize that this is no ordinary game. They are intrigued by the challenge of maneuvering a spacecraft that is subject to the force of gravity. Their skill increases, and they move on to other worlds, each more extraordinary than the last. Soon they become obsessed with the desire to explore all the strange (and lethal) planets of GRAVITRON.

If you want to give it a try, go ahead. But you've been warned. Once you visit the worlds of GRAVITRON, you may not want to come back to Earth.

GRAVITRON, \$19.95 plus \$1.50 shipping. Requires H/Z89, Z90, or H8-H19, 32k, and HDOS or CP/M. Specify disk format (hard or soft sector 5-1/4"). Free catalog of games and utilities. CP/M t.m. Digital Research.

APOGEE SOFTWARE

Box 15124
Savannah, GA 31416

(912) 925-3765

Getting Started with Assembly Language

(or, Now That I've Finished the Heath Assembly Language Course, HELP!!)

Pat Swayne
Software Engineer

This article is the second in a series on assembly language for the beginner who knows what "MOV A,M" means, but does not know what to do with it. The first installment included Part I of the series, the introduction, and Part II on console I/O in HDOS. New REMark readers, especially those using HDOS, are urged to buy, borrow, or beg a copy of issue #38 and get caught up.

PART III — Console I/O in CP/M

As I stated in last month's article, console I/O means inputting information from your keyboard or outputting information to your console screen. This is an area where a lot of otherwise good programmers seem to get tripped up because they fail to realize the power built into the operating system to accomplish this task, and waste time and memory space developing their own I/O routines that don't work right after all of the trouble spent. For example, I have seen many good programs that will only allow either the Back Space or Delete key to be used to correct mistakes (not both), and "crash" (or, at least, cough a little) if the other key is used. In most cases, there was no need for the program to process "line input" (where you type a line of information and then hit RETURN to enter it) by itself in the first place, because the operating system (either HDOS or CP/M) has the ability to do it for you.

This brings up my first rule for assembly language programming, which I will repeat for those who missed last month's article.

Rule 1. If the operating system can do it, let it do it.

The operating system has the ability to handle many of the simple jobs that have to be done in programs, and part of good programming is taking advantage of that ability whenever possible.

Type My Name...

As I did last month for HDOS, I will use the famous "Type My Name Ten Times" program to illustrate console I/O in CP/M.

...in CP/M

Listing 1 is the CP/M version of the program. As before, I have treated it as an assembly language translation of a BASIC pro-

Listing 1

```

;      THIS PROGRAM IS AN ASSEMBLY VERSION OF THE
;      FOLLOWING BASIC PROGRAM
;
;      10 PRINT "HELLO, WHAT'S YOUR NAME";
;      20 INPUT N$
;      30 FOR I=1 TO 10
;      40 PRINT N$
;      50 NEXT I
;      60 END
;
;      THIS VERSION IS FOR CP/M, AND UTILIZES SOME
;      BUILT-IN CP/M ROUTINES
;
;      BY P. SWAYNE, HUG 20-MAY-82
;
;      DEFINE CP/M ROUTINES, ETC.
BOTTOM EQU 0           ;BOTTOM OF USER MEMORY
BDOS EQU BOTTOM+5      ;JUMP INTO CP/M BDOS
TPA EQU BOTTOM+100H    ;TRANSIENT PROGRAM AREA
CONOUT EQU 2          ;CP/M CONSOLE SINGLE CHARACTER OUTPUT
TYPE EQU 9            ;CP/M TYPE FUNCTION
LINPUT EQU 10         ;CP/M LINE INPUT FUNCTION

ORG TPA                ;MOST CP/M PROGRAMS START HERE

;      10 PRINT "HELLO, WHAT'S YOUR NAME";

START LXI H,0          ;ZERO HL REGISTER
      DAD SP           ;ADD STACK POINTER VALUE
      LXI SP,STACK     ;SET OUR STACK POINTER
      PUSH H           ;SAVE CP/M'S STACK ON OURS
      LXI D,HELLO      ;POINT TO "HELLO" STRING
      MVI C,TYPE       ;GET TYPE FUNCTION
      CALL BDOS        ;PRINT "HELLO" MESSAGE

;      20 INPUT N$

      LXI D,NAME       ;POINT TO WHERE NAME GOES
      MVI C,LINPUT     ;GET LINE INPUT FUNCTION
      CALL BDOS        ;INPUT THE NAME
      LXI H,NAME+1     ;POINT TO NO. OF CHARS TYPED
      MOV E,M          ;GET THE COUNT
      MVI D,0          ;DE = COUNT
      INX H            ;MOVE TO THE START OF THE NAME
      DAD D            ;ADD COUNT TO GET TO THE END
      MVI M,0DH        ;INSERT A CR
      INX H            ;MOVE TO NEXT LOCATION
      MVI M,0AH        ;INSERT A LINE FEED
      INX H

```

gram, which is listed as comments at the beginning of the assembly source. The BASIC program, as listed, will run properly under MBASIC or BASIC-E.

After the initial comments, the program contains 6 equate statements. Three of them define important memory addresses and the other three define the CP/M system calls that the program uses. In CP/M, there are two parts of the system containing routines that programs may use: the BDOS (Basic Disk Operating System), and the BIOS (Basic Input Output System). When possible, BDOS calls should be used instead of BIOS calls, because the BIOS is

not really part of CP/M, but rather is a program provided by the computer manufacturer to link CP/M to the manufacturer's system. Because of this, routines within various BIOS' may not be 100 percent standardized. In this series, I will not use any BIOS calls in programs.

A call to the BDOS is made by calling location 5 in memory with a value indicating the function required in the C register. At location 5 is a JMP instruction placed by CP/M that transfers control to the BDOS. Any parameters required by the particular BDOS call must be supplied in the E register (if one byte is required) or in the DE register

pair. If a parameter is to be returned, it will be returned in the A register (if one byte) or in the HL register pair. The various BDOS system calls are fully explained in the section of the Digital Research CP/M documentation called "CP/M 2.2 Interface Guide". Digital Research refers to BDOS system calls as "BDOS functions".

An origin statement (ORG) follows the system definitions in the program. The argument to it, called the Transient Program Area, is the area in memory where all CP/M programs must start, unless the program is not designed to be run as a command at the CP/M prompt (A>). Programs that start at a higher address can be loaded and run using the DDT debugging program supplied with CP/M.

The actual program begins with the label START. It is not really necessary to have a label at the beginning of the program in CP/M, (as it was with HDOS), but it makes the program look nicer. The first 6 program lines translate line 10 in the BASIC program, but the first 4 lines actually have nothing to do with line 10. The first two lines locate the stack pointer by clearing the HL register to zero and adding the stack register to it. The next line then sets up our own local stack. This is done because CP/M does not guarantee any specific amount of stack space for user programs. Finally, line 4 saves the old value of the stack register on our new stack. The reason for doing this will be explained when we get to the end of the program.

The line LXI D,HELLO begins the actual translation of line 10 of the BASIC program. In CP/M, there is no equivalent to the HDOS \$TYPTX routine that allows text to be printed on the screen to be included in the body of the program code. It would only take a few lines of code to implement such a routine, but we are going to stick with built in system routines whenever possible. CP/M has one system routine for printing blocks of text on the screen, which is BDOS function number 9. I called it TYPE in the equate statements at the beginning of the program. To use this routine, the DE register pair must have the address of the text to be printed, and the block of text itself must end with the character "\$" (which itself is not printed, and therefore cannot be used in text to be printed this way). Our message is located at the label HELLO, near the end of the program.

The next lines in the program translate line 20 of the BASIC program, the INPUT statement. As I mentioned last month, you must always provide places to put things in assembly language, so the first line in this sec-

```

MVI    M,'$'      ;INSERT A "$" (STRING TERMINATOR)
MVI    E,0AH      ;GET A LINE FEED
MVI    C,CONOUT    ;GET CONSOLE OUTPUT FUNCTION
CALL   BDOS       ;PRINT LINE FEED
;
30 FOR I=1 TO 10
MVI    B,10       ;SET UP A COUNTER
;
40 PRINT N$
PRINT  LXI    D,NAME+2 ;POINT TO NAME
MVI    C,TYPE     ;GET TYPE FUNCTION
PUSH   B          ;SAVE THE COUNT
CALL   BDOS       ;PRINT THE NAME
POP    B          ;RESTORE THE COUNT
;
50 NEXT I
DCR    B          ;DECREMENT COUNTER
JNZ    PRINT      ;LOOP UNTIL IT'S ZERO
;
60 END
POP    H          ;RESTORE CP/M'S STACK
SPHL                      ;SET IT
RET                                ;RETURN TO CP/M
;
PROMPT STRING
HELLO  DB    'HELLO, WHAT'S YOUR NAME? $'
;
PLACE TO STORE NAME (VARIABLE N$)
NAME   DB    30,0 ;MAX ALLOWED CHARACTERS, COUNT
DS     33      ;ALLOW 30 CHARACTER NAME + CR, LF, "$"
;
OUR PRIVATE STACK
DS     32      ;ALLOW 16 "PUSHES" (2 BYTES EACH)
STACK EQU    $ ;START STACK HERE
END    START   ;TELL ASSEMBLER WHERE PROGRAM STARTS

```

tion points the DE register pair to a location (NAME) where we will store the input name. You may recall from last month that HDOS has only one console input system call (.SCIN), which works in different ways depending on how another system call (.CONSL) is used. In CP/M, there are 3 different BDOS functions for console input that work in different ways. The simplest one (function 6) just polls the keyboard and returns the character if a key was struck, or zero if no character was struck. The next one (function 1) waits until a key has been struck before returning, and echoes the character to the screen if it is printable or a return, line feed, or back space. Tabs are echoed as equivalent spaces. This function intercepts Control-S (start/stop scroll) and Control-P (start/stop printer) for processing by CP/M. The third input function is the one used in our program (function 10). It is a sophisticated line input routine that allows you to type a line of text, with several ways to correct mistakes, and then hit RETURN to enter the text. If a Control-C is typed as the first character when you use this function, it forces a warm boot back to CP/M from your program. To learn which other control characters affect this function, see the discussion of it in the "Interface Guide".

The first two bytes of the buffer where the BDOS line input function places input text are special. The first one limits the number of characters that can be input, and can be set to any value from 1 to 255. If you type that number of characters before you hit RETURN, CP/M will return control to your program without waiting for a RETURN. The next byte in the buffer is set by CP/M to the number of characters actually typed (minus the RETURN in the case that fewer than the maximum were typed). In our program, the character limit is set to 30.

After you type in your name, the program gets the character count that CP/M has set in and adds it to the address of the start of the actual text (the third byte in the buffer) to locate the end of the text. It then inserts a return, a line feed, and the character "\$" so that we can later use the same text printing function used earlier to print the name.

When it has processed the name you typed in, the program uses a BDOS function (CONOUT, function 2) to print a line feed on the screen. We had to do that because CP/M leaves the cursor on the same line after a line input. Just as there are three input functions in the BDOS, there are also three output functions. We have used the

block output function (function 9), and the single character output function. While either of these functions is in use, CP/M checks the keyboard for Control-S or Control-P and processes them if they have been typed. These functions also convert TAB characters to equivalent spaces, which can be undesirable if you are using TABs to achieve rapid screen writing or you are mixing TAB characters and escape sequences. To get around this, function 6, mentioned above, can also be used for output. This function, called the Direct Console I/O function, will perform input if OFFH is placed in the E register when it is called. If any other value is in the E register, it outputs that value as a character. I used function 2 in my DIR19 program (HUG disk 885-1226) when I first did it and wondered why it seemed sluggish at writing to the screen until I figured out that TAB characters were being expanded to spaces. So I switched to function 6 and the screen was "painted" much faster. Function 6 is a must if you are writing a fast action graphics game.

The next BASIC line translated is the FOR statement in line 30. In the BASIC program, we set a counter (I) to 1, and then enter a loop in which the task is performed (PRINT N\$) and the counter is incremented and tested (next I). When the counter is greater than the upper limit of 10, the program drops out of the loop. In assembly language, it is more efficient to set the counter to the number of times we want to loop, and then decrement and test it after performing the task. The reason for this is that when you decrement any register, the processor's condition flags are set according to the result of the operation, and so the zero flag will be set when the count reaches zero. On the other hand, if we chose to start at the bottom and count up as in BASIC, we would have to add additional instructions to compare the register being incremented with the upper limit to determine whether the loop was finished. So for the sake of efficiency, this program counts backwards, and line 30 of the BASIC program is translated as MVI B,10.

Next, we translate line 40, PRINT N\$. We do it by using the same BDOS function that was used to print the initial "HELLO" message. Since the character limit and the character count occupy the first two locations in the buffer NAME where the name typed in is stored, the program points the DE register pair at NAME+2 as the address of the name to be typed. The B register is saved while the program calls the BDOS because it contains our loop counter. The label PRINT at the beginning of this sections es-

tablishes the address to jump to for reprinting the name 10 times.

The next line in the BASIC program (NEXT I) is translated by decrementing the counter (B register) and jumping back to PRINT if it is not zero.

The END statement in the BASIC program is replaced with code that returns control to CP/M. Recall that we saved the value of the stack pointer as maintained by CP/M on our own local stack at the beginning of the program. Here, we restore that value, put it back into the stack register, and then RETURN. This gets us back to CP/M instantly without any disk access, because when CP/M runs a program, it actually CALLS it from the CCP (Console Command Processor) after loading it into memory from the disk.

As long as a program does not over-write the CCP, it can return to CP/M that way. Otherwise, a jump to address zero or just a RST 0 will do the job, but it takes longer since a complete warm boot is performed.

The storage place for the user's name follows the exit code. After allowing two bytes for the character limit and count, the DS 33 statement reserves 30 characters for the name plus one each for a return, line feed, and the terminator character "\$". Following this, another DS statement reserves space for our local stack pointer. For stack operations, the reserved space must come BEFORE the label that defines the stack because the stack always grows downwards in memory as it is used. So, after the DS statement, we have the line STACK EQU \$ to initially set the stack pointer after the reserved space.

The END statement at the end of the program has an argument START which, you may recall, was necessary in HDOS. This argument informs the assembler of the entry point of the program which is really not necessary in CP/M since all normal programs must start execution at the bottom of the TPA (100H).

This completes my discussion of console I/O. Next month, I will show you how to output to a printer in assembly language. ✖

**Join with other
HUGgies
at the
1983 NATIONAL
HUG CONFERENCE**



CheapCalc Revisited

Clifford C. Lundberg
112 Elk Hills Dr.
Elk River, MN 55330

The electronic spread sheet concept intrigued me, but I was never able to justify the expense until CheapCalc.

After reading Bob McFarland's article in Issue 37 of REMark I had to try this BASIC spread sheet. I found it fascinating and easy to use.

I had some experience with VisiCalc on the Apple and TRS-80 computers at the Elk River High School, where I teach an evening community education course in Beginning Computer Literacy, and wanted to get a similar program for use at home on my H-8. I find that this CheapCalc program fits my small needs very well. While it does not have the speed and frills of the high price brand, it does have the most important features. It provides for string headings and formula entries, including a special SUM formula. It provides for setting ones own column widths with a 9 character default width.

I did encounter some bugs which I have corrected and shall explain. I also added some cosmetic enhancements.

At line 165 add the statement:

```
165 PRINT E$+"w"
```

This will set the H/Z-19 terminal to discard at the end of the line which is necessary to provide a proper display on the screen.

I prefer to have my column headings right justified and I modified lines 780-790 as follows:

```
780 Z4=1: IF LEFT$(A$(Y7,X7),1)=CHR$(34) THEN Z4=2
```

This is to remove the "" as the first character in a string entry from the display variable B\$(Y%,X%).

```
786 H$=S$+MID$(A$(Y7,X7),Z4,L7)
```

This provides the right justification for the string by putting in leading spaces rather than following spaces as was done in original line 780.

```
790 H$=RIGHT$(H$,CN(X7))
```

This sets the string to the proper length for the column width.

By going to row 40 the screen would print additional rows which were outside the capability of the program. To solve this and provide a display (after using the GOTO command), make the following modifications:

```
1965 IF SY7+Y17>MR7 THEN 1990
```

This skips the printing of undimensioned variables outside the programs row capacity.

```
2350 IF Y7>18+SY7 THEN X37=-1:SY7=SY7+10:Y7=SY7+9
```

This backs up the row counters to put the desired item in about the center of the screen (top to bottom).

```
2360 IF Y7>MR7 THEN Y7=MR7:SY7=MR7-18
```

This correction just corrects an error in the variable identification but the function is to stop you from indexing past the capability of the program.

```
2500 SY7=SY7-10: IF Y7<=0 THEN Y7=1:SY7=1
```

This stops you from reverse indexing the rows past 1.

```
3462 SY7=SY7-18: IF SY7<1 THEN SY7=1
```

This helps format the rows on the screen so you can see the row prior and subsequent to the desired item.

```
3464 SX7=SY7-3: IF SX7<1 THEN SX7=1
```

This allows you to view the columns both before and after the desired item. Before this, when you used the GOTO command you lost sight of all the columns before the desired item.

```
3530 IF Y17<1 OR Y17>MR7 THEN X17=0:Y17=0
```

Because of other modifications I did not need the "-20" offset in the original line.

Please note that line 2180 moves DOWN one row and line 2190 moves UP one row contrary to the original comments to these lines.

I was unable to get the CLEAR command to work because MBASIC intercepts the symbol and would not return a 64 at line 2230. Therefore I changed the code word to the symbol and changed the number 64 in lines 2230 and 2580 to number 35. The CLEAR command now works well. (NOTE: this is the entry clear command - not the screen clear command.)

In order to properly clear the top 3 screen lines during certain routines I made these 2 line changes:

```
3180 GOSUB 3340:PRINT FN PC$(1,1);  
2690 PRINT FN PC$(2,1);CL$;"SUM(";A$;"THRU ";B$;")"
```

Also insert a CL\$ in the PRINT statement at lines 3590 & 3610.

In order to leave a flag to denote a string entry for an item, I changed line 2270 thus:

```
2270 IF A=34 THEN 2570
```

This will leave the " sign in the input variable to be used as a test flag for the replication command.

When using the screen Clear command I was not able to clear the spread sheet. I had thought that this command would give me a virgin sheet. To correct this, change the number 250 to 130 in line 2780 thus effectively restarting the program and insuring that all variables would return to their virgin state.

Because of the change we have made to the output variable string (right justification), a change should be made in the printer routine at lines 3660 and 3670. The word LEFT\$ should be changed to RIGHT\$ or perhaps left out entirely as superfluous.

Using a Control-D as input tells the program to exit. The exit line is 3740. Before ending I asked the computer to PRINT E\$+"z" to reset the H/Z-19 terminal to its startup status. This also clears the screen.

Another nice touch was to change the INPUT statement at line 2770 to a PRINT statement and then add line 2775 to provide a single key input. Then add an error catcher at line 2782.

```
2775 A$=INPUT$(1)
```

```
2782 GOTO 2760
```

The REPLICATION function gave me the most trouble. I could not accurately replicate a string heading that had the letters A-P in it because the program assumes such a letter to be a column heading and wants to change the next value by the difference between where we came from to where we are going. This is an excellent feature for the relative replication of formulas and I am most impressed with it for that purpose. However, it would not work properly with the SUM function formula starting at line 2640. One problem was the letter M in the word sum. I solved that problem by adding line 3815.

3815 IF I=1 AND A\$="M" THEN I=5

This allows us to skip over the troublesome letter M.

The next problem encountered was replicating a formula such as &SUM(B9-J9) at K9 to K10. I could not get the replication to give me the formula &SUM(B10-J10). The problem was caused by the increase in the length of the row number. This was solved by adding some lines to test for any change in formula length (either plus or minus) and then modifying line 3840 to correct of such

length change.

3840 A\$(Y7,XZ)=MID\$(A\$(Y7,XZ),1,I-1+LI)+CHR\$(A+DXZ)

3862 LD=LEN(N\$)-LN:LI=LI+LD

3802 LD=0

Also I changed the GOTO line number at the end of line 3850 from 3850 to 3880 to avoid an infinite loop that sometimes occurred.

Next add a line to test for a string constant and allow column or row headings to be replicated.

3812 IF I=1 AND A\$=CHR\$(34) THEN I=LA:GOTO 3880

Finally add a GOSUB 350 in line 3880 to compute and display the replication results.

3880 NEXT:CR=0:GOSUB 350:GOTO 209

With these modifications, I have enjoyed working with my affordable BASIC spread sheet. Happy Computing.



YOU ARE INVITED TO THE 1983 NATIONAL HUG CONFERENCE

O'Hare Hyatt Regency Hotel, Chicago, Illinois . . . August 19, 20 and 21

New DATABASE UTILITIES . . . AUDIT

AUDIT is a full-featured data base management system which is completely menu-driven for ease of operation. The program is supplied in binary format for efficient execution speed. **AUDIT** is user-friendly and all prompts are in everyday ENGLISH syntax. User-defined input forms provide for easy data entry. Up to 19 different alphanumeric data entry types are supported. Multiple input screens can be created for each database to allow different "views" of the same data. Index files are used to provide fast access to database records. Multiple indexes may be created for each database. Database records can be sorted and selected before data output. Multiple output formats can be created for each database. Each output format is user-defined and can include headings, computed values, subtotals, and totals. Label output is also available with 1 to 4 labels across.

Requires 64K (CP/M only) \$129.00
Includes 3 disks, 35 page manual, and 3-ring binder

MICRO-CABINET

MICRO-CABINET is an efficient electronic filing system. Since **MICRO-CABINET** is constructed entirely of assembly code, the program is extremely fast and efficient. Keyed access is used to retrieve database records very quickly. User-defined input forms provide for easy data entry. Fixed and Variable length fields may be defined in order to maximize disk space usage. Data editing is provided with 5 different data entry types. With **MICRO-CABINET**, records are always accessible in sorted order by key value. A HELP screen and record LIST option are available in UPDATE mode. Record selection criteria may be specified before record output. Output formats are user-defined.

Requires 48K, HDOS or CP/M \$75.00



P.O. BOX 790 MARQUETTE, MI 49855

Call: (906) 475-7151 10 am - 5 pm EST

Call or write for more information about our full line of products and our FREE Summer 1983 catalog.

All products are available in HDOS and CP/M format, except where noted. Payment may be via check or money order. Add \$2 for shipping. Michigan residents please add 4% for sales tax. PAYMENT IN U.S. FUNDS PLEASE.

Z-100 SOFTWARE! !

That's right! Our complete line of quality software products is available now on soft-sector disks for CP/M-85!

ATTENTION SOFTWARE AUTHORS:

GENERIC SOFTWARE is interested in high quality and well documented software for HEATH/ZENITH computers. GENERIC offers professional packaging, and high royalties. If you are interested in making money from the software you develop, then request our FREE booklet, "SOFTWARE AUTHOR'S KIT"!

And more New Software!

New Stock Market Program . . . FUND-MASTER

FUND-MASTER is a stock market investment aid which is especially useful for mutual fund investments. **FUND-MASTER** is for the casual investor who does not have time to become familiar with all aspects of the stock market, but yet needs effective buy and sell advice. **FUND-MASTER** takes a conservative approach to stock investments, with the emphasis on obtaining consistent returns. **FUND-MASTER** can be used on a periodic basis to obtain timely buy & sell instructions. **FUND-MASTER** logs investment transactions, as well as providing the current investment status. **FUND-MASTER** is an easy to use, yet effective, investment tool for stocks and mutual funds. Requires 56K, BASIC Interpreter, HDOS or CP/M

..... \$39.95

Unique New Adventure Game . . . CASTLE-GOR

Journey to the Black Forest in search of **CASTLE-GOR**, where strange creatures and huge fortunes await! The object of this new fantasy/adventure game is to collect as many treasures as you can and escape the mysterious dungeons of the **CASTLE**. Nearly 400 rooms await you, with surprises around every corner. During your expedition, you must face hostile guards, deadly traps, and challenging mazes. No two games are alike, whether you win or lose. The random nature of **CASTLE-GOR** makes it a truly unique adventure experience. **CASTLE-GOR** comes with a User's Dungeon Manual, two disk volumes, and a gift certificate for anyone who is able to get out of the Castle with all the treasures!

Requires 64K, CP/M and BASIC Interpreter . . . \$24.95
(HDOS version available soon)

Putting the H-8 LED's to Work

Glenn F. Roberts
 9035 F Countrywood Drive
 Knoxville, TN 37923

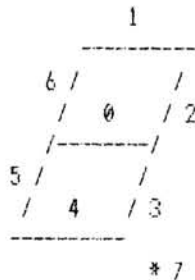
One of the main reasons I chose the H-8 over the H/Z-89 was the power and versatility of the H-8's Front Panel (FP). While designed originally to make the computer usable with only a bare minimum of peripherals (e.g. a tape recorder and 8K of memory), the FP can still serve you well with a 48K disk based system. (Gordon Letwin gives a good overview of the H-8 FP in his article "PAM/8: A New Approach to Front Panel Design", BYTE, Vol. 3, No. 10, Oct. 1978, pp. 70-84.) In this article I will provide the information needed to put the FP to work, and end up by describing a software real time clock for the FP l.e.d.'s.

The Hardware

Before you begin to program the FP you should take some time to understand the basic hardware used. The engineers at Heath came up with a very nice design here. The entire Front Panel including the 9 seven segment l.e.d.'s, the 16 button keypad, the 1024Hz speaker, and several other options are controlled using just two system ports!

Throughout most of this discussion, I'll be primarily interested in the FP l.e.d. display. This display consists of 9 light emitting diode (l.e.d.) display units each containing 7 lightable segments plus a decimal point. The segments are referred to by number as shown in Figure 1. The segments draw enough current that they cannot be controlled by standard TTL logic but must be buffered through the use of either switching transistors or special IC's such as the 7405 open collector hex inverter. To allocate one buffer to each segment would require 72 (9 x 8) such drivers and the allocation of 9 ports to address these digits.

Figure 1. Bits in segment latch corresponding to each segment.



A better approach is to turn the digits on and off in sequence using software to "multiplex" the display. If this is done fast enough, the human eye will not detect any flicker and the hardware requirement will be reduced to 17 drivers (9 + 8) and 1 1/2 ports. This is the approach used in the H-8.

The driver circuitry is divided into two portions: the segment select circuit and the digit select circuit. The segment select circuit consists of 8 PNP switching transistors (2N4121) interfaced to two quad D-type latches (74LS175) which are decoded to be addressed at octal port 361 (241 decimal). These 8 latches are cleared by a pulse from the clock every 2 ms., thus information placed here must be continuously refreshed. The digit select circuit consists of one quad D-type latch (74LS175) connected to a 4-to-10 line decoder (MC14028). This arrangement selects one and only one digit based on the binary value on the lowest 4 bits addressed at octal port 360 (240 decimal). The common cathode of the selected digit is pulled to ground via an NPN transistor (MPSA13)

and the pattern currently stored in the segment select latches is displayed. Since only outputs 1 to 9 of the 4-to-10 decoder are used, when output 0 is selected all digits are blank.

At this point I should mention that the upper 4 bits addressed at octal port 360 are also latched and used to control various functions. Table 1 shows these functions.

Table 1. Functions of highest 4 bits latched at octal port 360.

Bit	Function
4	Single Step interrupt
5	Monitor l.e.d.
6	Clock interrupt
7	1024Hz speaker tone

The two most useful of these are bits 6 and 7. More about these later.

The ROM Software

Contained in the H-8 monitor ROM are routines to support the FP l.e.d.'s, and specifically to allow four types of displayed information:

- 1) The contents of any memory location
- 2) The contents of any register pair
- 3) Any user defined pattern
- 4) Nothing at all.

The display mode is controlled by several bytes stored in the monitor ROM and in low user RAM. The addresses of these bytes are given in Table 2:

Table 2. Important bytes for controlling the H-8 Front Panel l.e.d.'s.

Byte	Address	
	Octal	Decimal
DODA	040.356	1006
REGI	040.005	8197
DSPMOD	040.007	8199
.MFLAG	040.010	8200
CTLFLG	040.011	8201
FPLEDS	040.013	8203
ABUSE	040.024	8212
UIVEC	040.037	8223

The following is a brief description of each of these bytes.

DODA is a table of patterns to form the familiar 7 segment representations of the digits 0 through 9.

REGI determines which register is to be displayed on the l.e.d.'s when in register mode (DSPMOD = 2). The bytes corresponding to each register are shown in Table 3.

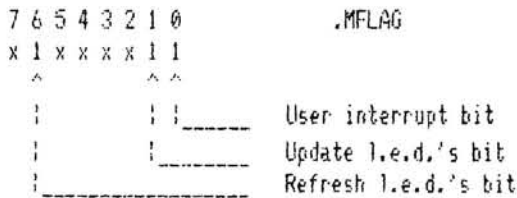
Table 3. Values in REGI corresponding to the various register pairs.

Value	Register pair displayed
0	SP
2	AF
4	BC
6	DE
8	HL
10	PC

DSPMOD determines the mode of display. The two commonly used modes are DSPMOD = 0 for memory display, and DSPMOD = 2 for register display.

.MFLAG determines the state of various user options. There are three such options pertinent to this discussion: user interrupt processing, l.e.d. refresh, and l.e.d. update (see Fig. 2).

Figure 2. Useful bits in the .MFLAG byte.



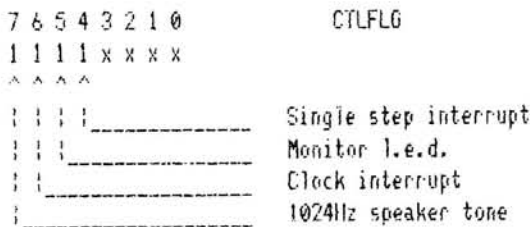
1) When the user interrupt bit (bit 0) is set, the user is indicating that he or she has supplied a routine to be processed at each clock (2 ms.) interrupt. The address of this routine must be stored in location UIVC. The routine will be processed after normal processing of the l.e.d. refresh and update.

2) When the update bit (bit 1) is set, the normal updating of the information on the display is skipped, however the l.e.d.'s continue to be refreshed. This mode is useful for displaying user supplied information.

3) When the refresh bit (bit 6) is set, normal refresh of the l.e.d.'s will stop. Since the segment select latch is reset every 2 ms., the l.e.d.'s will remain blank without this refresh.

CTLFLG contains the bit settings which are latched onto the high four bits at octal port 360 every interrupt (see Fig. 3). These bits correspond to the functions shown previously in Table 1.

Figure 3. Useful bits in the CTLFLG byte.



1) Bit 4 controls the single step mode of the H-8. When cleared, it requests a level 2 interrupt after completion of each instruction. This feature is used in memory display mode (DSPMOD = 0).

Using the FP l.e.d.'s

One useful application of the FP l.e.d.'s is in debugging a program using the Heath DBUG program. It is often useful to monitor a

register pair or memory location while single stepping through a program. This can be accomplished by placing the appropriate values in the various control bytes. For example, the following shows how to monitor the BC register pair by setting DSPMOD to 2 and REGI to 4:

```

>debug
HDOS DBUG # 102.06.00.
:B:040007=000/2
:B:040005=000/4
:B:
    
```

To then monitor a memory location, say 042305, we could change the value of DSPMOD to 0 and place the desired address in ABUSS:

```

:B:040007=002/0
:B:040024=000/305
:B:040025=000/042
:B:
    
```

Note that the display will remain in this mode indefinitely, (even if the system is rebooted).

Benton Harbor Basic provides some nice support capabilities for the FP including the CNTRL 2, PAD, and SEG functions. CNTRL 2 gives you direct access to the display update and refresh bits. PAD allows access to the ROM routine RCK which reads the FP keyboard. SEG provides access to the DODA segment lookup table in ROM. Page 6-72 of the HDOS Software Reference Manual shows a simple example of the use of these functions. Another example is shown in Listing 1. This is a simple subroutine to flash the word "Error" on the H-8 l.e.d.'s while turning the 1024Hz speaker off and on. The masks for the letters are constructed using the information in Figure 1. Note that 1's correspond to segments which are off and 0's to segments which are on. The speaker is turned on and off by toggling bit 7 in the CTLFLG byte.

Listing 1. A simple subroutine in B.H. Basic showing one way to use the FP to flash a message.

```

01010 REM *** Subroutine to flash "Error" on H-8 l.e.d.'s
01020 REM
01030 E=140           :REM ( 10001100 = "E" )
01040 r=222          :REM ( 11011110 = "r" )
01050 a=198          :REM ( 11000110 = "a" )
01060 b=255          :REM ( 11111111 = " " )
01070 F1=127         :REM ( 01111111 = speaker on )
01080 F2=128         :REM ( 10000000 = speaker off )
01090 CNTRL 2,1      :REM Turn off display update
01100 A=8203         :REM Address of FP l.e.d.'s
01110 C=8201         :REM Address of CTLFLG
01120 POKE A,b       :REM Blank first two digits
01130 POKE A+1,b
01140 POKE A+7,b     :REM And last two digits
01150 POKE A+8,b
01160 POKE A+2,E     :REM Then spell out "Error"
01170 POKE A+3,r
01180 POKE A+4,r
01190 POKE A+5,o
01200 POKE A+6,r
01210 POKE C,PEEK(C) AND F1 :REM Turn on speaker bit
01220 PAUSE 50       :REM Wait
01230 FOR I=A+2 TO A+6 :REM Erase "Error" message
01240 POKE I,b
    
```

```

01250 NEXT I
01260 POKE C,PEEK(C) OR F2 :REM Turn off speaker bit
01270 PAUSE 50 :REM Wait
01280 IF CIN(0)<0 GOTO 1160 :REM Repeat until CR key hit
01290 CNTRL Z,0 :REM Turn off display refresh
01300 RETURN

```

A surprisingly large number of letters can be displayed on the l.e.d.'s by considering both upper and lower case formats. Several Heath and HUG programs including ONECOPY and DUP utilize this capability nicely.

A Real Time Clock

In spite of these nice tricks one can do, most of the time you will probably not need the H-8 FP l.e.d.'s and they will sit idly flickering away. One obvious way to put this display to work is to make it show the time of day. This is fairly easy to do by making some modifications to the screen CK: clock driver that was presented in REMark issue #22 (also available from HUG on 885-1105). This also conveniently solves two problems that exist with this screen clock, the first being deciding where on the screen to put the clock, and the second being the relatively long time required to write 16 characters to the H-19 every second.

To implement the H-8 FP clock in HDOS you must perform the following. First of all, you may remove the following equates:

```

SPORT EQU 3550    CONSOLE STATUS PORT
SPURT EQU 3500    CONSOLE DATA PORT
OUTBIT EQU 400    OUTPUT READY BIT

```

and replace them with the following:

```

%DADA EQU 30101A    %DADA routine
DODA EQU 3356A     Pattern table
FPLEDS EQU 40013A  Front Panel LEDs
UO.DDU EQU 00000010B Disable Display Update
.MFLAG EQU 40010A User flag options byte
FP.DEC EQU 10000000B Mask for FP decimal point
FP.OFF EQU 11111111B Mask for FP all segments off
FP.S04 EQU 00010000B Mask for FP LED segment 4.

```

Since the H-8 clock will run much faster, change the calibration factor:

```
CAL EQU -1-500 Calibration factor.
```

Next delete the old routine PTIME (from the statement labeled PTIME through the RET instruction after the statement labeled OUTCH1) and replace it with the following new PTIME routine:

```

PTIME LXI H,.MFLAG    Point to user flag options
      MOV A,M          and get current settings
      ORI UO.DDU       Disable display update
      MOV M,A         then save MFLAG options
      MVI A,FP.OFF     Get mask for all seg's off
      STA FPLEDS       and turn off first LED
      LXI D,FPLEDS+1  Point (DE) to 2nd FP LED
      LXI H,TIMEBUF    Point (HL) to ASCII time
      DGLoop MOV A,M   Get an ASCII character.
      CPI NL           If NL then
      JZ CLKRET        update is done,
      CPI '0'          Check if character
      JC DGSKIP         is greater than zero

```

```

CPI '9'+i and less than nine, if not
JNC DGSKIP then skip to blank LED
SUI '0' [it's a digit! calculate offset
PUSH H save ASCII time pointer
LXI H,DODA point to LED pattern table,
CALL %DADA. add offset
MOV A,M and load the pattern
POP H restore ASCII pointer
ORI FP.DEC clear the decimal point
JMP DFFON and jump to turn on LED
DGSKIP MVI A,FP.OFF-FP.S04 Load mask for seg. 4 only
DFFON XCHG Point (HL) to FP LED,
MOV M,A set the FP digit,
XCHG then restore FP pointer to (DE).
INX D Point to next FP digit
INX H and next ASCII digit
JMP DGLoop and keep looping.
CLKRET EQU * CK: return. Address to
JMP 0 be filled in.

```

Finally, you may delete the two lines

```

SAVCUR DB 27,'j',27,'Y h',NL
RESCUR DB 27,'k',NL

```

The main change from the H-19 clock driver is that the routine PTIME, instead of writing the ASCII time to the screen, now updates the H-8 l.e.d.'s. Note that we must reset the UO.DDU (Disable Display Update) bit every time we update the display. This is necessary because some of the HDOS system routines (notably .EXIT) normally clear this bit. With the CK: driver programmed this way, the clock will continue to run until the system is rebooted. One other note: I use FP.S04 to cause segment 4 (see Fig. 1) to be lit between the hours, minutes, and seconds. You may want to experiment with other segments or the decimal point.

Once you've assembled the code, renamed the ABS file to CK.DVD, and rebooted your system, you may LOAD the driver and read and write to it using PIP or any user program. You may also use the prologue program in REMark issue #22 to explicitly load and set your clock each time you boot up.

Summary

This has been just a sample of the many ways to make use of the H-8 Front Panel in user programs. Others include home security programs, games, telephone dialers, etc. With a little imagination, you should be able to come up with your own clever uses for this versatile feature of the H-8 computer. As always, however, when PEEKing and POKEing around with the system control bits be extra careful!



**YOU ARE INVITED TO THE
SECOND
HUG NATIONAL
CONFERENCE
see details on page 7 of this issue**

"Escape" with FORTRAN and PASCAL

*John F. Sams
2313 Grandview
Springfield, Ill, 62702*

I read with interest the article by Kenneth Mortimer in the August 1982 REMark (Issue 31) regarding the use of escape codes in BASIC. Since I had just finished a similar study and implementation of the same with FORTRAN and PASCAL languages, I thought it would be of general interest to share some of my experiences with other HUGgies who may also be working with FORTRAN and/or PASCAL.

There seems to be a real lack of information available to bridge the gap between programming language tutorials and utilizing the capabilities of a particular hardware system they have been installed on. Part of the mystique surrounding the use of FORTRAN on current operating systems is due in part because the language was written in the era of punched card systems. Most of the tutorials I have personally encountered begin by describing the language with examples depicting punched cards to illustrate I/O format examples. Quite frankly I have not seen a card reader or punch on a micro or even a mini system for quite some time! Seriously though its easy to understand how a quick browse through these materials could cause the uninitiated to pass over the language as being outdated. This problem is not quite as serious for PASCAL as it is coming into vogue as one of "the" micro language of the 80's. I am sure that there is still some questions concerning the actual application of the PASCAL language elements for the H/Z-89/19 hardware environments.

This article will attempt to bridge a few gaps between the capabilities available in FORTRAN and PASCAL languages to implement the excellent software features of the Heath H/Z-89 and H/Z-19 terminals. Once a grasp of the initial concepts and ground rules are understood it will be quite simple to extend the methods to obtain a complete library of functions as Mr. Mortimer described in his fine article for BASIC. The program examples contained in this article were written in HDOS versions of Microsoft's Fortran-80 and Polybyte's Lucidata Pascal. This article will assume the reader has a working knowledge of program syntax and construction of either FORTRAN or PASCAL (or both).

The following text references examples of screen and cursor control functions shown in side by side comparisons in both languages. I will start the examples with one of the most essential functions required by most programs; that is the ability to clear the

screen and position the cursor at some desired location. Figure 1 shows how this may be accomplished. It is interesting to note that the program syntax for the PASCAL version bears a striking resemblance to the same function as it would be coded in BASIC.

FIGURE 1.

FORTRAN	PASCAL
Clear Screen, Cursor to Row 1, Col 1	
<pre> SUBROUTINE CLRSCRN BYTE ESC DATA ESC /27/ WRITE(1,10)ESC 10 FORMAT(1H ,A1,'E') RETURN END </pre>	<pre> PROCEDURE CLRSCRN; BEGIN WRITE(CHR(27),'E'); END; </pre>

The FORTRAN example is not quite as straight forward owing to the language's lack of true string handling capabilities; but with an understanding how FORTRAN passes information to the system devices it is possible to accomplish the task quite easily. FORTRAN is line oriented, reading and writing data which is first "described" with format statements. When writing to a device the first byte of information passed is a "format" control character which tells the device what to do (next line, top of form, etc.), prior to writing the following buffer. Unfortunately FORTRAN terminal I/O, unlike PASCAL or BASIC, is always preceded with a new line command prior to outputting the body of the format statement. Even though the format control byte is ignored and discarded, the byte must still be present or else the first character of a positioning command or text message will suddenly disappear. To satisfy this requirement, one need merely code a "space" character at the beginning of each format statement. Make no mistake regarding this small item, forget it and you'll be wondering why things cease to function in an orderly fashion!

Once the idiosyncrasies of FORTRAN's formatted I/O have been satisfied we can get on with the business at hand. To achieve the "clear screen" function on an H/Z-89 or H/Z-19 we must output an Escape and capital "E" characters. To generate the Escape code we first declare a byte variable and let the

compiler generate the decimal code of an Escape (27), with a DATA statement. The Escape is output with an "A1" format specification which passes the numeric value direct to the CRT terminal. The A-format performs the same job as the PASCAL "CHR(27)" and

BASIC's "CHR\$(27)". The "E" is specified and written as an ASCII literal character in the same manner for PASCAL, FORTRAN, or BASIC.

The examples in Figure 2 illustrate the basics extended to provide more complex capabilities such as cursor positioning and utilizing the 25th line of the terminal. In the case of FORTRAN they can be placed in a run-time library which is referenced when the code is linked prior to creation of the ".ABS" module. Depending on your PASCAL system the functions may be compiled into a library for linking similar to FORTRAN or be placed in a source library that can be included in automatically when the source mainline code is being compiled. The real power of programming in FORTRAN or PASCAL becomes apparent when you realize that these functions need only be coded and compiled once.

The examples shown in Figure 2 cover the different combinations needed to generate all the available Heath screen Escape functions. Please note the "Position Cursor" example in Figure 2, the Fortran subroutine does not contain the actual "write" command to position the cursor. The reason for this will be explained in a later section. I won't dwell on describing all the code sequences as you can readily see they are all similar in syntax and are described in detail in the Heath H/Z-89 and H/Z-19 reference manuals.

FIGURE 2

FORTRAN	PASCAL
Position Cursor	
<pre> SUBROUTINE MOVCUR(IROW,IR,ICOL,IC) BYTE ESC,IR,IC COMMON /AREA/ESC IR=IROW+31 IC=ICOL+31 RETURN END </pre>	<pre> PROCEDURE MOVCUR(ROW,COL:INTEGER); VAR R,C:INTEGER; BEGIN R := 31+ROW; C := 31+COL; WRITE(CHR(27),'Y',CHR(R),CHR(C)); END; </pre>
Turn ON Reverse Video	
<pre> SUBROUTINE REVON BYTE ESC COMMON /AREA/ESC WRITE(1,10)ESC 10 FORMAT(1H ,A1,'p') RETURN END </pre>	<pre> PROCEDURE REVON; BEGIN; WRITE(CHR(27),'p'); END; </pre>
Turn OFF Reverse Video	
<pre> SUBROUTINE REVOFF BYTE ESC COMMON /AREA/ESC WRITE(1,10)ESC 10 FORMAT(1H ,A1,'q') RETURN END </pre>	<pre> PROCEDURE REVOFF; BEGIN; WRITE(CHR(27),'q'); END; </pre>
Enable 25th Line Function	
<pre> SUBROUTINE ENAB25 BYTE ESC COMMON /AREA/ESC WRITE(1,10)ESC 10 FORMAT(1H ,A1,'x1') RETURN END </pre>	<pre> PROCEDURE ENAB25; BEGIN; WRITE(CHR(27),'x','1'); END; </pre>

To further illustrate the use of the "Building Block" approach to programming, the subroutines and procedures in Figures 1 and 2 have been combined to create "libraries" for use by the mainline examples in Figure 3. The simple FORTRAN and PASCAL mainline programs in the following example perform the following tasks to show how the individual components are brought together to accomplish a fairly complex sequence of events such as:

- 1-Clear Screen
- 2-Print message in center of Screen
- 3-Turn ON Reverse Video
- 4-Print message in 25th Line

- 5-Move Cursor to Row & Col 1
- 6-Turn OFF Reverse Video
- 7-Print final message and exit to Opr Sys

If you study the Fortran program you will note that the cursor positioning and message writing is combined in one WRITE statement. This is necessary to get around the Format processor which forces a new line prior to executing the contents of the format specification of the WRITE statement. If we follow cursor positioning immediately with the output message it will appear where desired. If the message is written in a separate statement after the cursor is positioned, you will find the message appearing in the next

row instead of the specified cursor position. This is exactly what happened to my example when I first coded it. More about "unwanted" carriage control in the last section.

Although I have not specifically done so it would be quite easy to combine most of the individual functions in one subroutine or procedure and control its operation by passing a control parameter to it. This parameter could be the actual ASCII control which is output with the escape sequence or you could number the functions and specify them as a certain numerical value which would reference a table of codes within the sub-program. Let your imagination go and develop routines which do the most for your actual needs!

SOMECAUTIONS!!

Tread lightly when you begin to test your code. Don't try to "Nest" a number of functions until you're sure each of the individual commands are executing exactly the way you want. One surprise I encountered occurred when I tried to execute the following program actions:

- Position the cursor,
- Turn ON Reverse Video,
- Write a Message,
- Turn OFF the Reverse Video,
- Then move into the 25th line.

When I turned the Reverse Video OFF, the message just written was erased!. I resequenced the code to move the cursor BEFORE I turned the Reverse Video OFF, and got the desired results. Watch out for interaction! Most of the time there is usually a logical reason for the results you get, but sometimes it may not be readily apparent.

Probably the most aggravating problem I have encountered pertains to the I/O buffer the device driver uses to communicate to the screen. You must remember that all communication to the terminal is via this buffer and it does have a finite size. Once filled it may be dumped when you don't expect it. The dumping usually results in a new line command when you don't want it. When you 'PRINT' in BASIC with a ';' or use 'WRITE' in PASCAL you are putting characters into the buffer, but not generating actual output at this time. Unless you force the buffer to dump (with a carriage return) before it gets full you'll eventually have it dumped for you!! You really have to experience effect in operation before it can be fully appreciated. This problem does not occur with FORTRAN because the formatter always issues a new line command at the beginning of each WRITE statement which causes the buffer to always be emptied. I usually try to keep track of characters being placed in the buffer and issue a "transparent" PRINT or WRITELN to clear the buffer when the count gets near the buffer size. Note that the control characters,

FIGURE 3

FORTRAN	PASCAL
PROGRAM HUG	PROGRAM HUG;
BYTE ESC	VAR ROW,COL:INTEGER;
COMMON /AREA/ESC	
DATA ESC/27/	(--- Procedures here ---)
CALL CLRSCN	BEGIN
CALL MOVCUR(12,ROW,25,COL)	CLRSCN;
WRITE(1,10)ESC,ROW,COL	MOVCUR(12,25);
10 FORMAT(1H ,A1,'Y',2A1,	WRITE('Center of Screen');
- 'Center of Screen')	ENAB25;
CALL ENAB25	MOVCUR(25,15);
CALL MOVCUR(25,ROW,15,COL)	REVN;
CALL REVON	WRITELN('Welcome to 25th line');
WRITE(1,20)ESC,ROW,COL	REVOFF;
20 FORMAT(1H,A1,'Y',2A1,	MOVCUR(1,1);
- 'Welcome to 25th line')	WRITELN('Quit at the top!!');
CALL MOVCUR(1,ROW,1,COL)	END.
CALL REVOFF	
WRITE(1,30)ESC,ROW,COL	
30 FORMAT(1H ,A1,'Y',2A1,	
- 'Quit at the top!!')	
END	
(--- Subroutines here ---)	

escapes, etc. are characters in the buffer, and they must be counted also. Some experimentation may be necessary to determine the approximate buffer size for your language and particular hardware situation. You'll find out fairly quickly if you miss the size on the short side! By the way this is not a problem unique to just micros. A co-worker who is working with a DEC 20 (36 bit 'mini'), just encountered this problem.

FINAL WORDS

I hope this article provides some enlightenment to those of you working with PASCAL and FORTRAN. Nothing is more pleasing than to see a video screen "paint" a menu, or data displayed on the screen without the usual "bottom-up" scroll. In addition to the slick, professional appearance, it is also highly efficient and the fastest the screen can be formatted short of coding the same sequences in Assembler.

The example in Figure 3 is a very simple implementation of the basics but it demonstrates a technique that can be utilized with a little effort to provide a very powerful programming tool limited only by your imagination. I try to define and code functions which minimize mainline code, letting the sub-routines or procedures, depending on the language, do all the work. Its quite simple to create your own libraries to perform a variety of functions and standard CRT "screens" to reduce the drudge work of a new project, letting you concentrate on the function of the design rather than tedious and mundane aspects associated with interfacing the machine with the Human user.

I have just begun to work with the "C" language and find its capabilities for performing tasks described in this article are similar to PASCAL. It should be quite simple to implement screen handling functions in a likewise manner in "C".

I almost hate to mention one other aspect of the techniques involved here but whether you realize it or not you've actually been exposed to a simple form of "Structured Programming"! Though I have tried to keep the content of this article from being overly technical or complex you have been exposed to a variety of advanced programming concepts! Strangely enough, most of these concepts amount to nothing more than simple ideas that have been compounded to provide some exotic program capabilities. Now, that really didn't hurt too much, did it?!

Fortran-80 is a TM of Microsoft
 HDOS is a registered TM of The Heath Company
 Lucidata P-8080 Pascal (c) Polybytes
 DEC 20 is a TM of Digital Equipment Corp.

PRIMERS FOR THE BEGINNER

GETTING STARTED WITH CP/M AND MBASIC

WITH PARTICULAR REFERENCE TO RANDOM FILES

Featuring a complete "menu driven" ready-to-run disk mail list program, program explanations, and complete tutorials, this package forms the perfect introduction to MBASIC programming under CP/M, and includes useful information for those new to the CP/M operating system (ZBASIC also supported). Included is a 56 page manual and a disk containing sample programs. Specify Heath/Zenith Computer model number, disk size and format—hard- or soft-sectored, single- or double-density, 5¼" or 8", and CP/M or ZDOS. \$25.00

GETTING STARTED WITH HDOS & ASSEMBLY LANGUAGE PROGRAMMING

A 36 page tutorial covering aspects of Assembly Language programming under HDOS. Provides significant information for HDOS which other manuals lack. \$15.00

PLEASE SEND CHECK OR MONEY ORDER TO:

WILLIAM N. CAMPBELL, M.D.
 855 Smithbridge Road
 Glen Mills, PA 19342
 (215) 459-3218

Dear REMark,

I received my H89 kit in November and completed assembly without any problems. I must congratulate Heath for its clear cut assembly instructions.. My H89 works like a charm. I wrote programs in MBASIC to do all my Finance and Household jobs I needed as well as type in a few Basic games.

However, I wanted a little more excitement than what I was getting with the games I programmed. So I ordered from HUG, "GALACTIC WARRIOR". I thoroughly enjoyed the WARRIOR software. I have been playing now for 2 months, and I have not gone past the fifth phase or scored more than 777 points. I have enjoyed every moment.

I do have a few hints for potential warriors. Attack the enemy head-on, this seems to confuse them and be patient, a lot of moving usually ends up with a sore finger and very little points. Congratulations again to Heath and Evryware for bringing 2 great products into the computer field.

CPT. James Lee Ross Jr.
Box 1131, 270 Signal Company
APO NY 09189

Dear Sir,

In the never-ending quest for higher and higher levels of automation I discovered your article on "Turnkey Operations" in issue 25. While putting Robert Conde's idea into effect I added a new wrinkle of my own.

Why not do away with the need for "SUB-MIT.COM" altogether? There is little enough disk space on my single-drive, single-density system, so why not simplify file management and do the whole job from within the BASIC program?

All you need to know is the format of the temporary file "\$\$\$\$.SUB" created by "SUB-MIT.COM". This is easily accomplished by a little research using DDT, plus a few abortive experiments. My first test run resulted in the commands being executed in reverse order. "Eureka," I shouted, "the system picks off the commands starting from the end of the file and working forwards with each warm boot." It worked, the results in both MBASIC and CBASIC are included for the interest of your readers.

Derek Richards
176 Woodside Drive
St. Catharines, Ont L2T 1X6

```

1 ' Submit a Command File under MBASIC for Execution by CCP
2 '   The system will look for a file "$$$$.SUB" on each warm boot.
3 '   Record length is 128 bytes, records are in reverse order.
4 '   Record format: byte 1       - command length
5 '                   followed by - the command
6 '                   terminated by - chr$(0) and "$"
7 '   The temporary file will be deleted by the system
8 '   when all the commands have been executed.
9 '

```

```

10 OPEN "R",#1,"$$$$.SUB" File name in caps
20 FIELD #1,128 AS CMD$
30 INPUT "Number of commands to be entered:";N
40 FOR I = 1 TO N
45   ; Input could just as easily come from a string array
46   ;   or an ASCII file of commands created by a
47   ;   text editor.
50   LINE INPUT "Command:";ANS$
60   LSET CMD$=CHR$(LEN(ANS$))+ANS$+CHR$(0)+"$"
70   PUT #1,N-I+1' write out in reverse order
80   NEXT
90 CLOSE 1
100 SYSTEM
111

```

```

rem - Submit a Command File under CBASIC for Execution by CCP
rem -   The system will look for a file "$$$$.SUB" on each
rem -   warm boot. The record length is 128 bytes, records
rem -   are in reverse order.
rem -   Record format: byte 1       - command length
rem -                   followed by - the command
rem -                   terminated by - chr$(0) and "$"
rem -   The temporary file will be deleted by the system
rem -   when all commands have been executed.

```

```

create "$$$$.SUB" rec1 128 as 1
input "Number of commands to be entered:";n%
for i%=1 to n%
    rem - input could come from file, or table.
    input "Commands:";line cmd$
    rem - print recs in reverse order, first - last
    rem - rec format: length of cmd, cmd, zero, "$"
    print using "&";#1,n%-i%+1;chr$(len(cmd$))+cmd$+chr$(0)+"$"
next
close 1: stop

```

Dear HUG,

Having recently upgraded my H11 to RT11-V4 and added the DMA drive controller I started to encounter erratic results. Memory traps, ill. inst. traps, etc.

After a couple of calls around the country I learned that the BIAKI line, Pin 2 on IC-2 needs to be terminated in the same manner as is the BIAKO line, Pin 3 on IC-9 on the later H-11-5 serial boards.

If someone has a very early board, then their schematic will not show any termination on either line, although there is a single resistor

on the foil side of the board on IC-9. The termination I used is to install two resistors from Pin 2, IC-2. One is a 680 ohm to ground and the second is a 330 ohm resistor to the +5V line. I also removed the terminating resistor on the foil side of IC-9 (not shown on schematic) and installed the same two values on Pin 3 of IC-9 as used on IC-2.

I hope this information is helpful to someone.

Rick Hayward
P. O. Box 23248
Harahan, LA 70183

Dear HUG,

I've found a bug in my copy of the Grade and Score Keeping program HUG P/N 885-1091. I haven't seen a correction in REMark so I assume that your disks may still have the problem.

One of the program options is deletion of a student from a class roster. This is where I have found the bug. If the roster contains Smith and Smithson and you wish to delete Smithson you respond with "n" to line 1300 when line 1290 prints Smith. Line 2040 then assigns "N" as A\$. Line 1320 puts you back in the loop at 1270 and you get into trouble at 1260 because A\$ is now "N" instead of the student name that was assigned as A\$ when line 1220 put you into the subroutine at 2080. The loop finishes with no matches and line 1280 tells you you're out of luck and can't delete the student from the class. I have made two changes which eliminate the bug. The lines are from MENU.BAS. The corrections are:

```
ADD line 1245 R$=LEFT$(A$,4)
CHANGE line 1260 IF LEFT$(S$(I,1),4)=R$
THEN 1290
```

Bill Sparrow
1421 West Huron
Ann Arbor, MI 48103

Dear HUG,

I'm a great one for trying out programs that appear in REMark because they are cheap and most of the time very, very clever. Many times, however, they contain errors! I know how this can happen. I have encountered problems with the MUSIC program which appears in Issue #38 of REMark.

After I very carefully typed the source in and assembled it, to my surprise, it would not work. Well, I really wasn't that surprised. So, I started to dig as any good programmer would.

The first error I found made the program "hang-up" when it didn't find the [file]. This is caused by a jump to a system call instead of a call. This happens in the CHR\$OUT routine.

```
CHR$OUT EQU $
MOVE,A
MVIC,02H
JMP 05H <— This is the one
Replace with these two:
```

```
CALL 05H
RET
```

The next error found was that the JZ NO\$FILE does not jump. The jump address is wrong,

```
NO$FILE CONPRT <— This should be
CON$PRT
```

The last item is not an error, but the reader is apt to make one. So, I might as well tell you about it. The MBSX subroutine contains a JMP 110014. Now anyone with a magnifying glass will be able to tell you that this is really JMP LL0014. Also be sure the label matches the JMP.

The last major problem is that your music file name "MUSIC" has an extension of .SCR. For some reason, the author loaded the FCB with this extension. The file must match it or after the preceding fixes are made the message "File does not exist." will appear.

Two last items; if you try to change the breath value once you start, you get a solid tone, and always preceded a dotted note with at least the weight (Whole, Half, etc.).

Despite these problems, this program has been a lot of fun to play with. Keep the good programs coming, the variety of ideas are very educational.

Douglas Shenk
RD #2 Box 161
Millerstown, PA 17062

Dear Douglas, Thank you for your comments on my article. The article did not clearly state that the program was written for CP/M ASM which ignores the '\$' character imbedded in labels and doesn't care about upper/lower case. I should have cleaned up the program so that it did not depend on those features. I also neglected to mention the required '.SCR' extension. The change of JMP to a CALL and RET is totally unnecessary though it certainly doesn't hurt. There have been many complaints about the quality of the listing. The lowercase 'l'(L) looks a lot like a '1'(one) and some 8's look like 3's also.

Frank T. Clark

Dear HUG,

I am writing to commend Roy Reichert on his thorough approach to terminal input in "Keyboard Polling from BASIC". The article is as well written and as educational as I can imagine.

I would also like to point out several errors which I found and would like to suggest changes which might serve to improve the routine. In lines 130 and 530, " " was omitted after IF A\$< and A\$= respectively. In addition, the value of M in line 510 should be 56688 for HDOS 2.0 and 56691 for HDOS 1.6.

Improvement can be made by incorporating the HIGHDAT. address and pointer offsets described in "Using the HDOS Type-ahead Buffer in BASIC" (Issue 19, pg. 18). These

allow the keyboard input routine to be version and memory size independent and may permit reliable operation in the HDOS stand-alone mode. The address of the queue tail pointer (low orderbyte) is obtained by offsetting the contents of HIGHDAT, (decimal address 8422) by 6 and peeking the location establishes the pointer value. This is implemented by changing line 510 to:

```
510 M = 6 + Peek (8422) + 256 * Peek (8423)
```

Hopefully this information will aid others in using the routine.

Bill Theisen
17 St. Albans Ave.
Madison, WI 53714

Dear HUG,

When using the H-89 computer with CP/M, Magic Wand, and the Epson MX-80 Printer, getting the pages of the second copy to be properly numbered is not explained very well in the Magic Wand manual. I have laboriously determined this procedure by trial and error.

When in the edit mode, at the end of the last page of the text, disable the page numbering by embedding the following commands preceded by a backslash:

```
\HEAD0
\NP
```

Re-start page numbering at the beginning of the succeeding copies by following the above HEAD0 and NP commands with HEAD1, RIGHT, Page, and %PAGE preceded by backslashes. %PAGE should also be followed by a back slash.

```
\HEAD1
\RIGHT\Page\%PAGE\
```

Re-set the page number counter in the SETUP commands at the beginning of the file by following the TEXT command with the following command preceded by a backslash:

```
\PG1
```

William L. Torow
1229 National Ave.
Oshkosh, WI 54901

Dear Sirs,

Please publish the following offer:

After a great deal of difficulty, I now have Ward Christensen's MODEM-7 program working on the Heath/Zenith H-89. This is a public domain terminal program that allows communications and file transfers between two computers equipped with the program. One of them may be an unattended bulletin

board system.

This is one of the best modem programs available, but it is extremely difficult for anyone, let alone beginners, to get configured and working for a new system. I am therefore offering to provide a copy already patched for the Heath system to anyone who will send me a disk and mailer and \$5 to cover return postage and handling. I can provide 5 1/4" hard or soft sector or 8" standard copies. This program is for CP/M only, not HDOS.

This free program outperforms many selling for hundreds of dollars. It can send an entire disk, including .COM files, with one command in batch mode. Error checking and retransmission is provided. I demonstrate this by whistling into the phone while a file is being sent. The program simply resends the trashed sector and goes on. I hope to save other Heath/Zenith owners the hassles I had getting this fine program to work.

Paul Pennington
2912 Palmetto Drive
Martinez, GA 30907

Dear Sirs,

I have been using WordStar, an excellent product from Micropro, for half a year now and I am generally very pleased with it. I chose WordStar over several other packages locally available, mainly because of its ability to display page breaks, which is a necessity for typing any paper where footnotes must be printed at the end of a page.

In spite of all its good points, I have had several problems and frustrations with it, and hope that I can help someone else so that they will not have problems that I had.

Some of the default parameters set up in WordStar are less than satisfactory. The print parameters, especially, are definitely less than desirable, as they are far from standard. Most colleges and universities demand a one inch margin all around an 8 1/2 by 11 sheet of typed paper, but Micropro in all their wisdom set a default 8 space left margin offset together with default 65 space margin ruler, leaving a 12 space right margin which appears rather lopsided on paper. The top and bottom margins are also only defaulted to approximately 5/8 of an inch. Rather than having page numbers printed at the top with a default to off, they default on at the bottom of the page. This means that if you want to have numbers at the top, and for any reason must change the page number with the ".pn x" command, you must not forget to switch it off again immediately with the ".op" command.

After quite a period of frustration, I found a

method to help alleviate this problem in a fairly simple fashion. You simply set up a file with all the standard parameters which you will normally use, (margins, page numbering, embedded ruler line, etc.) and write it to disk in a unique file which you at all times have on your working data or master disk. When you start a new file, you merely have to read this file in before entering any other text, and all parameters are re-set.

One problem which I have encountered, and may just have to live with, is WordStar's method of underlining. Only words themselves are underlined, and not the gaps between them, which means that if I need a solid underline, I must do it by hand.

Another problem I found was the use of the four user definable commands in the "P" menu. The manual simply tells you that it can be done, and tells you how to enter data into the program through the INSTALL program, but not what data to use. Micropro refused to answer my questions since Heath had modified WordStar. Quite accidentally, I came across the answer to this problem, and would like to pass it on.

After finishing the first part of INSTALL, you are asked if modifications are complete. On answering "no", the program prompts for the address to change. This can be either a label or a hexadecimal address in WordStar. In order to modify the four P menu user modifiable commands, you enter one of the four labels USER1: through USER4: (commands Q,W,E,R). Until this point, the information in the manual is complete; from here on, they fail to tell you what information to enter. You must now determine what you wish to place into these locations, as these are commands from my printer, is the command for underlining, which is "ESC R". This is a 2 digit command, and as a result, this becomes a 3 character entry, since the first one entered lets WordStar know how many control characters are to follow. After the label "USER1:" is entered, Install displays its address in hexadecimal, displays the value of its contents (00 if unused), and prompts for a new value. In this case, since my printer command is 2 characters long, my first entry is "2". Install then prompts for a new location, and after converting the data to hex, you enter the location given you in the first entry plus 1 (USER1: loc+1), and the data (1b (hex) for Esc). Do this again for the second character of the printer command (USER1: loc+2, data (R) 52 hex). After this entry, instead of another location, simply enter a "0" (zero) and a carriage return, and the patching is completed. INSTALL returns you to the menu, and you write the modified program back to disk.

I also discovered that you may not change the name of WordStar by use of the CP/M "REN" command. WordStar contains its own name, which on the original program is "WS.COM", internally for use of the "Run" command on the main menu. If you run another program from WordStar, WordStar reserves some memory and stores all set parameters there, plus a sort of bootstrap loader for WordStar under the name stored internally. Upon completion of a program, this bootstrap automatically tries to reload WordStar. If no program by this name exists on disk, the system crashes. The only way to legally change the name of WordStar to any other name that you may prefer, is through the Install program, as listed in the manual.

Hopefully these hints will help someone else to overcome some of the problems that I encountered, especially as a beginner with this program.

J. Allan Bartel
P. O. Box 1241
Steinbach, Man.
Canada, ROA 2A0

Dear HUG,

I have seen other letters explaining the interconnecting of the Heathkit computers with various printers. I also had a problem in getting my Prowriter (C. loth 8510A) printer to interface with my H-89. So for those out there who are having similar difficulties, here is the key.

H-89	Prowriter
1 <----->	1 system ground
3 <----->	3 serial data
4 <----->	14 handshaking
7 <----->	7 signal ground
20 <----->	20 terminal ready

I have found that this printer is at least as good as the Epson MX80 F/T. It has dot addressable graphics and 3 standard prints: Pica (10 CPI), Elite (12 CPI), and Compressed (17 CPS). It features descendings and will support super and subscripts.

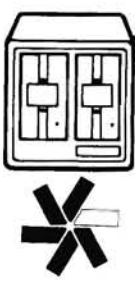
One other feature I have found to be very nice is the ability to set the left hand margin. You can then just PIP a file to the printer and it is moved over so you can punch holes without destroying any of the print.

When using it under CP/M you will need to configur in order to change the standard Heath setting for the printer from DTR to RTS and the LOW to HIGH for printer ready.

Hopefully this will help someone to complete their system.

Ralph R. Bollinger, Jr.
1244-B Hermes Road
Redstone Arsenal, AL 35808





Heath Related Products



Tom Huber
Related Products Editor

NOTE: The following information was gathered from vendors' material. The products have not been tested nor are they endorsed by HUG.

Enhanced Version of ZenCalc

The Software Toolworks' is now offering an enhanced version (3.0) of ZenCalc, their spreadsheet program for Heath/Zenith computers and terminals and, at the same time, reduced the price. Although a bit late for this tax season, unless you filed an extension, the package now includes templates for computing the 1982 federal income tax forms. A numeric keypad mode has been added along with a table lookup function. The package also includes a loan amortization spreadsheet for computing mortgage payments and schedules.

Vendor: The Software Toolworks
15233 Ventura Blvd. Ste. 1118
Sherman Oaks, CA 91403
(213) 986-4885

Price: \$49.95 (Version 3.0)
(Update) \$10.00 & \$2.00 S&H for registered owners of earlier versions.

Dual and Single 8" Disk Drive Line

The model I-8480 is a dual drive, double-sided thinline 8" floppy disk system that is plug compatible with the Heath/Zenith H/Z-100 Computer series. The I-8480 features side-by-side horizontal mounting, power supply, 50-pin standard interface cable, and a formatted capacity of 2.4 megabytes (1.2 megabytes/disk). Other configurations are also available and includes an I-8481 single drive, double-sided thinline 8" disk system; an I-8482 dual drive, single-sided thinline 8" disk system; and an I-8483 single drive, single-sided thinline 8" disk system. Each system comes fully assembled, with documentation, and includes a 90 day manufacturer's warranty.

Vendor: Data Compass
Peripheral Products Division
2730 Regal Park Drive
Anaheim, CA 92806
(714) 630-7450

Prices: I-8480 — \$1990.00
I-8481 — \$1300.00
I-8482 — \$1700.00
I-8483 — \$1130.00

Retail Point of Sale Software

Point of Sale software allows the use of the computer as a complete inventory control and cash register system. XtraSoft, Inc., has announced a series of Inventory/Point of Sale software for the H/Z-89, Z-90, and H/Z-100 Computers running under CP/M. Also available are electronic cash drawers for use with RS-232 ports and the software. Contact the vendor, specifying your system, for full details. Dealer inquiries are welcome.

Vendor: XtraSoft, Inc.
5427 Bardstown Rd.
Louisville, KY 40291

Fabric Ribbon Renewer

Le Ribbonizer(tm) is a motor-driven fabric ribbon reinker that will result in a significant cost reduction for fabric ribbons. Ink and instructions are included. Typically, ribbons will take from fifteen to twenty minutes to be completely reinked. Most ribbon types can be renewed many times (about ten times for dot matrix printers and 20 times for letter quality printers). Specify printer brand and model (such as Epson MX-80) when ordering. For more information, send a large SASE envelope to the vendor. The vendor can also supply a reload service for used cartridges for many printers.

Vendor: Le Ribbonizer(TM)
Box 1727
Redlands, CA 92373

REMREF II — Update for REMREF

REMREF II is an ASCII text file containing Author, Title, and Keyword information for all articles appearing the first 36 issues of REMark. By using string search capability of many text editors and most word processors, you can quickly locate a specific article or articles by subject or author.

Vendor: Generic Software
POB 790
Marquette, MI 49855
(906) 475-7151

Price: \$24.95 & \$2.00 S&H
(MI residents add 4% sales tax). \$13.00 & original disk for update to registered owners of REMREF.

The Complete Book of Random Access & Data File Programming

This new book by H.J. Muller is the second volume in a series that started with "DOS Randon Access & BASIC File Handling." Subjects covered include blocking and de-blocking, The Shell-Metzner sort, record pointers, record space recovery, alpha-index record retrieval, Mach/BASIC sort, linked lists, sort/merge, multi-key file organization and retrieval, relational data base programming, hash code file handling, and span blocking. The techniques are illustrated in Microsoft BASIC, starting with a simple program and building into the more complex routines needed for the advanced techniques being illustrated. 200 pages, self-instructional tutorial style. Supporting program disks are available for a wide variety of machines, including TRS-80 Models I, II, & III, CP/M 8-inch, and IBM Personal Computers.

Vendor: D.S.C., Publishing
P.O. Box 769, Hayestown Rd.
Danbury, CT 06810

Price: Book: \$29.95 + \$2.00 S&H
Disks: \$49.95 (Specify type)

Forth Computer Language

FORTH-79 is a complete software system, meeting all the provisions of the FORTH-79 standard, and providing an interactive environment allowing for rapid program development. The high-speed compiled code makes it suitable for real-time applications. MicroMotion's FORTH-79 (version 2.0 for Z-80 CP/M systems) base system includes a screen editor, macroassembler, string package, 3-bit integer arithmetic, and a 200 page tutorial/reference manual. Prices run from \$99.95 to \$139.95. Contact the vendor for pricing on your particular system (specify).

Vendor: MicroMotion
12077 Wilshire Blvd., #506
Los Angeles, CA 90025
(213) 821-4340

**FIG Membership Includes
Subscription of "FORTH Dimensions"**

The FORTH Interest Group (FIG) publishes "FORTH Dimensions," a non-profit publication about FIG and the FORTH Computer Language, as part of the FIG membership (\$15.00 USA and \$27.00 foreign). For further information, contact FIG.

Vendor: FORTH Interest Group
P.O. Box 1105
San Carlos, CA 94070
(415) 962-8653

**Agricultural and
Farm Management Programs**

Naso Computer Systems is a hardware and software vendor that specializes in CP/M and Z-DOS software for the Agricultural and Farming industry. Examples of programs (many of which may be customized on order) include AG-Accountant II, an integrated single entry Farm Accounting and Payroll system, menu driven for easy use, password protection, and proven performance in the field. Also available is a Crop Data Management system that is designed for pest control and/or production activities and costs and can be used for all field, row, and tree crops. Additional software includes Farm Management Diary, Orchard/Vineyard Plotter, Fruit Crop Sorter (for best grower value), Farm Equipment Maintenance, Rice Tissue Analyst and Grapher, and many others. Some software takes advantage of the Z-100 color graphics. For full details or custom requirements, contact the vendor.

Vendor: Naso Computer Systems
1115 Gray Avenue
Yuba City, CA 95991
(916) 673-6276

HA8-3/HA89-3 Graphics Board Software

SKETCH.ABS is a graphics editor for the HA8-3 and HA89-3 color graphics boards for the H-8 and H/Z-89 computers. SKETCH allows the user to define sprites and move them; draw and erase lines; and define, scale, rotate, translate, duplicate, and move polygons. A circle generator is also included. Pictures created with SKETCH may be saved and retrieved for use in other programs. Available in HDOS or CP/M 5.25-inch hard sectored formats, it requires a minimum of 48K RAM and the HA8-3, HA89-3, or Entertainer color graphics boards.

THREED is a three dimensional file handler and when used with SLIDES 2.0 (included), will allow the user to display sequences of pictures created with the package. The package offers the capability to (for example) create an imaginary city and then view it from any perspective—above, in, or below it. The program features full hidden line and surface removal; 256 x 192 pixel resolution; object location, and duplication; and complex scenario creation (up to 150 polygons and 500 vertices).

Both programs include full documentation, instructions and examples; require the HA8-3, HA89-3, or Entertainer color graphics board; and 48K RAM. Available in HDOS or CP/M 5.25-inch formats (specify when ordering).

Vendor: Colorworks
5337 E. Bellevue Ave.
Tucson, AZ 85712
(602) 325-0096

Prices: SKETCH—\$35.00
THREED—\$50.00

**Version of Adventure
is Endorsed by
Crowther and Woods**

The Software Toolworks announced that Will Crowther and Don Woods, creators of the original Adventure game, have endorsed Toolworks' version of the game. The creative team said that this version "includes all the features of our game", and, "its additions and enhancements are faithful to the spirit of the original". This version contains additional rooms and treasures, as well as an expanded end game and players who attain the maximum score may send for a free "Certificate of Wizardness" bearing Crowther and Woods' signatures.

Vendor: The Software Toolworks
15233 Ventura Blv. Ste. 1118
Sherman Oaks, CA 91403
(213) 986-4885

Price: \$19.95 + \$2.00 S&H.

**Join with other
HUGgies
at the
1983 NATIONAL
HUG CONFERENCE II**

Changing your address? Be sure and let us know since the software catalog and REMark are mailed bulk rate and it is not forwarded or returned.

----- CUT ALONG THIS LINE -----

HUG MEMBERSHIP RENEWAL FORM

REMEMBER — ENCLOSE CHECK OR MONEY ORDER

CHECK THE APPROPRIATE BOX AND RETURN TO HUG

When was the last time you renewed?

Check your ID card for your expiration date.

IS THE INFORMATION ON THE REVERSE SIDE CORRECT?
IF NOT FILL IN BELOW.

Name _____

Address _____

City-State _____

Zip _____

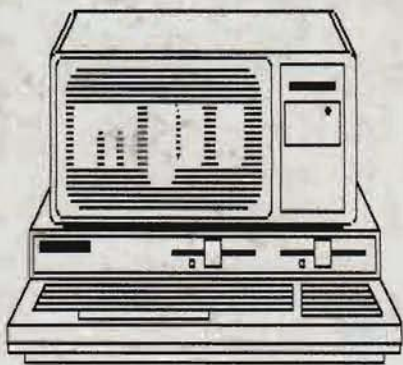
NEW MEMBERSHIP
FEE IS:

RENEWAL RATES

US DOMESTIC	\$15 <input type="checkbox"/>	\$18 <input type="checkbox"/>
CANADA	\$17 <input type="checkbox"/>	US FUNDS \$20 <input type="checkbox"/>
INTERNAT'L*	\$22 <input type="checkbox"/>	US FUNDS \$28 <input type="checkbox"/>

* Membership in England, France, Germany, Belgium, Holland, Sweden and Switzerland is acquired through the local distributor at the prevailing rate.

PRINTMATE™ AP-PAK'S™



Print in:
SPECIAL FONTS
SPECIAL FONTS
SPECIAL FONTS
special fonts
Special fonts

PrintMate GOES GRAPHIC WITH H/Z 100!

Picture this—your logos, graphs and fancy print fonts on paper with a PrintMate. Our exclusive AP-PAK lets you print what you picture. See and believe at your Heath and Zenith dealer today.



PrintMate 99

PrintMate 150

Micro Peripherals, Inc., 4426 South Century Drive,
Salt Lake City, Utah 84107 **1-800-821-8848**



H/Z 100 is a registered trademark of the Heath Company.



Heath
Users'
Group

Hilltop Road
Saint Joseph, Michigan 49085

BULK RATE
U.S. Postage
PAID
Heath Users' Group

Volume 4, Issue 5

POSTMASTER: If undeliverable,
please do not return.

885-2040