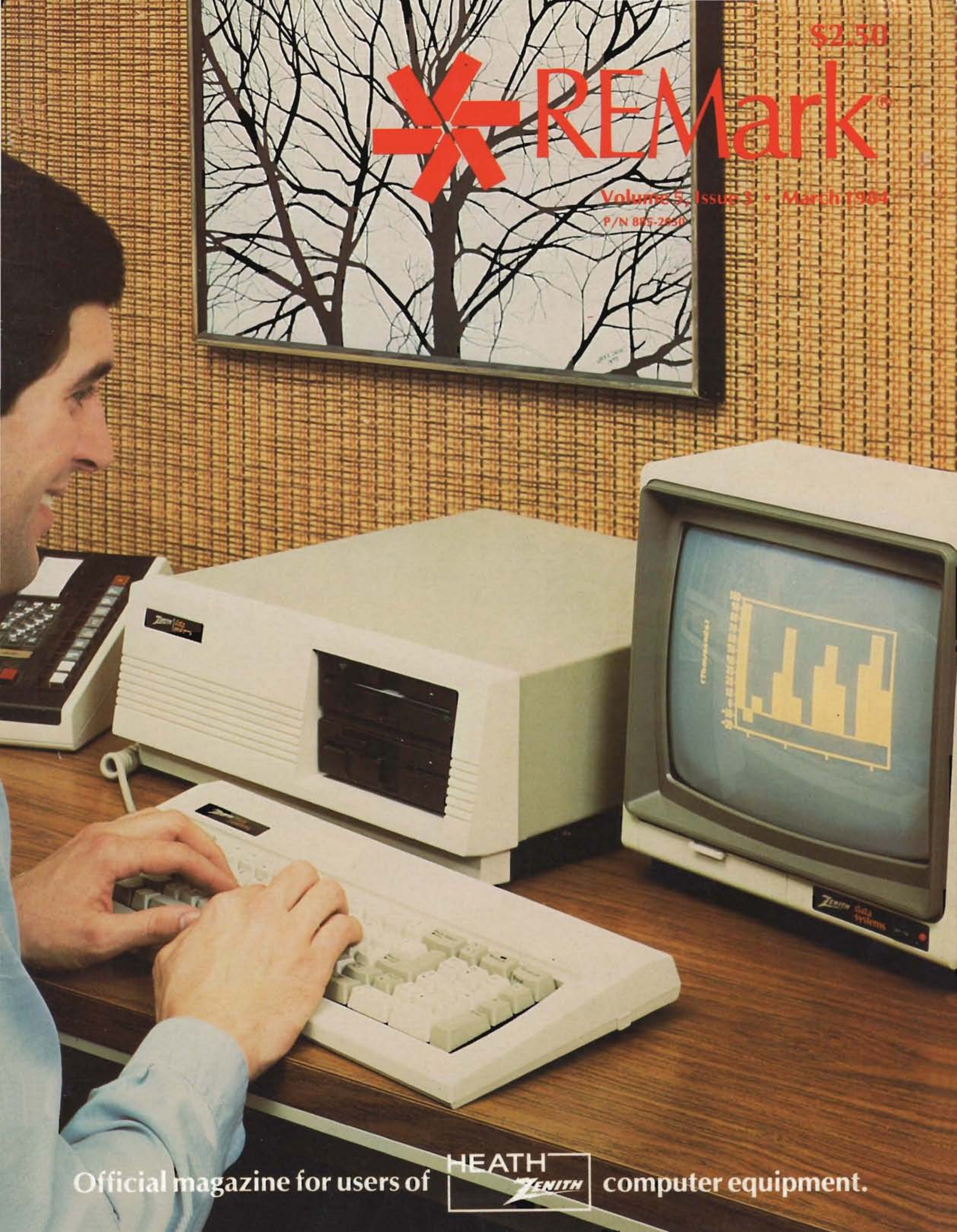
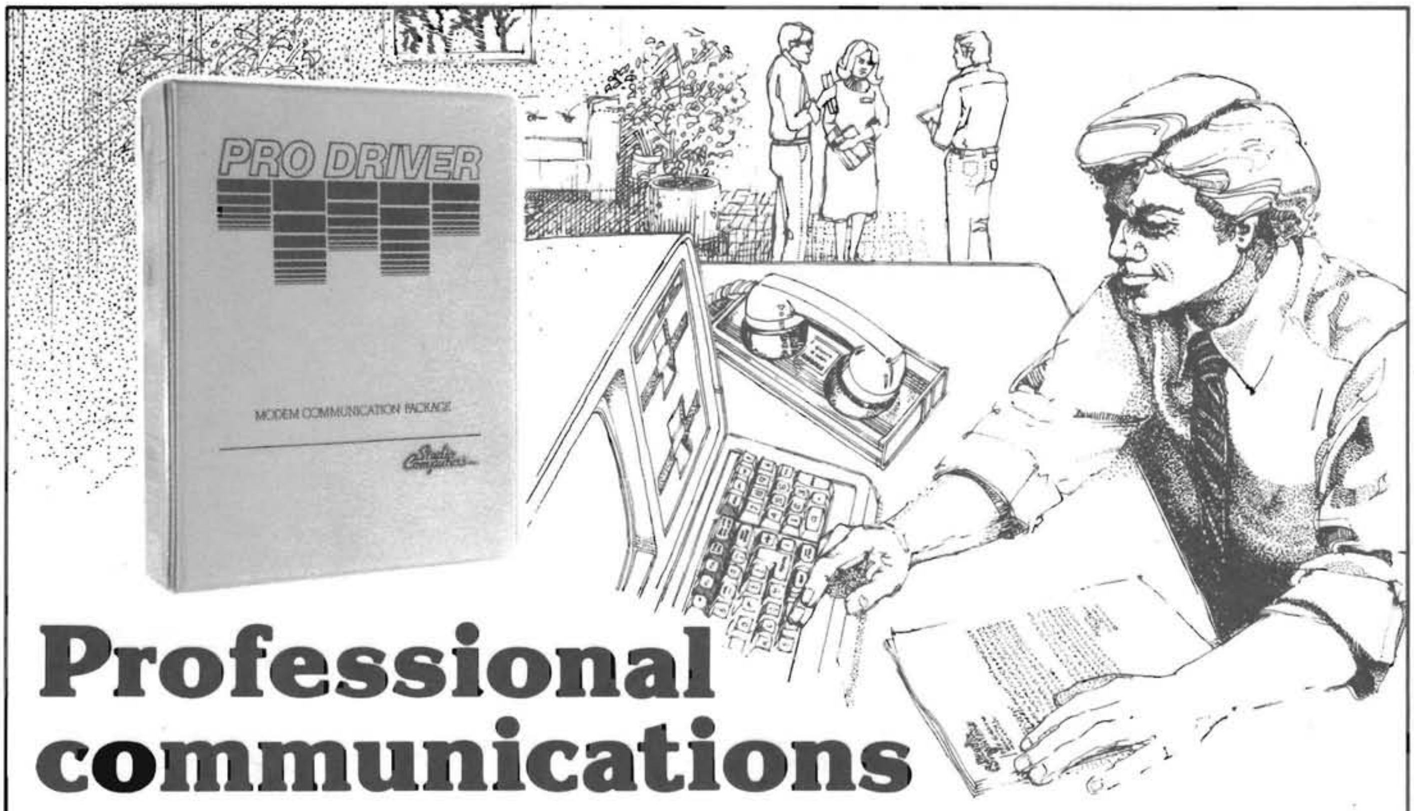
$2.50

REMark®

Volume 5, Issue 3 • March 1984
P/N 885-2050



Official magazine for users of HEATH ZENITH computer equipment.



Professional communications

Pro Driver is a new professional communications package for Z100 computers using the ZDOS operating system. It allows you to 'talk' to remote computers using any modem or directly to another computer through a 'null modem cable', and allows transmitting and receiving of both ASCII and binary files. Pro Driver lets you perform operating system commands such as renaming files, deleting files, directory listings or resetting disks. The clear, uncluttered menu-driven displays make using Pro Driver a pleasure.

Here are just some of Pro Driver's features:

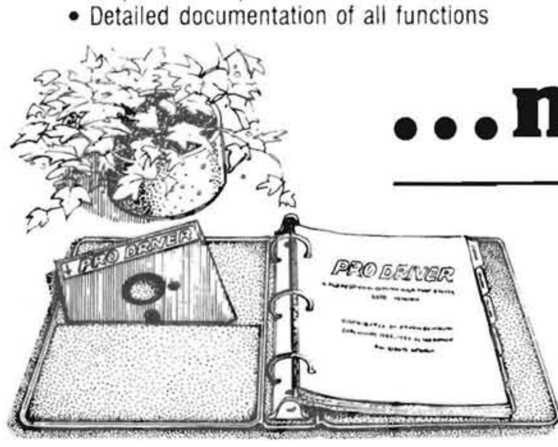
- Menu-driven request and control options
- Direct support for Hayes Smartmodems
- Report Generator for detailed tracking of calls
- XMODEM, Compuserve B and X/ON X/OFF protocols
- 32 User-Definable Automatic Logon Sequences
- Printer Trace function for hardcopy of all data
- User-Definable Control Sequences within Pro Driver
- Selectable Baud rates from 45 up to 19,200
- Configurable entry and exit foreground/background colors, cursor modes and keyclick options
- Help files for quick assistance on Pro Driver functions
- Detailed documentation of all functions

Pro Driver is a professionally designed package from top to bottom. Simple to operate even for the novice, yet with many features an experienced computer user would demand. For example, the 'Reporter' feature not only logs all calls made and the length of time per call, but gives you the option of outputting your report to a printer, a disk file or the video console. And, you can have that report sorted by telephone number or dates! Imagine being able to quickly check at the end of a week, month or even year just how long you spent connected to any specific time-sharing or bulletin board service. A great way to verify the connect-time bills they send to you!

Pro Driver comes with over 180 pages of documentation and pictorial screen displays, but don't let that scare you. Pro Driver is easy to operate! All major sections are organized with tab dividers for quick locating. And a complete index is also included.

The price? Just $49 for the entire package with all of the above features and more. Pro Driver may be purchased at our retail store location, by mail (please add $2 for shipping) or check your local Heathkit Electronic Center for availability.

Pro Driver — the "professional" communication interface for your Z100 computer.



...made simple!

Studio Computers inc.

999 South Adams
Birmingham, MI 48011
Phone (313)-645-5365

ZENITH data systems

Call or write for our complete catalog, or additional information on Pro Driver.

Heath/Zenith Users' Group

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 sparately without membership. The following rates apply.

	U.S. Domestic	Canada & Mexico	International
Initial	$20	$22*	$30*
Renewal	$17	$19*	$24*

*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 availble 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
616-982-3463

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 © 1984, Heath Users' Group

REMark®

Volume 5, Issue 3 • March 1984

on the stack

ON THE COVER: Pictured is the new Z-100 Personal Computer, Model 150. See page 5 for further details.

Great Zenith Software Bargains!

Video*Professor: Programming Instruction Set

The popular Video*Professor line of products use the computer itself to teach, not just to supplement a textbook. Courses include self-study aids such as quizzes and examples, and you learn on the computer. They are available under ZDOS or CP/M85(soft sector) on the Z100 and under HDOS and CP/M(hard sector) on H/Z89 systems. The Video*Professor price for each course is $29.95...but REmark readers can buy our programming instruction series of courses at the following great prices!

2 COURSES FOR $44.95, A $14.95 OR 25% SAVINGS
3 COURSES FOR $64.35, A $25.50 OR 28% SAVINGS

Courses available:	Assembler I Programming
	MBasic I Programming
	Structured Programming

Business Support Mini-Pak

Here are two quality products that can help the Zenith owner who is using his computer to support his business. **Multiple Location Perpetual Inventory System (MLPI)** has features found only in other more expensive products. Many reports are available; including usage, reorder, and cost figures. MLPI is available under ZDOS, CP/M-80, and CP/M-85. The regular low price is $89.95. We also offer the **Office Wizard** product, an array of 5 handy business utilities which includes a mailing label system, file manager, automatic letter generator and a business graphics generator. The normal price for Office*Wizard is $59.95.

NOW...GET BOTH SYSTEMS FOR $104.90 AND SAVE $45.00 OR 30%--ORDER TODAY!!!

ALSO...BUY ANY SINGLE PRODUCT NOTED ABOVE FOR 15% UNDER THE LIST PRICES SHOWN!!

Communications Trio: A Three-In-One Value

Communications capabilities are becoming very important to today's micro-computer user. You want to be able to reach bulletin boards, time sharing services, friends' computers, and even your own computer from a remote location. Such functions can be readily accomplished with the following fine products: **ZED-COMM**, number one modem control program available for Z100s, has many advanced capabilities such as single keystroke commands, XON/XOFF, host/terminal modes of operations, and multiple set options. ZDOS is the operating system used, and the regular price for ZED-COMM is $49.95. **REM-CON**, a fine product which enables you to use your computer from remote locales, runs under CP/M on H/Z89s and ZDOS or CP/M on the Z100. REM-CON can be accessed via any ASCII terminal which attaches to your computer via telephone modems, and it includes password protection. REM-CON's normal price is $44.95. .....Ever wish you could move programs and files between H89s and Z100s? Now you can with **Z6000**, the transfer program that uses a direct cable connection in moving files from CP/M-80 to ZDOS or CP/M-85. It can accept unlimited file sizes and wildcard designations ...and regularly sells for $34.95. **BUT NOW...** REmark readers can get all three products for the REmarkably low price of **$94.95...a savings of $34.90 or 27%!!!**

ZMAG: The Software Library

A ZMAG subscription gives you a magazine and a diskette once every 6 weeks for a year. Issues contain 10 to 15 good programs for many uses, ...business, personal, and fun. The $147 price per current subscription gives you a cost of less than **$2.00 per program** ...or instead, you can buy single issues for **just $30.00**..a great buy so order now!

Husker Systems of Nebraska, Inc. Fine Computer Products and Services

Current Date__________ Payment Method: Cash____ VISA____ MasterCard____

Credit Card Number__________ Expiration Date______ Check For New Catalog ______

QUANTITY	ITEM	SYSTEM/FORMAT	UNIT PRICE	TOTAL
	*Nebraska Orders Must Include 5.5% Sales Tax.		TOTAL*	

Name____________________

Address____________________

City__________ State______ Zip________

Send Orders To: Husker Systems of Nebraska, Inc.
6657 Redick Avenue
Omaha, NE 68152

Or Call Toll-Free: (800) 835-7427 extension 277

Zenith Data Systems Announces The Z-100 Personal Computer Series

Gerry Kabelman
Zenith Data Systems

Zenith Data Systems proudly announces a new Z-100 series of computers, which is compatible with virtually all IBM-compatible software. The new Zenith Data System's personal computers are being offered in two basic models. The first is the desktop system and the second is the transportable (shown above) with a built-in amber 9-inch CRT. Both systems have the same enhanced features and improved price/performance over the IBM personal computer.

The standard configuration includes the 8088 microprocessor, 128K bytes of random access memory, video outputs for both RGB color and monochrome, an improved keyboard, two serial ports, a parallel port, one or two disk drives, and still four expansion slots left after all that.

The Z-100 PC also has the same electrical structure in its expansion bus.

The Z-100 PC processor runs at 5 MHz and has a standard 128K bytes of RAM, expandable to 320K bytes on the included memory board. Another memory board may be added to increase the memory to 640K bytes. Each memory board also contains a parallel port. The floppy disk controller card has two serial ports. The video and CPU boards are also separate with LED status indicators for easy troubleshooting.

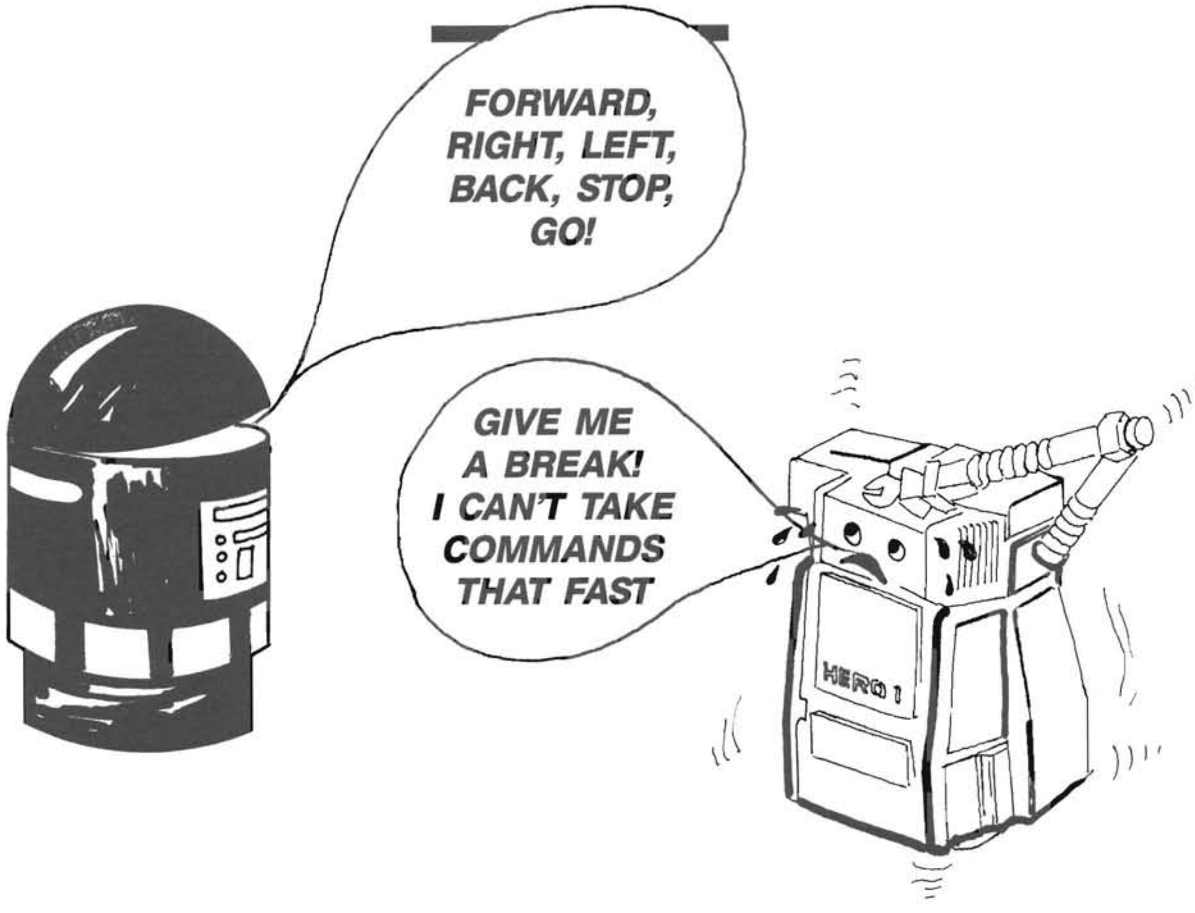
The Z-100 PC is based on a motherboard design with a total of eight slots and a separate highly reliable switching power supply. Four of the slots are available for expansion after already expansioning the memory to 320K bytes. Also the CPU card has an IC socket to allow the addition of the 8087 math co- processor.

The Z-100 PC has automatic bootup with built-in powerup diagnostics, menu driven ROM based diagnostics, a detached keyboard.

The Zenith Data Systems personal computers are missing the high resolution video of the Z-100 dual processor computers. Other advantages of Zenith's dual processor family of computers are support for 8-bit CP/M and 16-bit MS-DOS software, an attached Selectric-style keyboard, S-100 bus and built-in eight inch drive support. Two new hardware boards were recently announced for the Z-100 which are the networking cards and the 8087 co-processor. The new 8087 co-processor board is a piggy back board, for the 8088, leaving S-100 bus free for other boards.

Zenith Data Systems has also announced the availability of Microsoft's Windows, an operating system environment for multiple applications, for all Z-100 computers. Also to be available shortly from Zenith are concurrent and multi-user, multi-tasking operating systems, the UCSD p-System, SuperCalc 3, GrafTalk, and updates to WordStar, MailMerge, Peachtree accounting, Condor rDBMS, and Condor File Manager.

New product information will be published here in REMark and is available from your nearest Zenith Data Systems dealer, your local Heathkit Electronic Center or the Heath Company catalog. Also a special 24-hour a day on-line computer bulletin board is available for you the end user, for the latest information on new products. The phone number is 616-982-3503. Be sure to have a modem handy.


FORWARD, RIGHT, LEFT, BACK, STOP, GO!
GIVE ME A BREAK! I CAN'T TAKE COMMANDS THAT FAST
HERO 1

With Our Voice Recognition System All Robots Can Talk To One Another

VOICE COMMAND SYSTEM: PRICE: **$595**

► VOREC—CMOS Microprocessor (65C02) based speech recognition board mounts next to and interfaces with the HERO MEMCOM BOARD.
  * Speaker-dependent with nearly instantaneous word recognition rates.
  * Recognition accuracy about 98%.
  * Operates without need for microphone.
  * 256 word vocabulary.
  * 16K onboard battery backed static RAM.

► **VOCOL**—High level language firmware in EPROMS for the HERO MEMCOM BOARD.
  * Supports Immediate and Deferred execution of voice commands.
  * Automatic command training session handled by robot.

► **VOCOL Source Code: $55.00** extra (not sold separately.)

Send check or money order to:*

MICROMATION INC.

9104 Red Branch Rd.
Columbia, MD 21045

VISA® MasterCard

**Allow 2 weeks for check to clear*
Add $3.00 for shipping

For information about this and other products call:
(301) 730-1237
9 am-5 pm Monday through Friday
MasterCard & Visa welcome

Local HUG Club News

The **Oklahoma City HUG** has a new contact person, Bill Cadwallader, and they now meet the 3rd Thursday of each month at 7:30 p.m. at the HEC in Oklahoma City.

The **Middle Georgia Heath/Zenith Users' Group (MGHUG)** would like to announce a name change. Because of their proximity to Robins Air Force Base and the U. S. government's positive response to the Z-100 by allowing employee purchases at substantial discounts, they expect a great number of Z-100 owners to be members of MGHUG in the near future. Therefore, they have decided that their name should include the Zenith name in anticipation of this greater participation. Also note that the MGHUG contact person is their president, Garth K. Haygood, who can be reached at (912) 922-6470.

Club Name: **El Paso Heath Users' Group (EPHUG)**
Club Address: 9213 Waverly, El Paso, TX 79924
Contact Individual: Carlo De Shouten, Sec.
Phone: 915-757-3320
Group Size: 18
Meeting Location: The offices of A. G. Edwards, Ste. 100, 444 Executive Center Blvd.
Meeting Time: Every third Wed. of the month, 7:00 p.m.

LUVAHUG of Woodland Hills, CA now has 80-90 members, also new contact person is Rick Gaitley, 22504 Ventura Blvd., Woodland Hills, CA 91364.

CT, Mystic
Mystic ZDS/HUG
11 Allen Street
Mystic, CT 06355
203-536-6953
Contact Person: Larry Moxon
Meet last Thurs. 7:00 p.m. at 11 Allen, Mystic, CT

INTERNATIONAL HUG CONFERENCE

Official Conference Registration Form
Pheasant Run Convention-Resort Hotel
July 27, 28 and 29

Name(s) ______________________________

Address ______________________________

Company ______________________________

City ______________ State ________ Zip ____

Enclosed is $22.00 per individual to attend the International HUG Conference to be held the weekend of July 27, 28 and 29, 1984. Please send tickets along with information regarding hotel reservations and transportation.

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

If yes, do you want exhibit space during the Conference? ☐Yes ☐No

Special Notice 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. You must contact us prior to May 1, 1984.

For your information:

The $22.00 you are paying for your reservation to the International HUG Conference entitles you to all functions of the Conference. This includes Saturday breakfast, buffet lunch and hors d'oeuvres in the evening. The Prize Drawing will be held during the Saturday evening Cocktail Party. You must be present to win. Vendors and $22.00 ticket holders will be eligible for prizes. ALL prizes will be awarded at that time.

Visitor tickets, for those of you simply attending the seminars and looking at the exhibits, are available for $10.00. Visitor Tickets do not include meals or eligibility to the Prize Drawings.

Send your registration form or a suitable copy to:

Heath/Zenith Users' Group
Attention: International HUG Conference Registration
Hilltop Road
St. Joseph, Michigan 49085

Registration(s) must be postmarked no later than July 15, 1984. Cancellations will not be accepted after this date.

BUGGIN' HUG

Special Announcement

Heath/Zenith Data Systems Software Consultation is proud to announce the implementation of a new source of information regarding Heath/Zenith computers and software. An informational bulletin board is now available at (616) 982-3503. This is a general bulletin board completely open to the public and the phone number is also available for general distribution.

The bulletin board is intended to be an up-to-the-minute source of information concerning Heath/Zenith software products, helpful hints, sample programs, and general information.

The bulletin board is a single-user system and will be available 24 hours a day, seven days a week. The phone will not be answered during periodic required maintenance and updating.

The bulletin board is implemented on a Z-90 computer using a Hayes 1200 Smartmodem. Communication is supported at rates of 300 and 1200 baud. Standard RS232 ASCII protocol using 8 data bits, 1 stop bit, and no parity are used. This is the standard interface of all Heath/Zenith CPS packages as well as the standard setup for modem communications of all Heath/Zenith operating systems.

If you have any comments or questions about the bulletin board, please feel free to contact the Software Consultation Group at (616) 982-3860 during the normal business hours 8:00 a.m. to 7:30 p.m. (Eastern Time Zone).

"QUIC-N-EASY"

Dear HUG,

I have been a HUGger now for two years and still using my old H- 89 which has been going strong for 2 1/2 years without a single breakdown. I have performed a few operations on it like adding memory, double density controller, and parallel printer output just to show that one is not tied to the basic configuration one purchases. From time to time, I notice letters in "Buggin' HUG" praising certain peripheral manufacturers but never see one in behalf of a software manufacturer. This may be of interest to other "HUGgers and HUGgies" who, like me, hate programming but must indulge to turn out good custom programs.

Two years ago, I purchased the "QUIC-N-EASY" applications development system for the H-89 and since then have not written a single BASIC program. This system consists of two main programs. One is the actual development program which dramatically reduces development time and produces a real professional and user friendly application. The other program is the interpreter program used during RUN. It handles index sequential, random, and sequential files in a way that is so much easier than BASIC (BASIC won't handle index sequential!). It enables one to configure the screen using graphics so that the input document looks just like a printed form.

Back to the main subject. Since my original purchase of this program, I have received free updates which were not "bug" corrections but added enhancements. A couple of months back, the entire program was rewritten to add a new report generator and additional goodies. Since this included a new manual and three disks, there was a reduced charge for these. This new program is called Q-PRO-4. Several times I have written to the company with dumb questions (being a beginner), but each time I have received a very prompt and courteous reply and it was signed by the president. The customer service is as good as the product!

I have heard people say that programming is too complicated and that is the reason for getting rid of their computer or not buying one. These are the people who should try Q-PRO-4! Their first program with Q-PRO-4 can be more professional than any BASIC program they could ever write. I think it is worth noting that the company is Quic-n-easi Products, Inc., 136 Granite Hill Court, Langhorne, PA 19047.

Arthur Feilen
Avda Carrero Blanco 169
Mahon, Menorca, SPAIN

Cancel My Call For Help

Dear HUG,

A few days ago I wrote to you requesting some help with the LPH24.ASM program contained in the Device Driver disk of HDOS 2.0 and the article in REMark (June '81). The problem was that the program wouldn't assemble without two errors (ERRNZ failures). I had looked and looked for the cause but wasn't having any success so I called for help. The driver I was trying to fix up was for use with my new DTC-380Z printer.

Tonight (after a scotch and soda - tall, thank you) I launched into some more study and solved the problem! The ERRNZ failures both had the same failure differential (061A). Further, they are both linked to the current counter value (*); and lastly, they occurred after an ERRNZ test that was successful. This meant that someplace in a finite section of the source code, something ran the counter up 061A too high.

The answer is in the HELP CALL $TYPTX section. In the DB grouping that follows, Mr. Chandler used SPACES to make the format. This didn't occur to me before nor did I make the association with the NOTE that preceeds the HELP label. If the appropriate number of SPACES are removed, the counter value is corrected and the ERRNZ tests pass. The program will then assemble error free. I took some SPACES out here and there but kept the format - TABS can also be used as long as the counter value comes out right if that is preferred. The counter value is critical in relation to the driver entry point (DVD.ENT) - that's why controlling it is the key.

Anyhow, my DTC-380Z roars along like gang busters at 9600 baud and will suck up the 174 sector listing in a couple seconds. The alteration to the source code to run the DTX-380Z is simple and requires only changing one line after the WAIT0 label as in the article by Mr. Andrews (REMark #18, June '81). Since there will probably be lots of these printers in use if Heath markets them, you might want to pass this info on to other HUĜgies. If you wish, I will be more than willing to write an article on the printer and how to fix the driver for it.

R. A. Frazier
3368 La Mesa Drive #12
San Carlos, CA 94070

A Problem Doing High Resolution Graphics With the MX-80 F/T

Dear HUG,

I have been trying to do high resolution graphics with the MX-80 F/T and a problem has arisen that I can't seem to overcome. Using HDOS MBASIC, I'm unable to pass the high order bit to the printer. I have tried to set the eighth bit in the MX-80 but I still can't print any more than 127 bits at a time. I am also using the Lindley systems driver which supports the graftrax option. By setting the 8th bit in CP/M MBASIC, I am able to pass more than 127 bits without getting garbage after that point.

In the HDOS version, I entered 'INP(&0344)' and found that the port was configured for 5 bit operation so I 'OUT &0344,3' but still no workie. In the CP/M version, the port was set for 7 bit operation so I did my trick and it worked after setting the 8th bit in the MX-80.

Vectored to 60 ☞

What is a "BB", anyway???

Terry Jensen
Software Developer

Well, once again it is time to write about the HUG Bulletin Board (HUGBB) on the CompuServe timeshare system. Before I dig into the how's of the HUG Bulletin Board, I would like to use this article as a brief overview of "Bulletin Boards" for those of you who are newer members of HUG. Also, many of you may find the history of the HUGBB enlightening.

Let's go back ...

Shortly after I arrived at HUG, while attending one of our staff meetings, mention was made of a "Bulletin Board" or the "BB". What in the world was a "BB"? I had no idea what the other staff members were talking about. I am sure many of you have this same question on your mind right now.

What is a "BB", anyway?

Those of you who have worked in a large manufacturing plant or office building, have undoubtedly become familiar with the area where you will find hanging two, three, or more bulletin boards. The boards are for posting messages and bulletins for all to see. These cork boards provide a facility for tacking pieces of paper containing information pertinent to the particular bulletin board. As you have time, you drop by the boards and read the latest bulletins. One bulletin board may hold the "for sale" items, another may have company related announcements, another may have the coming social events outside the company, another bulletin board may be job postings within the company, and the list could go on and on.

The computer term "bulletin board" is really quite similar. The major differences are; 1) the location of the "bulletin board" is not hanging in a lobby of a building but rather stored in a computer, 2) the messages are not posted to a "cork board" but are "posted" and saved in a message base, which can be accessed only by terminals, and 3) the messages can be read by anyone who has a "terminal" (or computer) and the telephone number (and in many instances, a password) to access the computer bulletin board. The computer bulletin board messages relate to a variety of topics. In addition, computer bulletin boards can offer many unique features, as you will soon see.

Bulletin Boards vary from small local BBS's (Bulletin Board System's) to large host computer services such as CompuServe or The Source. The local BB's are run by users (like yourself) on their own computers, e.g. an H8, H/Z-89, or H/Z-100. The host computer services are running on one, two or more mainframe computers, e.g. PDP-10's and others.

The local BB system is limited in many respects but yet provides a unique service to members in a local area. Only one person can access the BB at any one time. The disk area is limited by the amount of disk space the computer can have "on-line". For example, a computer with two 320K drives, can have "on-line" only something below 640K of disk space, taking into account the lost disk space for the operating system and BB software.

For most practical applications, the local BBS's offer just what is needed. Most of the time, only one person would be using the BB anyway. If you were to call and the line is busy, you just wait a few minutes and try again. 500K of disk space is enough space for a message base and several programs to be shared with others. In addition, these local systems don't cost you anything but your time and the electricity you use.

The mainframe host systems offer much more to the user, but then there is the cost of paying for their services. "Timeshare" systems like CompuServe can have several hundred users from all over the country "on-line" at any one time. The disk area is limited not by the disk space but by the amount of revenue the BB produces for them. Most mainframes will have several hundred megabytes of disk storage on-line that they must divide among their many services including bulletin boards. The amount of disk space allocated to a bulletin board is dependent on the amount of money the group (or BB) generates. This revenue comes from the hourly rate you pay for your time using the bulletin board.

Local BBS's are popping up all over the country and provide help and information for you on a variety of subjects. (You may even have a local Bulletin Board in your area, and you don't even know it.) The national BB's have the "power" of leaving messages that can be read by users from around the country (and world for that matter), simultaneously. Both services have their pros and cons, and both services complement each other.

If I may, allow me to squeeze in a comment here. Whenever you use a BB, whether local or national, be polite to both the operator and other users. Many BB's have been "pulled" or shut down because of rudeness and worse. Bulletin Boards are there for your benefit, and as always it only takes a couple "wise guys" to ruin a good thing.

Meanwhile, back at the ranch ...

It was quite some time after I started working for HUG before I understood what a BB was and the difference between national and local boards. I had to learn the hard way by being "placed" as the SYSOP of the National HUGBB on CompuServe (at that time known as MicroNET). A SYSOP is the SYStem OPerator of the Bulletin Board. This means you have additional "powers" over the other users of the BB. Actually what it means is the SYSOP has to do the cleanup work of the BB! (In addition, as SYSOP of the National HUGBB, I am quite often the middleman between you and Heath-/Zenith ... not a pleasant task!)

Anyway, "cleanup" was exactly my job when I became SYSOP! The BB software was far from "bug free". I didn't know who wrote the program, who supported the program, or who was to fix the problems within the program. My first few months as SYSOP were spent finding solutions to the problems of the original HUGBB software.

At the time, there were around 300 members (if that many) on the HUGBB. Those few members can remember what it was like to logon to CompuServe and sit there for several minutes, waiting for the system to get to you. Sometimes, it didn't make it back except to say you were logged off. (Often times, it didn't even say that!) Ahhh yes, we can reminisce the days of duplicate message numbers, which resulted in a system crash, or the days of copying directly from one another's UFD (User File Directory). Ah, those were the good ole days! (Or were they?)

The Bulletin Board or SIG (Special Interest Group), as CompuServe calls it, presently has over 2400 members and a completely different software package controlling the message base and file space. Those of us oldies remember some of the "great" options we had available, but in all reality, the SIG of today offers many more options and in most cases is faster than the BB of yesteryear. (Sometimes, some of you new SIG users question the faster access. You think access time is slow now, you should have seen ...)

A closer look at the SIG today ...

The SIG has two separate and distinct areas; the message base and the file "Access" database. Both areas are divided into ten (10) sections. Each section has a name or title. This is like "posting" your messages or programs to the appropriate Bulletin Board in the company lobby; e.g. "for sale" items, telecommunication messages, hardware related questions, software related questions, etc.

Members of the SIG can browse through the message base. You have your choice of looking at all sections (or boards) or a single section. Every member has the option of retrieving messages or leaving messages. There are special features for retrieving by name, user ID, or subject. You can retrieve messages "addressed" to or from your number, someone elses number or simply read all messages. The messages may vary in content from problems with hardware or software, to what happened at the latest computer show, to 'How's the weather in San Francisco?'.

The Access area of the SIG is where programs are stored. These programs are executable programs that you can use on your computer. The Access (database) area is simply a place to store the programs, the same as you do on your disk drives. The big difference is that any member of the SIG can have "access" to the programs. The programs can be "downloaded" to your computer, meaning you can copy and save the program to your disk and then run it on your computer. (The program, of course, must have been written for your computer.) Also, if enough free space is available, you can submit your programs to the Access area for other members to use.

The SIG is large enough now that I have a couple of other members helping me out as SYSOP. Kevin Hauser, Dwight Ernest, and Bill Parrott give of their time to answer questions, add new members and hang around to give me some free time to read what is happenin' out there in HUG Land.

Many of the features of the SIG will be discussed in coming issues of REMark. For now, I had better explain briefly what you need to do to get on a bulletin board.

What do you need to get started?

To access a computer bulletin board whether local or timeshare, you will need some special hardware. First, you will need a computer. (I assume you must have one of those things already!) To "talk" with a BB, you need to have a telephone line. The last piece of hardware you need is a modem and connector cable. The modem is the link between your computer and the telephone line. The telephone line is your link to the bulletin board.

You will need two other items to get on a BB. You will need a software modem program that runs on the computer that will handle the input and output to/from the modem. There are many modem programs available in the HUG Software Library and from other vendors. The last item you need is the telephone number of the bulletin board you are going to call.

That is it! You now will be equipped for communicating with any computer system, provided it is set up for communicating with minicomputers (ASCII transmissions). By the way, you are not limited to talking with bulletin boards or timeshare systems. If you have a friend next door or across the country, you can transfer programs back and forth with this same set-up instead of sending diskettes. You may have to pay for a long distance call, but it's kind of nice to have the program right away rather than waiting a few days for the mail.

If you have questions about how to get on the National HUG Bulletin Board or SIG on CompuServe, refer to the MicroNET Connection packages P/N 885-1122 and P/N 885-1224[-37] in the January 1984 issue of REMark magazine (page 32 and 37, respectively). Everyone should have the January '84 issue. Refer to this issue (March 1984) for ordering information on the "Handbook" we have made up to aid in making your way through the HUG SIG.

If you would like to see if there is a local HUG group with a local bulletin board in your area, refer to the Local HUG Club listing in the January 1984 issue of REMark, page 47. Many local HUG groups have a bulletin board system. In addition, many users' (like yourself) have implemented their own BBS. A listing of local BBS's is available on the HUG SIG. (If you have a BBS and would like to have it listed with the others, leave a message to SYSOP on the HUG SIG.)

For now, this is your SYSOP signing OFF.

Exiting at 10:33:10
Thank you for visiting
Off at 10:33 EST
host: call cleared

ARE YOU MOVING?

Don't leave your REMark behind.

Send your change of address in now to:

Heath Users' Group
Hilltop Road
St. Joseph, MI 49085

Part V

An Introduction To 'C'

Brian Polk
86-02 Little Neck Parkway
Floral Park, NY 11001

This is the fifth of a series of articles designed to introduce the 'C' programming language. I am still in the process of learning about this fascinating language, and I intend to continue to pass on information regarding obstacles which I have overcome for each program I write.

The goal today is to learn how to pass options into a 'C' program. Along the way we will delve deeper into the construction of arrays, the use of pointers, and some tricks on handling file I/O more easily. We will also see how to expand wildcards (* and ?) on the command line.

While writing the program we are going to analyze today, I ran into a problem with the 'C' compiler. At one point in the compilation of the program, the system would hang up. After some experimentation with the syntax of the program did not correct the problem, I pulled out the C/80 manual and looked through it again. On page 41, section 15.5, there was a discussion of I/O buffers, and how and where they are allocated. A special note to HDOS users indicated that a device driver could spill over into this buffer area. Even though the manual said this problem will rarely arise when only one driver is loaded, I was still suspicious. I had only one device driver loaded at the time (LP:), but it was rather large because it incorporates some spooling functions. Since I had run out of ideas otherwise, I was willing to look into this further. I re-booted the system, and did not load the LP: driver. When I now compiled the program, the problem disappeared. Just to check on my hypothesis, I re-booted again, this time loading the LP: driver. When I compiled the program again, the problem reappeared. So a word of caution. If you have more than one device driver loaded, or are using a very large driver -- beware! If you experience any unusual problems either when compiling programs or running them, try again without any device drivers loaded. You also might want to check the C/80 manual for a change to C LIBRARY.ASM which will leave additional room for device drivers. Unfortunately this won't help when the problem arises compiling a program. In order to change the 'C' compiler, we would have to recompile the source code (which is not supplied) after we changed C LIBRARY.ASM. Keep in mind that the 'C' compiler is itself a 'C' program, so it is also subject to the same conditions for allocated I/O buffers.

Let's make a few more comments about arrays. In particular, let's look at character arrays. Technically, a string is an array whose elements are single characters. The compiler automatically places the null character (\0) at the end of each such string, so programs can conveniently find the end. Almost all 'C' programs are written to rely on the fact that a string is terminated by '\0'. But keep in mind that the '\0' is not actually part of the string, it just delimits it. Here's an example of how the string "hello" would look like in an array:

```
a[0] = 'h'
a[1] = 'e'
a[2] = 'l'
a[3] = 'l'
a[4] = 'o'
a[5] = '\0'
```

If the string is shorter than the length of the array, all remaining positions (after the '\0') contain 'garbage'. 'C' also provides for rectangular multi-dimensional arrays, although they tend to be much less used than arrays of pointers. Here's why. Consider these two declarations:

```
char a[10][10];
char *b[10];
```

Both arrays are similar, in that a[5][5] and b[5][5] are both legal references to a single 'char'. But 'a' is a true array with 100 storage cells allocated, and the conventional rectangular subscript calculation is done (internally) to find any given element. The 'b' declaration only allocates 10 pointers, each of which must be set to point to an array of 'chars'. If each points to a ten-element array, then 100 storage cells are being used, plus the ten pointers. Thus, the array of pointers uses slightly more space, and may require an explicit initialization step. But it has two advantages: accessing an element is done by indirection through a pointer rather than by multiplication and addition, and the rows of the array may be of different lengths. That means that if a pointer 'points' to the beginning of a character string, then the end of the string will be indicated by '\0'. Therefore, the string can be any length. This brings us back to our discussion of the 'argv' command line array. I made one mistake in the definition of this array. It is an array of pointers to character strings (not just characters). By convention, 'argv[0]' is the name by which the program was invoked, so 'argc' is always at least 1. Also, 'argv[argc]' will always be '\0'. Here's an example which I hope will clear this up for both you and I.

```
>more test.dat

argc=2
argv[0] --> "more\0"
argv[1] --> "test.dat\0"
argv[2] --> "\0"
```

Read back over the last paragraph, referring to this example, and see if this all makes more sense. I have spent some time getting this straight in my mind. Once you do, you will able to use the 'argv' array to great value in your programs.

Now let's look at a new command and a new operator. We have used the 'if' statement before, but we haven't mentioned the 'else' option. This allows us to execute a statement when an 'if' test is false. The 'else' statement can also be another 'if' statement, so that a nesting affect is produced, such as:

```
if (c == 'L') flag = 1;
else if (c == 'W') flag = 2;
else if (c == 'C') flag = 3;
else printf("Bad Option.\n");
```

This structure provides for testing of several conditions and also for a 'no-match' situation. We will see another way of accomplishing this same thing with another command in a later article.

In the past, we used the '++' operator to increment a variable by one. Along the same lines is the '+=' operator which increments one variable by another variable or constant. For example,

```
     x = x + y;
```

is equivalent to

```
     x += y;
```

There are operators for subtraction (-=), multiplication (*=), and division (/=).

Now we are ready to look at our program. It's called 'wc' (word counter) and it counts the lines, words, and characters in a file. For our example, a word is defined as a sequence of characters ending with a space, tab, or new line. The program accepts options in case we only want one or two of the counts produced. It will also allow more than one file on the command line, and for file name wildcards (* and ?). Here's the program:

```
#include "tprintf.c"
#include "command.c"
#define EOF -1
#define NULL 0
#define YES 1
#define NO 0
extern int fin;
main(argc, argv)
int argc;
char *argv[];
{
int character_flag, word_flag, line_flag;
int total_characters, total_words, total_lines;
int file_characters, file_words, file_lines;
int i, j, c;

character_flag = YES;
word_flag = YES;
line_flag = YES;
total_characters = 0;
total_words = 0;
total_lines = 0;
i = 1;
j = 0;
command(&argc,&argv);

if (argv[i][j] == '-')
   {
   character_flag = NO;
   word_flag = NO;
   line_flag = NO;

   for (j=1; (c=argv[i][j]) != NULL; j++)
      if (c == 'L') line_flag = YES;
      else if (c == 'W') word_flag = YES;
      else if (c == 'C') character_flag = YES;
      else {
           printf("Bad Option.\n");
           exit(8);
           }
   i++;
   }

for (; i < argc; i++)
   {
   if ((fin=fopen(argv[i],"r")) == NULL)
      {
      printf("File Not Found.\n");
      exit(8);
      }
   file_characters = 0;
   file_words = 0;
   file_lines = 0;
   while ((c=getchar()) != EOF)
      {
      if (character_flag)
         file_characters++;
      if (c == ' ' || c == '\t' || c == '\n')
         if (word_flag)
            file_words++;
      if (c == '\n')
         if (line_flag)
            file_lines++;
      }
   fclose(fin);
   printf("%d %d %d %s\n", file_lines, file_words,
                   file_characters, argv[i]);
   total_lines += file_lines;
   total_words += file_words;
   total_characters += file_characters;
   }
printf("%d %d %d Total\n", total_lines,
                   total_words, total_characters);

}
#include "stdlib.c"
```

Notes about the program:

1) command (&argc,&argv);

This function expands file name wildcards in the command line. Notice that the arguments are the addresses (&), not the actual argument. We must include the source code for this function (#include "command.c"). I had to correct one problem with this source code. At one point in the code, a reference is made to a function 'err'. Since I could not resolve this reference in any of the supplied 'C' libraries, I changed it to a 'printf' statement. This seems to work fine. If you get an unresolved reference from the assembler, you will also have to make this change. The 'command' function uses several other functions which are supplied in 'stdlib.c'. We put the 'include' for this library at the end of the program, because it uses a preprocessor feature whereby only the routines required by the program are included with our program.

2) #include "tprintf.c"

This is a smaller version of "printf.c" which can be used if left justification or precision are not needed.

3) extern int fin;

This defines to our program a variable which has been allocated elsewhere, in this case the HDOS/C interface. 'fin' is initialized to the channel number for the terminal. 'fout' serves the same function for terminal output. This variable is the channel number used by 'C' in I/O commands where no channel number is specified (such as 'getchar()'). What this means to us is that by changing the value assigned to 'fin', we can change the file which 'getchar()' uses for input. Check how we use the variable in the statement where the file is opened.

4) The convention used for passing options to a program is to precede then by a minus sign (-), following the command name (eg. wc -l test.dat). The options used by our program are:

'l' - to count lines
'w' - to count words
'c' - to count characters

By default, all three options are on. If the first parameter on the command line (argv[1]) starts with a minus sign (argv[1][0] == '-'), we turn all three options off, and then search through the characters following the '-', turning on the appropriate option. If an option other than those indicated above is detected, the program stops with an error message.

5) Try experimenting with running this program, specifying different options, and using multiple file names and wildcards on the command line. Once you understand the program, try these exercises:

a) Only print out a count for the options which were selected (suppress zeros).

Vectored to 16 ☞

NEW FROM FLOPPY DISK SERVICES, INC.

the H-89 TWOET SYSTEMS

We've got a great idea for your H-88, 89 or 90. It's a dual *internal* half height drive system. Two of our half height 5¼" drives can replace your built-in disk drive, doubling your information storage capacity.

Floppy Disk Services provides you with everything you need. That's two double-sided, double or single density, half height drives in either 48 or 96 tpi format, ail hardware, cables and power connector adaptors. And most important, you get easy, step-by-step instructions, in the Heath/Zenith tradition of good, clear documentation.

We've thoroughly tested the TWOET/Heath set-up. Remember that a double sided 48 tpi will work perfectly as a single sided drive right out of the box! Hard or soft sectored—so you can even use this system with your H-17 controller. And of course we have the software drivers (additional cost) to run 48 or 96 tpi double sided, single density drives on the H-17.

Model TWOET 455
2 Shugart SA-455 half height
48 tpi double sided
All hardware
Metal, shielded mounting plates
Data cable with chassis connector
Power 'Y' connector
Complete documentation
Price .... $605.00 complete

Model TWOET 465
2 Shugart SA-465 half height
96 tpi double sided
All hardware
Metal, shielded mounting plates
Data cable with chassis connector
Power 'Y' connector
Complete documentation
Price .... $755.00 complete

Wondering what to do with your internal drive if you buy this system? Here's the solution. If you purchase a dual half height system for your Heath computer from Floppy Disk Services, just include an extra $50.00 plus shipping and receive a single 5¼ case with power supply *and* data cable ready to receive your SIEMENS internal drive! The case with data cable is normally a $70.00 item. And the cable that comes with your TWOET system includes the external chassis disk I/O connector.

Due to production deadlines, prices in this ad are 2 months old, so we encourage you to call us for current prices and new product info. Prices and specs subject to change without notice.

Dealer inquiries invited.

PAYMENT POLICY — We accept MasterCard, VISA, personal checks and Money Orders. We reserve the right to wait 10 working days for personal checks to clear your bank before we ship. All shipping standard UPS rates plus shipping & handling. NJ residents must add 6% tax.

Toll Free Order Line:
(800) 223-0306

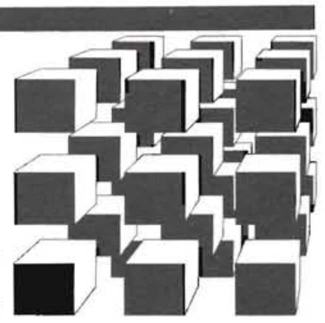
Tech Help or Info:
(609) 799-4440

H-88/89/90 are registered trademarks of Heath Corp.

FLOPPY DISK SERVICES™ INC.

741 Alexander Rd. Princeton, NJ 08540

Random Files



David E. Warnick
RD #2 Box 2484
Spring Grove, PA 17362

Last month we took an overview-type look at data files. We discovered that there are two types of files, random and sequential, and that the differences between those types are the way that data is stored when it is placed in RAM or on a disk. We looked at some of the advantages and disadvantages of each type of file and concluded that each has a purpose and that the type we use for a given application should be determined by what we want the file to do. Finally, we looked at a single entry for the file and called it a record. We discovered that each record is made up of smaller pieces of data which would be meaningless by themselves, and we called them fields.

With the background gained from that article, we'll go another step on our journey through file handling by looking at random files this month. We'll look at the special MBASIC commands which permit us to place our data records into a random file and retrieve them from it. Then we'll actually build and manipulate some random files. As we do this, some of the things we try may not work. Actually, they're working just the way they should, but not as we'd probably expect. The best way to learn these things is to experiment, see the results, and redo them. This way, not only will you learn to use random files, but you'll also learn a bit about how your system works. So if something appears to be wrong as you write and operate the programs presented here, read on. I may have intentionally led you into a common pitfall for a purpose. The important thing is that by the end of the article, these will all be covered and corrected.

That brings up the subject of study habits. While these articles are intended to be informal, they will, hopefully, guide you into a learning experience. You'll get the most from them and the other articles you find in REMark if you read them completely, then go back to the beginning and go through them again, completing the programming or experiments as you do. It's also a good idea to try to work when rested and without interruptions. Believe me, I know it's impossible at times as I've got a family, too.

Now for our introduction to random files. The first thing we'll do is create a file and put information into it. Having done this, we'll go into the file and retrieve that data. Let's get started.

No matter what kind of file we're working with, we have to open it to store or retrieve data. In keeping with past practice, we'll begin our program on line 1000. The format of the MBASIC command to open a random file is:

```
OPEN "R",#1,"FILENAME",80
```

The number 1 specifies channel 1 open for use as a random file. Check your operating systems manual to see how many channels your system and computer can access at one time. The file name is the name you want assigned to the file. In multi-disk drive systems, it could include the drive designation the file is located on. The final number, in this case 80, indicates how many characters are in each record for this particular file. As you recall from last month's article, the record is made up of fields. In our example, the sum of the characters in each of the fields within the record equals the record length, or 80. The record could contain the following information:

Variable	Data Field	Character Pos.	No. of Char.
LN$	Last Name	1 - 15	15
FN$	First Name	16 - 30	15
MI$	Middle Initial	31	1
A$	Address	32 - 55	24
T$	Town	56 - 73	18
S$	State	74 - 75	2
ZC$	Zip Code	76 - 80	5
		TOTAL	80

So that the computer will know where each field is within each record of the random file, we describe the record using the MBASIC statement FIELD. We could describe the 80-character record with names and addresses shown above for file number 1 as follows:

```
FIELD #1, 15 AS LN$,15 AS FN$,1 AS MI$,24 AS A$,18 AS T$,
2 AS S$, 5 AS ZC$
```

Every record in our file would contain 80 characters. Within each record, the last name (LN$) would always be within the first 15 characters, the first name would always be in characters 16-30, etc.

Of course, you've already noted that all last names are not made up of 15 characters. The same thing applies to the other fields too. So how do we fill up the extra space so we'll be able to find the record and the data we want? We move the data to the left or the right side of the field and fill the extra character positions with spaces. There are two MBASIC commands available for this purpose. LSET left justifies the data (places the data in the extreme left portion of the field) and RSET right justifies the data. We could input the last name and prepare it for insertion into a random file as follows:

```
INPUT "LAST NAME";N$
LSET LN$=N$
```

The LSET command would take the information in N$ and put it in the left portion of LN$. Any of the 15 characters not needed would be filled with spaces.

Looking back at the FIELD statement above, we see that there are seven fields for each record. When data has been input and set with LSET or RSET for each field, we are ready to insert the complete record of seven fields into the random file. To do this we use the PUT command. When using this command, we specify the channel the data is going to (in our case it's number 1) and the record number the data is assigned to within the file. If we omit the record number, the first PUT in our program will assign its data to record number 1 and each successive PUT command will assign its data to the next higher record. Our program line would look like this:

```
PUT #1
```

Thus, we would build a file of names and addresses. When we're finished with our file, we'll close it using the command line:

```
CLOSE #1
```

Armed with this information, we're now ready to build a data file. The file we'll create will have twenty-six records. Each record will have three fields containing:

1) A letter of the alphabet
2) The phonic for that letter
3) The number of the letter within the alphabet

We'll insert the information into the file in the correct order. If this weren't done, we'd have to sort the file and that's a subject requiring a couple of articles by itself. I'll cover sorting in a later article, but first we must understand the basics of random files. To create the file, type and save the program MAKEFILE.BAS.

```
2 '**********MAKEFILE.BAS**********
4 '**********DAVID E. WARNICK******
6 '**********COPYRIGHT 1983********
1000 OPEN "R",#1,"ALPHA.DAT",26     'OPEN THE FILE
1010 FIELD #1,1 AS L$,10 AS A$,15 AS N$
     'DESCRIBE THE RECORD
1020 FOR X=1 TO 26          'FOR EVERY LETTER OF THE
                             ALPHABET
1030 INPUT "LETTER";IL$          'INPUT A LETTER
1040 INPUT "PHONIC";IA$          'INPUT PHONIC
1050 INPUT "NUMBER";IN$          'INPUT NUMBER
1060 LSET L$=IL$        'READY TO PUT IN RECORD
1070 LSET A$=IA$        'READY TO PUT IN RECORD
1080 LSET N$=IN$        'READY TO PUT IN RECORD
1090 PUT #1             'PUT RECORD INTO FILE
1100 NEXT X             'CONTINUE TO END OF ALPHABET
1110 CLOSE #1           'ALL DONE - CLOSE THE FILE
1120 END
```

Now run the program and enter the following information as you are asked for inputs. At the end of 26 letters, your data file will be complete.

LETTER	PHONIC	NUMBER
A	ALPHA	FIRST
B	BRAVO	SECOND
C	CHARLIE	THIRD
D	DELTA	FOURTH
E	ECHO	FIFTH
F	FOXTROT	SIXTH
G	GOLF	SEVENTH
H	HOTEL	EIGHTH
I	INDIA	NINTH
J	JULIETT	TENTH
K	KILO	ELEVENTH
L	LIMA	TWELFTH
M	MIKE	THIRTEENTH
N	NOVEMBER	FOURTEENTH
O	OSCAR	FIFTEENTH
P	PAPA	SIXTEENTH
Q	QUEBEC	SEVENTEENTH
R	ROMEO	EIGHTEENTH
S	SIERRA	NINETEENTH
T	TANGO	TWENTIETH
U	UNIFORM	TWENTY-FIRST
V	VICTOR	TWENTY-SECOND
W	WHISKEY	TWENTY-THIRD
X	X-RAY	TWENTY-FOURTH
Y	YANKEE	TWENTY-FIFTH
Z	ZULU	TWENTY-SIXTH

Now if you can't wait to see what's in your newly created file, return to your operating system and turn on your printer. If you use CP/M, type:

```
LIST ALPHA.DAT
```

If you're using HDOS, type:

```
PIP
LP:=A:ALPHA.DAT
```

Notice that as each record is printed, the data we put into it is exactly where we laid it out with our FIELD statement.

So far, we've seen how to put data into a random file. The same rules apply no matter how many fields are in each record, and no matter what the size of the record may be. With the information you now have, you can write a program to put information into a random file in any format you want. The next logical step is to find a way to retrieve that information from the file so you can use it. The last paragraph showed you how to print your entire file, but we want to be able to pick a specific record and look at that.

To get information from a random file, we must first open that file, just as we did when putting data in. Then we must tell the computer the size of each record and the arrangement of the fields within that record. Again, we do this the same way we did it when adding to the file. We use the FIELD statement. In fact, if we were to write a program to both enter and retrieve data from a random file, only one OPEN and one FIELD statement would be required.

The GET statement is used to read data from a random file into the random buffer. It is written in the format:

```
GET #filenumber, recordnumber
```

Just as with the put statement, if the record number is omitted, the computer will begin with record number 1 and advance through the file with each successive GET statement.

Once the information is retrieved, it can be processed, printed, or used in any way we want. When we're finished with the file, it must be closed. This is, once again, done the same way it was done when we were adding information to the file.

If we want specific information, it appears we must know its record number. If we do, life is easy. Many files can be built so the record is obvious. An inventory file could use item numbers as record numbers and the request for that item could automatically select the correct record number and retrieve it. The file ALPHA.DAT which we just built has an obvious record number for each item. Letter A is record #1, B is record #2, etc.

However, the record number is not always so obvious. With a sorted random file that's no problem. We can use what is called a binary search routine to find any item quickly, even in very large files. We'll set up two programs. The first will calculate a record number and go directly to it, and the second will use a binary search routine. Each will ask for a single letter input, go to the ALPHA.DAT file, give us the phonetic word for that letter, and tell us where the letter appears in the alphabet. An input other than an upper case letter will cause the program to end.

In our first program we'll ask for a letter. Then we'll calculate the record number where that letter and its associated information is stored by using the ASCII value of the letter chosen (A is 65) and subtracting 64. If that value is not within the limits of 1 to 26, we know that other than an upper case letter was selected and the program will end. Here's the program.

```
2 '**********READFILE.BAS**********
4 '**********DAVID E. WARNICK******
6 '**********COPYRIGHT 1983********
1000 OPEN "R",#1,"ALPHA.DAT",26     'OPEN THE FILE
1010 FIELD #1,1 AS L$,10 AS P$,15 AS N$
     'DESCRIBE THE RECORD
1020 PRINT            'SKIP A LINE ON THE SCREEN
1030 INPUT "WHAT LETTER SHALL I LOOK UP";I$  'REQUEST INPUT
```

```
1040 RN=ASC(I$)-64  'CALCULATE RECORD NUMBER
1050 IF RN<1 OR RN>26 GOTO 1110    'NOT A CAP LETTER - END
1060 GET #1,RN          'RETRIEVE THE RECORD
1070 PRINT              'SKIP A LINE ON THE SCREEN
1080 PRINT "THE LETTER ";L$;" IS THE ";N$;" LETTER OF THE ";
"ALPHABET."      'PRINT MESSAGE
1090 PRINT "ITS PHONETIC IS ";P$    'PRINT MESSAGE
1100 GOTO 1020          'GO BACK AND DO IT AGAIN
1110 CLOSE #1           'CLOSE FILE BEFORE ENDING
1120 END
```

Looking at the message, you'll see that the spaces we added to fill the 15-character number field are still there. They're also attached to the phonetic but can't be seen because it's at the end of a line. We'll take care of that in a future article. For now let's look at a binary search routine. With a sorted random file (ours is sorted because we put the information into it in the correct order), the binary search lets us find the data we want in a minimum amount of time.

To understand how a binary search works, think of a deck of 100 file cards. Each card has a different name on it and all cards are in alphabetic order. Now suppose we want to find my name, David Warnick, and it's on card number 98. The first thing we do is compare what we want to card 50 (the middle of the deck). If they're not equal we'll find card 50 too small or too large. In this case it would be too small. Because the cards are in order, we know that cards 1 - 49 are also too small and may be discarded.

Next we look at the middle of the remaining cards, number 75. Again it's too small and cards 51 - 75 are discarded. The process would continue by looking at the middle of the remaining cards until a match is found. To find our match we would look at cards 88, 94, 97, 99, 98. I chose number 98 as it is a worst possible case. We had to look at 7 cards. The most looks a search can require to find the item we want using a binary search routine in a correctly ordered (sorted) file is equal to the lowest power of two which equals or exceeds the number of records in that file. Two to the seventh power equals 128, so the most looks in a file of 65 - 128 records is 7 to find what we want. A total of only 12 looks is required if our file contains 2049 to 4096 records. This is a tremendous advantage of random files over sequential files which must read and compare each item until the correct record is found. The average number of looks would be half the total records in the file. The program which follows uses a binary search to find the letter we want in our file.

```
2 '**********SRCHFILE.BAS**********
4 '**********DAVID E. WARNICK******
6 '**********COPYRIGHT 1983********
1000 OPEN "R",#1,"ALPHA.DAT",26     'OPEN THE FILE
1010 FIELD #1,1 AS L$,10 AS P$,15 AS N$
     'DESCRIBE THE RECORD
1020 INPUT "GIVE ME A LETTER";I$    'GET AN INPUT
1030 IF I$="?" GOTO 2000         'TYPE A ? TO END
1040 IF ASC(I$)<65 OR ASC(I$)>90 THEN
     PRINT "SORRY I ONLY TAKE "; "CAPITOL LETTERS":
     GOTO 1020      'ERROR MESSAGE FOR BAD INPUT
1050 A=1           'SET LOWER LIMIT OF SEARCH
1060 C=26          'SET UPPER LIMIT OF SEARCH
1070 B=INT((A+C)/2)        'FIND MIDDLE OF SEARCH RANGE
1080 GET #1,B              'GET THE MIDDLE RECORD
1090 IF L$<I$ THEN A=B+1:GOTO 1070        'ITEM TOO
     SMALL SO  MOVE LOWER LIMIT OF SEARCH UP AND
     DO IT AGAIN
1100 IF L$>I$ THEN C=B-1:GOTO 1070        'ITEM TOO
     LARGE SO  MOVE UPPER LIMIT OF SEARCH DOWN AND
     DO IT AGAIN
1110 PRINT  "THE  LETTER  ";L$;"  IS  THE  ";N$;
     "LETTER  OF  THE ALPHABET."
     'YOU FOUND IT. PRINT THE MESSAGE.
1120 PRINT "ITS PHONIC IS ";P$     'ADDITIONAL MESSAGE
1130 GOTO 1020          'DO IT AGAIN FOR ANOTHER INPUT
2000 CLOSE #1           'CLOSE FILE BEFORE ENDING
2010 END
```

With this program saved, you can run it to look up information from a sorted random file. There are a lot of subjects to cover yet, like:

1) Adding information to an existing random file
2) Deleting information from a random file
3) Changing information in a random file
4) Sorting a random file

These items require a lot of space and will be covered in the next couple of months. For now you've learned to create a random file, retrieve information from it, and search for information with a binary search routine. That's quite enough to absorb at one time so we'll break at this point. As always, if you have questions, don't hesitate to write. Please include a stamped envelope if you need an answer, and try to limit your questions to these articles and the principals presented here. I'd love to answer more but time doesn't permit. Also if your question concerns one of these programs not running for you, run a listing on your printer and send it along. It may help.

See you next month.

Vectored from 12

b) If only one file was specified to be counted, don't print the 'Total' message.

c) If no file names are supplied (argc=1), print a 'help' message of how the command should be formatted.

d) Try any other modifications which you think will make the program more useful.

Until next time, 'C' you later.

PRIMERS FOR THE BEGINNER

GETTING STARTED WITH CP/M AND MBASIC

WITH PARTICULAR REFERENCE TO RANDOM FILES

SPECIFY "SOFT SECTOR" DISK *FOR Z-100* CP/M—ZDOS BASIC PRIMER.

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

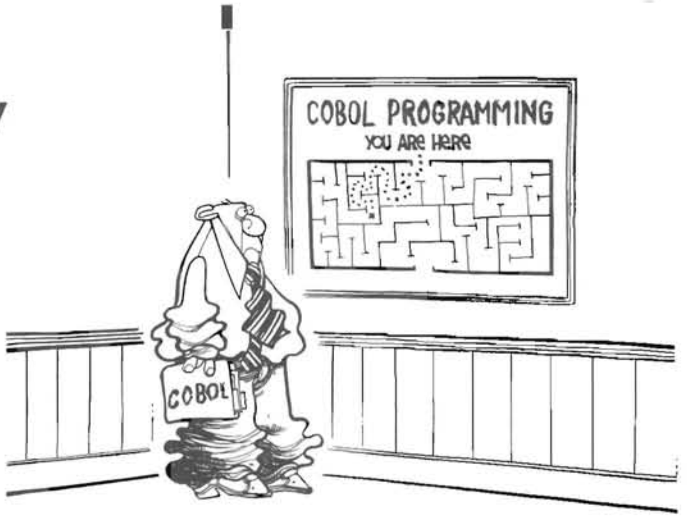
GETTING STARTED WITH HDOS & ASSEMBLY LANGUAGE PROGRAMMING

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

COBOL Corner V



H .W. Bauman
493 Calle Amigo
San Clemente, CA 92672

Introduction

Did you Key-In the Data File from the last COBOL Corner? Did you find it to be a lot of work? For future programs we will supply the Data Files for you on the HUG Disks! Be sure that you obtain the HUG COBOL Corner Disk-I.

Last time we left off at the end of Phase II in developing our Sample Program #1. Now, let's try writing the COBOL code on your COBOL Coding Forms. (Did you get your forms? We will be using them for every program.) As previously stated you MUST work with "COBOL Corner", not just READ it!

Developing Sample Program #1 (Continued)

Phase III -- Step 1

*(**Note:** Do not type the first line shown below, with all the numbers. It is furnished so that you can see what Columns should be used!)*

```
12345678901234567890123456789012345678901234567890123456789012345678901234567890
000010 IDENTIFICATION DIVISION.
000020 PROGRAM-ID.     PRGM01.COB.
000030 AUTHOR.         YOUR NAME.
000040 INSTALLATION.   COBOL Corner.
000050 DATE-WRITTEN.   SPRING, 1983.
000060 DATE-COMPILED.  SPRING, 1983.
000070 SECURITY.       NONE.
000080*
000090*****************************************************************
000095*                                                                *
000100*   This program reads a customer account, name and address      *
000110*   INPUT file and prints a CUSTOMER LIST.                       *
000120*                                                                *
000130*****************************************************************
000140*
```

The purpose of the IDENTIFICATION DIVISION. (DO NOT forget this period) is to identify and document the program. All paragraphs MUST start in column 8. This DIVISION may contain up to six (6) paragraphs. Only the PROGRAM-ID. (DO NOT forget the hyphen or the period!) is required by COBOL-80; but, we want to learn to write "Good", portable programs, so we will use them all. The Program-ID is a User-Defined Name and MUST meet the rules given previously; plus, since this is a disk-file-name, COBOL-80 will allow us six (6) characters and the .COB extension.

The coding lines 000080-000140 are all comment lines, so we MUST put the * in column 7 as previously described. Also, every "Good" program MUST have DOCUMENTATION explaining what the program does. I have furnished a paragraph, about as short as you could conceive of, to describe this program. This should be enclosed in a "BOX" of *'s to improve readability.

The entries after Program-ID, Author, etc. will always start under the "D" in DIVISION. Again "Good" programming requires this!

```
12345678901234567890123456789012345678901234567890123456789012345678901234567890
000150 ENVIRONMENT DIVISION.
000160*
000170 CONFIGURATION SECTION.
000180*
000190 SOURCE-COMPUTER.   HEATH-H-8.
000200 OBJECT-COMPUTER.   HEATH-H-8.
000210*
000220*
000230 INPUT-OUTPUT SECTION.
000240*
000250 FILE-CONTROL.
000260     SELECT FILEL1
000270         ASSIGN TO DISK
000280             ORGANIZATION IS LINE SEQUENTIAL.
000290     SELECT CUSTOMER-LIST-LINE
000300         ASSIGN TO PRINTER.
000310*
```

Most of COBO L is machine independent. The ENVIRONMENT DIVISION takes care of the differences; such as, disk and printer call-outs. System-Names assigned by manufacturers for specific I/O devices can vary. Each file that is used within the program MUST be ASSIGNED to an I/O device in this DIVISION. Note the indentation and the use of one verb per code line. Also, note the blank comment lines between groups of coding. This provides us with a readable, "Good" program! DO NOT miss a period, space or hyphen and watch your spelling of reserved words. These can cause compiler errors! Watch for "O" vs "0" and "I" vs "1".

Now would be a good time to refer to your COBOL-80 Reference Manual and re-read about the FILE-CONTROL and SELECT Entry! You MUST know how to structure these clauses!

Remember the CONFIGURATION SECTION. (DO NOT forget this period) allows you to specify the computer that you will use for your program. The INPUT-OUTPUT SECTION. (that period again and a hyphen this time) allows you to specify the I/O devices, and the FILE-CONTROL. (period and hyphen again) paragraph is used to ASSIGN your files by User-Defined Names to a specific I/O device.

```
12345678901234567890123456789012345678901234567890123456789012345678901234567890
000320 DATA DIVISION.
000330*
000340 FILE SECTION.
000350*
000360 FD  FILEL1
```

```
000370       VALUE OF FILE-ID IS "A:FILEL1.DAT"
000380       RECORD CONTAINS 80 CHARACTERS
000390       LABEL RECORDS ARE STANDARD.
000400*
000410 01    CR-CUSTOMER-ACCT.
000420       05  FILLER                                 PIC X(02).
000430       05  CR-CUSTOMER-ACCT-NO                    PIC 9(05).
000440       05  CR-CUSTOMER-NAME                       PIC X(20).
000450       05  CR-CUSTOMER-ADDRESS                    PIC X(22).
000460       05  CR-CUSTOMER-CITY                       PIC X(13).
000470       05  CR-CUSTOMER-STATE                      PIC X(02).
000480       05  CR-CUSTOMER-ZIP                        PIC 9(05).
000490       05  FILLER                                 PIC X(11).
000500*
000510 FD    CUSTOMER-LIST-LINE
000520       RECORD CONTAINS 132 CHARACTERS
000530       LABEL RECORDS ARE OMITTED.
000540*
000550 01    CL-CUSTOMER-LIST-LINE.
000560       05  FILLER                                 PIC X(03).
000570       05  CL-CUSTOMER-ACCT-NO                    PIC 9(05).
000580       05  FILLER                                 PIC X(02).
000590       05  CL-CUSTOMER-NAME                       PIC X(20).
000600       05  FILLER                                 PIC X(03).
000610       05  CL-CUSTOMER-ADDRESS                    PIC X(22).
000620       05  FILLER                                 PIC X(03).
000630       05  CL-CUSTOMER-CITY                       PIC X(13).
000640       05  FILLER                                 PIC X(02).
000650       05  CL-CUSTOMER-STATE                      PIC X(02).
000660       05  FILLER                                 PIC X(02).
000670       05  CL-CUSTOMER-ZIP                        PIC 9(05).
000680       05  FILLER                                 PIC X(50).
000690*
000700*
000710 WORKING-STORAGE SECTION.
000720*
000730 01    WS-PROGRAM-SWITCH.
000740       05  WS-END-OF-FILE-SWITCH                  PIC X(03).
000750*
```

The DATA DIVISION. (that period again) is where you describe your files. Each file in the ENVIRONMENT DIVISION MUST be defined in this division. The FILE SECTION. (period!) is where the files, records, and fields to be READ or WRITTEN by your program are defined.

FD is a two (2) character alphabetic reserved word (means File Description) used to head a paragraph that is used to identify a specific type of file. THERE IS NO PERIOD until the end of the FD statement! The File-Names, FILEL1 and CUSTOMER-LIST-LINE, MUST be exactly the way they appear in the SELECT clauses in the ENVIRONMENT DIVISION! FD must start in Column 8 with two (2) spaces following; so that your User-Defined File-Name starts in Column 12. (You might want to look up "FD" in your COBOL-80 Reference Manual for more information.)

There are three (3) possible clauses that we will be using before the PERIOD:

1) VALUE OF FILE-ID IS "Disk File-Name" -- REQUIRED for DISK- ASSIGNED Files. The "A:FILEL1.DAT" describes the name of your DATA-FILE on Disk A in Drive A.

2) RECORD CONTAINS 80 CHARACTERS clause comes from your RECORD CHART which has 80 Columns. When you are describing the output file which was diagrammed on the PRINT CHART, you will call out 132 characters!

3) LABEL RECORDS ARE STANDARD or OMITTED clause is used for I/O purposes. STANDARD is required for DISK and OMITTED is required for PRINTER!

Now, we will use our Record Chart for reference to describe FILEL1, as shown by code lines 000410-000470 using LEVEL NUMBERS (we will go into level numbers in a future article but you can read about them in your COBOL-80 Reference Manual).

Line 000410 has the 01 Level Number starting at column 8 with two (2) following spaces and a User-Defined Data-Name (remember the rules for Names and that they must be unique) which will describe the RECORD as a whole. We tried to make the Data-Name self-documenting. The prefix "CR-" means that it belongs to our program specified Customer Record input from disk-file and referred to as "CUSTOMER-ACCT" (for Account). Be sure to end the 01 Level with a period!

Lines 000420-000470 are assigned 05 Level Numbers (Remember the number can be any number from 02-49 and must be ascending as we change levels!). Assigning the number is up to the programmer BUT it is best to leave a gap between level numbers for future program upgrades and/or modifications. We will be doing many things with level numbers in future programs and explaining them as we go.

Each 05 line has a User-Defined Data-Name that will start with the prefix "CR-" in this program and will be a self- documenting, readable name. FILLER is a COBOL reserved word that serves as a "placeholder" in the record.

"PIC" is short for PICTURE CLAUSE (this would be a good time to look up FILLER and PICTURE CLAUSE in your COBOL-80 Reference Manual--we will explain the PICTURE CLAUSE more as we use it). The "X" stands for Alphanumeric fields and the "9" is for Numeric fields. The (0n) states how many characters there will be in the field. The PIC X(0n). (must always end with a period here) clauses should line up for a "nice" appearance, readability, and for the "Good" programming style. Also, you MUST remember that the "(0n)" numbers must total up to the 80 characters specified in the RECORD CONTAINS clause!

Next, we will refer to our Print Chart which describes our Output File--CUSTOMER-LIST-LINE. It is named with a User-Defined Name for our "CUSTOMER LIST" program. I used the File-Name CUSTOMER- LIST-LINE because our program will output a LINE at a time: thus, suffix "-LINE" appended to the program name! You might have better ideas for names. It is up to the programmer.

Line 000550 has the 01 Level Number, starting at column 8, with two (2) spaces following the User-Defined Name which will describe our Output. We will start our names with the prefix "CL-" so we will know that they pertain to the output "CUSTOMER-LIST-LINE".

Lines 000560-000680 are 05 levels with a Data-Name or FILLER for each data item or blank space shown on the Print Chart output line that we have diagrammed. Each item has been named with a User-Defined Name per our Rules. Each blank space has a FILLER line to represent the spacing on the output line. Each FILLER or Name MUST have a PICTURE CLAUSE (PIC X(0n) or PIC 9(0n) depending on whether the field is alphanumeric or numeric) and END with a period.

The numbers inside the parenthesis must total 132 as specified in the RECORD CONTAINS clause for the output file.

Lines 000710-000740 describe the WORKING-STORAGE SECTION of the program (do not miss the hyphen or the period). This SECTION is rather short in this program but as we get into complex programs, this section might have 100 LINES OF CODE. We use the WORKING- STORAGE SECTION for records and fields required by the program that are not defined in the file part of the program; because, they are not part of the external files.

Line 000730 is an 01 Level Record which in this program will have only one (1) lower 05 field. We start our User-Defined names in the WORKING-STORAGE SECTION with the prefix "WS-" so we know they are defined in this section. We are calling the names "switches" because we are going to require a two-state indicator for program control. Two-state devices are often used to turn lights or electronic circuits ON or OFF and are called "switches". In our program we are going to use a "switch" to tell the program whether the end of the input file has been reached (a value of "YES") or not (a value of

"NO"). Our User-Defined name -- WS-END-OF-FILE-SWITCH -- is very readable and self-documenting. It requires a PICTURE CLAUSE -- PIC X(03). -- to provide space for "YES" or "NO" (Three (3) spaces!).

Closing

In the next "COBOL Corner" we will complete Phase III -- Step 1 in the development of Sample Program #1. This will cover the PROCEDURE DIVISION. In the PROCEDURE DIVISION, we will use some of the many COBOL Verbs that we will explain as we progress.

I would suggest that you prepare another Disk A by duplicating your present Disk A and ERAse the *.COB and *.REL disk files plus any executable (.COM) files you might have saved on the disk. I use a "new" Disk A for each Sample Program we work with (this will give you a library of COBOL programs to refer back to and safeguard them, Back-Ups).

Now, for your "HOMEWORK", I would like you to do the following three (3) projects before the next "COBOL Corner":

1) Put this "new" Disk A in Drive A and using your Editor, Key-In the coding lines we have designed above. That will include lines 000010-000750. Save this keyed code on Disk A with the name PRGM01.COB. Now, make a print-out of this file and carefully compare your listing to the COBOL coding lines in this article. Make any corrections that may be required. This will save you time next month!

2) With this Disk A still in Drive A, put your HUG COBOL Corner Disk-I in Drive B. Using PIP, copy the B:FILEL1.DAT file to your Disk A. Use STAT A:*.* to check for the two (2) files on Disk A.

3) Read about the following COBOL Verbs in your COBOL-80 Reference Manual:

OPEN
CLOSE
STOP
PERFORM

This will help you understand the next "COBOL Corner" article. Of course, we will explain these Verbs as we go. Good luck on your "HOMEWORK". I will see you next month.

EXTENDED SCREEN EDITOR

☞ Extended Screen Editor is a powerful full screen editor designed specifically for the Z100 keyboard. It is a versatile tool, with many daily uses. Use ESE to write programs... to compose letters... to type charts which are wider than the screen... to enter data... to create and modify files of any length.

☞ ESE runs under all versions of ZDOS. Its speed when refreshing the screen is even faster than the rate provided by the operating system because optimized assembly code is called. So, you can jump quickly from one screenful to another.

☞ Every disk comes with a tutorial demo to sit back and watch. (Ask to see it at your local dealer.) You get a printed manual. There are plenty of plain English prompts to guide you. The HELP key brings information on a variety of topics, and you can expand the 'help' file to include topics of your choice.

☞ At no extra charge, you will receive CTEXT, a text formatter that beautifies margins and paginates documents before printing. CTEXT now allows you to print a personalized letter and envelope for each person on your mailing list.

Outstanding Features

SPLIT THE SCREEN TO VIEW TWO FILES SIMULTANEOUSLY.

SCROLL THE SCREEN RIGHT AND LEFT TO WRITE WIDE LINES.

THE USER MAY ALTER WHICH KEY PERFORMS ANY COMMAND.

SET TABSTOPS FOR ALL FUTURE EDITING, IF DESIRED.

CONFIGURATION OPTIONS ARE BUILT INTO THE EDITOR.

$49.97

Quantity discount available.
Cherry Engineering

Phone this evening!
(301) 498-4236

CLIP OUT AND MAIL

Quality software at a fair price.

Send more information about ESE____

Send____ copies of ESE, $49.97 ea.

Send____ demonstration disks with manual, $15.00 ea.

Name ____________________

Address ____________________

(Make check payable to Cherry Engineering.)

PO Box 216, Laurel MD 20707

DISK DRIVES DISK DRIVES DISK DRIVES DISK DRIVES DISK DRIVES DISK DRIVES DISK DRIVES DISK DRIVES

PRICE BREAKTHROUGH

Super Sale on New Disk Drives

Starting at **$169.00!**

Tandon — Siemens — Remex — MPI — Teac — Shugart — Tabor

40 or 80 Tracks — Single or Dual Head — New 3½″ Drivette™
Our Disk Drives Are Capable Of Single **And** Dual Density Operation

The **NEWEST** Technology Capable Of Operating On Most Popular Computers

Drive a Hard Bargain!!™ — 5 M.B.-20 M.B. Complete Systems . . . . . . from $999.95

Diskette Breakthrough — 10 Pack in Library Case — $1~~8~~.95 NEW LOW PRICE

Since We Are Always Finding Ways To Save You Money, Please CALL For Our Most Current Pricing.

TOLL FREE ORDERING **1-800-343-8841**

GENERAL AND TECHNICAL **1-617-872-9090**

Model I/III/IV Drives (0 1 2 3)	starting at **$169.00**
Color Computer Drive (0 1 2 3)	**$ Call Toll Free**
Apple/Franklin Compatible Add-On Drives with Case & Cable	Now 40 Tracks ~~$219.95~~ NEW LOW PRICE
Apple/Franklin Compatible Drives with Controller	**$259.95**
Model I/III/IV Memory Upgrade	**Call Toll Free**
Printers — Daisywheel/Dot Matrix	**$ Call Toll Free**
Diskettes in Library Cases	$1~~8~~.95 **New Low Price**
Cases and Power Supplies — (Single-Dual-1/2 Height)	starting at $59.95
Printer **Buffers** 8K to 512K	starting at $143.95
Percom Double Density Controller (Model I)	**$ Call Toll Free**
Holmes Model I/III Speed-up Mod-VID/80	starting at $90.00
Color Computer Printer Interfaces	starting at $29.95
Cables — Printer/Disk Drive	starting at $23.00
DOSPLUS	**$ Special Prices**
Repair Services Now Offered — FAST Turn-a-Round	**$ Call Toll Free**

Warranty on Disk Drives — 6 Months — Extended Warranty $ Call Toll Free

SOFTWARE SUPPORT, INC.

One Edgell Road, Framingham, MA 01701 (617) 872-9090
Hours: Mon. thru Fri. 9:30 am to 5:30 (E.S.T.) Sat. 10 am to 4:30 pm

DEALER INQUIRIES INVITED.

TERMS:
M.C./Visa/Amex and personal checks accepted at no extra charge.
C.O.D., please add $3.00.
Shipping: Please call for amount.
Not responsible for typographical errors.

© Copyright 1983

CANADA
MICRO R.G.S. INC.
751, CARRE VICTORIA, SUITE 403
MONTREAL, QUEBEC, CANADA, H2Y 2J3
Regular **Tel. (514) 845-1534**
Canadian Toll Free 800-361-5155

Service! Service!

All in stock products are shipped within 24 hours of order. Repair/Warranty service is performed within 24 hours of receipt unless otherwise noted. We accept C.O.D., foreign and APO orders. Schools and D&B corporate P.O.s accepted.

Device Drivers and Communication

Michael Porter
Chebeague Marine Enterprises, Inc.
Chebeague Island, ME 04017

I. Making the Connection

In this age of communication, when (we are told) the computer is to take over many of the more mundane operations of our lives (paying bills, typing letters, etc.) and free us for higher things, it is just a bit paradoxical that communications are often the computer user's biggest problem - communications between the various parts of his system, that is. It is not only quite possible, but more or less the norm to have a computer and a printer which will not speak to each other, despite the fact that they are theoretically compatible, in that both have RS-232C interfaces. And the most ironic aspect of this situation is that it can be very easily corrected, particularly in HDOS, where printers (and all other peripheral devices) are controlled by individual driver programs which are quite small and thus easy to modify.

A number of patches to accommodate the Heath drivers to particular printers have been published in REMark and in the newsletters, but suppose your printer is not the exact one mentioned in the article. You could (probably) buy a driver from one of the small software houses which would do the job, but then you wouldn't have the satisfaction of doing it yourself, and that was one of the reasons for buying a Heathkit in the first place. Then too, some printers are not common enough (in this country anyway) to have achieved widespread support, so even if you might be able to make it work, you wouldn't be able to use all the features that made it look like a good thing to buy. (This phase error happens to me quite often; when everyone else had big cars with teeth, I had a Peugeot and could never get it fixed, and now that everyone has orange Volvos, I have a Chevy truck. Someday maybe I'll get it right.)

But fear not. With only the most minimal knowledge of assembly language, you can make that XX-100 printer you bought on sale (provided it has a serial interface) work with your H-89, and you can adapt a Heath driver to do it. The HDOS driver format is discussed in REMark, Issue 20 (September, 1981), with particular attention to the relationship between the driver and the SET program. The advantage of starting with one of the drivers supplied with HDOS is that the basic driver format is taken care of, and thus there are fewer chances to make a mistake. The second section of this paper discusses the choice of SET options and their implementation in multi-unit drivers, and requires some knowledge of assembly language. But in this section, we will concentrate on the relationship between computer and printer, as it is expressed in and controlled by the driver, and although some theoretical background has to be covered before we can get on to the good stuff, there is really very little knowledge of assembly language required.

In the beginning, there are several variables that your computer and printer must agree on before any communication can take place: the data transfer ('baud') rate, the 'output port' which corresponds to the particular plug on the computer where your printer is plugged in, the number of bits forming a data 'word' or character, the number of 'stop bits' marking the end of each transmitted character, and finally, the way the printer signals the computer when it can't take any more characters and needs a break. This last is called 'handshaking'. Since it is the source of many difficulties, we shall come back to it in greater detail.

Let's start with baud rate. The requirement is that the computer must transmit data at the rate expected by the printer, and that rate is buried somewhere in the printer manual. Note that I don't say it will be clear or obvious. I once found the appropriate baud rate mentioned in a footnote in the corner of an illustration showing the RS-232 signal levels, but it will be there somewhere. If the printer baud rate is not fixed, there will be a DIP switch somewhere down in its depths and the manual will tell you how to set it. Once the rate is set at the printer, all you have to do is use the SET command under HDOS; set the driver to the same baud rate as the printer (as in SET LP: BAUD 4800).

The data format (number of bits per word and number of stop bits) is really mentioned here only for completeness' sake. All printers I have dealt with have either been preset, or could be set by a combination of switches to accept eight-bit words with one stop bit (except two stop bits at 110 baud). And since this is the Heath standard as well, there is no problem here.

Setting the proper port number is really an internal matter for the computer, rather than an aspect of the relationship between computer and printer, but it is one link in the chain and thus one more chance for something to go wrong. All that is involved is making sure that the driver actually sends data to the port where your printer is plugged in, rather than somewhere else. On an H89, the three ports on the serial interface card are numbered (from top to bottom) 340, 320, and 330, and all we have to do is verify the number and then use the SET command to tell the driver about our choice (e.g. "SET LP: PORT 320").

So far, so good; we are sending the proper kind of data at the speed the printer expects, and we are actually sending it where it is supposed to go, so our printer should work, right? Well, the answer is, as many people have discovered, "not necessarily"! This is where handshaking comes in. If the driver thinks the printer is busy when it isn't, we won't get any output no matter how long we wait, and conversely, if the computer sends more data than the printer can handle, we will get something printed, to be sure, but we will lose some characters, or some will be garbled. To avoid this problem, the RS-232C standard includes some control lines from printer to computer, as well as the data lines. There is nothing very standard about the way printer manufacturers use these lines (alas!), but the lines most often used are those connected to pins 4 and 20 of the [DB25] connector on our printer cable. These are called Request to Send ('RTS') and

Data Terminal Ready ('DTR'), respectively.

My Olympia printer, for example, uses RTS (pin 4) as its control line. This line is 'active' ('on', or set to a logic '1') whenever the printer can accept data (this is logical; the line is called 'Request to Send', remember). When its buffer is full or the printer interface is busy, the printer switches the RTS line to a logic '0', or 'off', and this must be read and interpreted by the HDOS driver program as a command to wait until the printer has room again before sending more data. Well and good, but the H-14 uses just the opposite 'protocol', as it's called. The H-14 keeps the RTS line 'off' until its buffer is full, then it turns RTS 'on'. The H-14 driver, which has to interpret this, would simply not work with my Olympia printer because it would be getting what it considered a busy signal all the time. And just to complicate matters, many other printers don't use RTS at all, but use the DTR line (pin 20) in just the same way.

Fortunately, however, we don't have to get out our soldering irons and switch wires around in the cable to get the right signal in the right place. Instead, the integrated circuit which takes our data from the computer and sends it to the printer (the 8250 Asynchronous Communications Element, a remarkably clever device) not only takes care of such matters as the proper baud rate and data format, but also provides us with a way to check either the RTS or the DTR line, or both if we wanted to. The last few pages of the H-89 manual has a lot of information about the 8250 in outline form; we can't really even begin to consider all its functions without taking over the whole magazine. We are chiefly concerned at the moment with the 8250's 'Modem Status Register', which contains our two signals, RTS and DTR, along with some others, and which can be read by the computer's Central Processing Unit (CPU) at a port which is numbered 6 above the base port. If our printer is plugged into port 340, for example, the CPU will read the Modem Status Register at port 346. This is an eight-bit register just like those in the CPU itself, and the CPU 'reads' it by transferring its contents to the CPU's own Accumulator with the instruction "IN 346Q". In the Modem Status Register, bit 4 holds the state of the RTS line while bit 5 holds the state of the DTR line, and never mind that the documentation (the H-89 manual or Heath's driver source code) labels these bits Clear to Send (CTS) or Data Set Ready (DSR); these are just the same lines looked at from the other direction, from the point of view of the computer (which is sending) rather than that of the printer (which is receiving).

Now, how does the driver make use of this information? Consider the source code for the Heath LPH24 driver, as supplied with HDOS 2.0; this driver is intended for the Heath printers which turn the RTS line 'on' as their busy signal. Well, down in the listing there is a little routine called "WAIT" which does all the work, and in assembly language it looks like this:

```
WAIT    PUSH    H
WAIT0   LDA     SCAADR+1      Check to see if a control character
        ANA     A             has been hit (ctrl-A,-B,-C,-Z) and
        JNZ     WAIT3         if one has, jump to return

        LDA     TLP.POR       Get the port nr. from storage to
                              register A
        MOV     H,A           put it in H
        MVI     L,UR.MSR      L = the offset (6) to Modem
                              Status Register
        CALL    IN            the routine "IN" reads port (H+L) and
                              returns with its contents in register A

        ANI     UC.CTS        AND the contents of the Modem Status
                              Register with the bit which reflects
                              the RTS line--the result will be
                              the status of that line (1 or 0)

        JNZ     WAIT0         If the result is not 0 then the RTS line
                              is 'on', indicating busy, and we jump
                              back to the beginning to try again

WAIT3   POP     H             If the RTS line is 'off,' then the
        RET                   result of the AND was 0 and we don't
                              jump back but come here instead
```

This routine checks the RTS line and keeps on checking it until it is turned 'off', indicating that the printer can take another character; it then returns to the output routine which called it.

Now, if we have a printer (like my Olympia) which keeps the RTS line 'on' except when busy, we just change the instruction "JNZ WAIT0" to "JZ WAIT0", so that we loop back to try again if RTS is 0 instead of when it is 1. If our printer uses the DTR line (pin 20) instead of the RTS line (pin 4), we have only to change the line "ANI UC.CTS" to "ANI UC.DSR" so that we isolate and look at bit 5 instead of bit 4. Again, if the line is 'on' when ready for data, we use "JZ WAIT0", while if it goes 'off' when ready, we would use "JNZ WAIT0". UC.CTS and UC.DSR are the Heath symbolic equivalents contained in the file "U8250.ACM", and the assembler knows what they mean because there is a statement (XTEXT U8250) early in the driver source code which tells the assembler to read that file and take note of its contents. The practical result for us is that we don't have to remember which is bit 4 or 5, the assembler does all that for us.

Communication between computer and printer using either the RTS or DTR line is often referred to as 'hardware handshaking', presumably because the 8250 (once set up properly) takes care of most of the details without any more effort from the programmer. Another method, somewhat less widely used, but apparently becoming more common with newer printers, is 'software handshaking', in which the printer sends commands to the computer over the data line when it is time to stop or start sending characters. One set of instructions commonly used is XON/XOFF, which is also called Ctrl-S/Ctrl-Q, or DC1/DC3. All of these mean the same thing. The printer sends a Ctrl-S to the computer when its buffer is full in just the same way as we type it on the H-89 keyboard to stop a listing so we can read it. The computer must then wait until the printer sends a Ctrl-Q before sending more characters. This requires that the computer monitor the data line from the printer and then react properly to these instructions. We might replace the "WAIT" routine from the Heath LPH24 driver with one that looked something like this:

```
WAIT    CALL    INCHAR        look for a character from printer
        JC      WAIT2         CARRY flag set indicates no character
                              received, so it's OK to proceed

        ANI     177Q          we have a character, so strip off the
                              parity bit

        CPI     CTLS          see if it is Ctrl-S
        JNZ     WAIT2         if not, don't bother with it, just
```

```
                                  return

                              apparently we do have Ctrl-S, so now we have
                                  to look for a Ctrl-Q

        LDA     SCAADR+1          first, check for an abort character
        ANA     A
        JNZ     WAIT2             and return if we have one
        CALL    INCHAR            look for a character
        JC      WAIT1             this time, we must have one before
                                  we can go on
        ANI     177Q              got one, strip parity bit
        CPI     CTLQ              see if it is Ctrl-Q
        JNZ     WAIT1             if not, keep trying until we get it

WAIT2   RET                       we return:
                           a. if no Ctrl-S was sent
                           b. if some character was sent, but not Ctrl-S
                           c. if we have Ctrl-Q after a prior Ctrl-S
                           d. if the user typed a control character
                              (usually to get out of the program)
```

This "WAIT" routine can be called for each character, like the RTS/DTR "WAIT," or it can be called every so many characters, depending on how full the printer buffer is before the printer sends the Ctrl-S. If, for example, the printer sends Ctrl-S when 256 spaces remain in the buffer, the computer can send data in 'bursts' of 256 characters without any fear of overrunning the buffer. This speeds things up quite a bit because the computer doesn't have to check every time, and it gives the 'smart' printers bigger blocks of data to exercise that intelligence on. Bi-directional printers, for example, need to be able to 'look ahead' in order to find the shortest route to the next place where they have to print.

Another problem which has sometimes arisen when trying to use the Heath drivers for other printers is the presence of a few spurious characters at the beginning of a printout. These characters are part of an escape sequence sent by the "OPENW" code which prepares the printer to write. They are understood by the H-14's processor as instructions setting the character width and lines per inch. To adapt a driver like this to a printer which is not 'software-controlled', one like my old Selectric, for example, whose options (such as they are) are set by mechanical or electrical controls on the printer itself, we need only find the "OPENW" processor in the driver source listing (the label in the left column is "LPOPENW") and delete the line "CALL INITLP". We could also delete the routine "INITLP" itself, but that is not necessary except to save space.

On the other hand, if our printer is software controlled, we can use the format of "INITLP" to send our own setup codes to the printer. My Olympia, for example (I hate to sound like a broken record, but most things are clearer with concrete examples and I don't like to write about things I haven't tried), interprets the ASCII characters "ESC CR CTRL-E" as an instruction to print 12 characters per inch, or elite. The initialization routine takes the variable, in this case characters/inch, and plugs it into a preset string of characters, which is then sent to the printer, thus:

```
INIT     LDA     CPI          get the preset variable corresponding
                              to the characters/inch we want
         STA     INITB        put it into the string
         LXI     D,INITA      DE points to the text to be sent
         LXI     B,INITLEN    BC holds the nr. of characters to send
         CALL    WRITE        send them, using the regular output
                              routine
         RET

INITA    DB      ESC,CR       constants for chars/inch instruction
INITB    DB      0            variable
INITLEN  EQU     *-INITA      length of character string to send
```

The label CPI itself refers to a particular byte at the end of the driver code, one of a group of such bytes which store the variables (port, baud rate, lines/inch, etc.) used by the driver; it is in these bytes that the driver's SET options reside, and what SET does is to set these values as directed. Now that our connection is established, it is time to consider what we want the driver to do for us. In the next section of this article, we'll consider the choice of SET options and outline the structure of a multi-unit driver.

II. SET Options and Multi-Unit Drivers

Now that we have a working driver, at least in the sense that our printer should print what is sent to it, it is time to consider both how to use our printer's abilities (variable pitch, etc.), and how much work we want the driver (and printer) to do for us. Do we want it, for example, to put page breaks in a file automatically, to expand any tabs in the file to some number of spaces, or to wait between pages so we can put in a new sheet? Generally speaking, under HDOS this sort of formatting is carried out by the driver itself (although some of the newer printers could do a good deal of it). Answers to the yes-or-no questions are stored in a 'flag byte' where each bit holds the selected answer to a particular question, while values like the chosen lines-per-page are stored in a table at the end of the driver. The formatting work is done by the routine CPOUT which also takes care of such things as adding a carriage return (CR) to the linefeed (NL) character which HDOS uses to mark the end of lines in a file. This routine is also a good place to put any changes needed to adapt the H-89 keyboard to a different typewheel or element.

On many new printers, functions which used to require setting switches on the machine itself have been made 'software-controlled', which is to say that the machine will switch itself from 10 to 12 characters per inch, for example, on receiving the proper instructions. Other instructions may set tabs, strike each letter twice for a bolder effect, or shadow each letter by moving the printhead slightly between double strikes. These instructions are generally in the form of Escape Sequences, which means only that each instruction consists of one or more ASCII characters preceded by the ESC character. The ESC signifies to the printer that the next character does not carry its normal ASCII significance, but instead has the meaning given it in the printer's private code. The Olympia, for example, interprets the combination ESC BEL as a command to roll the platen down a half-space, putting it in position for a superscript, while ESC CR nn defines the horizontal spacing step (or pitch) where "nn" is the number of 1/60" steps required to make up the desired pitch.

The driver's responsibility for functions which the printer is meant to carry out (character pitch, line spacing, etc.) ends with initializing the printer by sending it the proper commands, but the driver processes commands itself in a number of different ways. A command setting a certain number of lines per page, for example, is typically handled by comparing the line count to the predetermined value at each CR and adding a form feed to the file when enough lines to fill a page have been sent, while a command like

'FORM' (send form feed on close) usually requires the driver to check a flag byte for the relevant flag and respond appropriately. Still other commands (TABX, for example, which expands tabs to the proper number of spaces) are handled by letting the initialization routine modify the output routine so that certain sections of code are either skipped or executed, depending on the command.

"O.K., but how do these functions get into the driver and how do you choose which ones to use? And what do you mean by 'respond appropriately'?"

Well, the choice of functions is up to the user, but perhaps it will be helpful if I list my choices and the reasons for them. My first list of options for the Olympia was almost a page long, because the printer will do quite a lot for you if you let it, but I finally cut it down to a short list which has proven satisfactory for several months now, to wit: three values (PAGE, PITCH, LPI) and four flag bits (AUTO-CR, TABX, PS, SEHTUP). I decided not to bother with making the baud rate and port changeable by SET because I wanted more space in the 'SET Preamble' part of the driver, and because I figured I could always change those values by using Patch. Some options chose themselves; if you have a printer with variable pitch but you can only change it by software (no switch), it doesn't take long to decide that character pitch should be one of your variables. The same applies to line spacing and to the codes which switch the printer's automatic proportional spacing on and off. Similarly, the AUTO-CR function (stolen from the Heath Diablo driver) is an excellent solution to the problem posed by the absence of a distinct 'newline' character in ASCII. If this option is SET, then CR's are passed through, allowing multiple passes over a single line, but line feeds (used by HDOS to mark ends of lines) are converted into a carriage return, line feed sequence for the printer. This is usually what is wanted, but if for some reason we don't, we can always use SET to turn the translation off. PAGE (printed lines per page) is useful for printing unformatted files (ASM source files, for example), and so I included it, as I did TABX (which expands tabs to the appropriate number of spaces). Conversely, there seemed to be little point in including a right margin, or width option, since there could be no guarantee that it would not be exercised in the middle of a word, and the controlling program would have to do all the work anyway.

I hedged my bets, however, by including an option called SETUP which allows me to send any initializing codes I want directly to the Olympia from the H-89 keyboard, so I can set tab stops, or a right margin, or anything else I want. I can even use this option to make the printer into a remotely controlled typewriter, although that is not its primary purpose. My list includes three 'value' options and four 'flag' options (the value options, remember, require that the appropriate value be stored somewhere, while flag options answer yes-or-no questions and use one bit each in a flag byte). The breakdown looks like this:

Value	Flag
PITCH	AUTO-CR
LPI	TABX
PAGE	PS
	SETUP

Our driver, then, has to store four bytes (plus, of course, the baud rate and port number), three values, and one flag byte. Two of the value options, LPI and PITCH, are really instructions to the printer and are sent as part of the initialization process. These are stored (and changed by SET) as the binary equivalent of the appropriate decimal number, like '10' or '12' for characters per inch, but the Olympia deals in horizontal increments of 1/60th of an inch, and requires that the pitch command contain the proper number of 1/60th jumps. The procedure is to divide the number of steps per inch by the desired number of characters per inch. The result is the number of machine steps per character, and this is what we put into the initializing string to send to the printer (the routine is similar for both horizontal and vertical pitch).

```
        LDA     PITCH           A = characters per inch
        MOV     E,A             put it in DE
        MVI     D,0
        LXI     B,HSTEP         B = machine steps per inch
        CALL    $DU66           divide using routine in H17 ROM
        MOV     A,L
        STA     INITB           put result in INIT string
```

The PAGE value, on the other hand, is handled by the driver itself. The character output routine checks the current line number against the PAGE value after every carriage return and sends a form feed when the page is full.

Most of the flag options are handled by the driver as well. Two of them, TABX and AUTO-CR, require the initializing routine to change sections of the character output code to include or skip the relevant sections, while the third determines whether or not the routine which accepts characters from the keyboard and sends them to the printer is executed or not. However, the fourth flag option, proportional spacing, is again something which the printer itself handles. The procedure for all of these options is to check the flag byte for the flag we're interested in. Then if we find our flag, we do whatever it requires, like this:

```
        LDA     CP.FLAG         get the flag byte
        ANI     F.SETUP         check for our flag (manual setup codes)
        CNZ     SCP             call the setup routine if flag is there
        ...     etc.
```

Most of these options are available on one or another of the drivers that come with HDOS 2.0, and the details can be studied there, but the routine to send manual codes to the printer might be worth a closer look. There were two difficulties in writing this routine. The first was to avoid all HDOS processing (we might want to use CTRL-C, for example, as part of an escape sequence, and we certainly would want to distinguish between the Carriage Return and Line Feed characters), and the second was to devise a way to tell the computer when we are finished, without using a character we might want to send to the printer. The solution to the first problem is to use an undocumented feature of HDOS. In the 'RAW' mode, HDOS takes characters just as the user types them and puts them in the console buffer, where a '.SCIN' can find them. The raw mode is selected by setting a bit in S.CSLMD, just like line or character mode, and its symbolic equivalent (CSL.RAW) is found (like other console options) in the file 'ESVAL.ACM'. The second problem can be resolved by using some character rarely used for our setup codes, and then using it twice, just to make sure. I chose the DELETE key which my printer hardly uses and which had the added advantage of seeming logical. (Theoretically, we should be able to use the BREAK key to generate an interrupt, but I have never been able to make this work consistently on my machine.) The code looks like this:

```
        Send setup codes to printer

SCP     LXI     H,S.CSLMD
        MOV     A,M
        STA     SCPA                            save current CSLMD
        MVI     A,CSL.RAW+CSL.ECH+CSL.CHR       set RAW mode, no
                                                echo, character mode
        DI                                      no interrupt until set
        MOV     M,A
        EI
        CALL    $TYPTX
        DB      'Setup Codes?',NL,':'+200Q    print prompt

SCP1    CALL    GETCHR                          read a char. from TT:
        CPI     DEL                             might be 1st DEL
        DB      JRZ,SCP3-*-1                    if so, look for 2nd

SCP2    CALL    CP09                            not DEL, send to printer
        DB      JR,SCP1-*-1                     back for more characters

SCP3    CALL    GETCHR                          was DEL, try next char.
        CPI     DEL
        DB      JRZ,SCP4-*-1                    2nd DEL, jump to exit
        STA     SCPB                            not DEL, save it
        MVI     A,DEL                           send the DEL that got us
        CALL    CP09                            here
        LDA     SCPB                            send next char & loop
        DB      JR,SCP2-*-1                     for more

SCP4    LXI     H,S.CSLMD                       get original CSLMD
        LDA     SCPA
        DI
        MOV     M,A                             put it back
        EI
        RET                                     go home

SCPA    DB      0                               original CSLMD

SCPB    DB      0                               char. after DEL
```

We don't really need to know just how the SET program goes about changing our various flag and option values, but we must know the format required to use SET, and fortunately both are discussed at length in REMark, Issue 20. There is much to be gained by studying the source code for the various drivers Heath supplies. Once one is used to the rather cryptic Heath style, the use of SET to modify drivers is quite well documented and actually fairly clear.

Having everything under software control (as opposed to setting switches on the machine) has many advantages. Perhaps the main one is that a program can set the machine the way it wants to without the operator's having to bother (and perhaps to forget). It has the disadvantage, however, that one either has to use the SET command every time before printing anything or have a number of different copies of the driver on the system disk, each with a different configuration. The first is tedious, and the second takes disk space and could crowd HDOS, whose capacity to accept drivers is limited by the size of the device table. A better solution is to have the same driver serve several units with independently SETable options, like the HUG/UltiMeth SY: driver; there is nothing that says the different 'units' cannot all be differing configurations of the same physical machine. There is no theoretical reason, either, why the various units need to be limited to the same physical machine. It is just that different machines vary enough in their requirements so that a truly general driver might have to be quite large, and that my need (and maybe that of most people) was for different configurations of the same printer, since I couldn't afford more than one.

There are only a few changes required in our basic driver in order to give it the ability to handle more than one unit. We have to change the header - which tells HDOS what is going on, we have to make provision for SETting the various units independently, and we have to modify the initialization routine so both machine and driver are properly set up.

First, the header. The first 5 bytes of the disk file are the "PIC" header, which tells HDOS that the file is relocatable and how long it is; these bytes are automatically generated by ASM in response to the "CODE PIC" instruction. It is the next 15 bytes that we are concerned with. In these bytes, the driver identifies itself to HDOS as such and tells HDOS both how many units it serves and what those units can do. Certain bytes of the header depend on the number of units we want to use; up to eight units are possible (numbered 0 through 7). If, for example, we decided to use four units for our multi-unit driver, these bytes would be defined like this:

```
*       CODE HEADER

        DB      DVDFLV          307Q--arbitrary flag to tell HDOS
                                this is a device driver
        DB      DT.CW           general device capability: can write
        DB      00001111B       DVD.MUM:  this byte must have a 1 for
                                each unit we are going to use--
                                example shows 4 units mounted
        DB      4               DVD.MNU:  max. nr. of units
                                (in this case, 4)

*       the next 8 bytes define the units' specific capabilities, one
*       byte for each unit.  since we have 4 units, we only need
*       4 bytes, but space must be left for the others anyway

        DB      DT.CW           each unit capable of write only
        DB      DT.CW
        DB      DT.CW
        DB      DT.CW
        DS      4               space for the other 4 possible units
        DB      DVDFLV          repeat the flag to show SET that
                                driver can take SET options
        DW      0

        etc.
```

If we wanted to use more than four units, we would change those bytes that refer to the number of units, remembering to keep the total unit capability block (used + unused) at eight bytes.

Besides changing the header, we also have to change the way the various options are stored by the driver. Instead of having just one flag byte and one storage location for lines per inch, etc., we will need one for each unit. And if we arrange these in a table, they will look like this (note that we make the assembler define for us the offset to each value from the base address): ⟶

```
CP.UTAB EQU     *                       Table address

I.UTFLG EQU     *-CP.UTAB               offset to flag byte
        DB      FLAG0                   flag, unit 0
I.UTLNP EQU     *-CP.UTAB               offset to lines / page
        DB      LNPG0                   lines / page default
I.UTPIT EQU     *-CP.UTAB               offset to pitch
        DB      P0                      pitch default
I.UTLPI EQU     *-CP.UTAB               offset to lines / inch
        DB      LPI0                    lines / inch default
CP.UTEL EQU     *-CP.UTAB               length of entry in table for
                                        each unit

        DB      FLAG1,LNPG1,P1,LPI1
        DB      FLAG2,LNPG2,P1,LP12
           etc.
```

The values 'LNPG0' or 'FLAG2' are the symbolic equivalents of the default settings which are listed at the beginning of the program, where they are easy to find and change.

If we never wanted to change anything, we could stop here - we'd only have to make sure that the default values were the ones we wanted to work with and we'd be all set. Unfortunately, things have a way of coming up, and today's ideal values may be hindrances tomorrow. This situation is what the SET program was designed for. Now, the tables SET uses ('OPTTAB' and 'PRCTAB' in the driver source code) are designed to use a single address for the byte to be changed, so it is not convenient to simply go into the unit table and change the particular value we want (maybe LPI for CP3:). Instead, we find the entry we want and copy it out to the special table 'SETTAB', which has just room enough for the data for one unit (four bytes in our example). How do we find the entry we want? Well, when SET enters the driver code, the unit requested is in register 'A', the accumulator, and the procedure is just to multiply this unit number by the length of the data for each unit and add the result to the address of the table so that we end up with the address of the data specific to the unit we want to SET. Because we want to save space here, we use a few special Z-80 instructions (although the same thing could perfectly well be done in 8080 code) and the new SET entry code looks like this: ⟶

```
SETNTR  EQU     *
        DB      EXX             use Z-80 alt. registers instead of
                                pushing everything
        LXI     D,CP.UTEL       DE = table entry length
        PUSH    D               save it
        CALL    $MU86           multiply it by unit nr. (already in A)
                                result: HL = offset to unit data
        LXI     D,CP.UTAB       DE = table address
        DAD     D               HL = unit data address
        POP     B               BC = entry length
        PUSH    B               save it again
        LXI     D,SETTAB        DE = destination for data
        DW      LDIR            move it
        POP     B               entry length back to BC
        DB      EXX             save all values & back to primary regs.
        MOV     B,D             now continue with standard Heath
        MOV     C,E             SET entry code
        LXI     D,PRCTAB
        LXI     H,OPTTAB
        CALL    $SOP            change the specified value
        RC                      return, making no changes if any error

        DB      EXX             back to alt. registers
        DCX     D               backup DE & HL each to point to top
        DCX     H               of our data
        XCHG                    swap to get source & dest. right
        DW      LDDR            move new data back to table
        DB      EXX             get primary regs. back
        JMP     HELP0
```

This change is all that is really necessary before we can SET values for our several units, but readers who were paying attention will note that the segment of code just quoted ended with a jump to a routine called 'HELP0' instead of a return to the main program. Admittedly, it's a frill, but with all these units, it seemed as though it would be nice to have the selected options for each unit printed out on the screen, both in response to the command 'SET CP: HELP' and after any option was SET.

The screen display produced looks like this: ⟶

```
UNIT    TABX    AUTO-CR SETUP   PS      PAGE    PITCH   LPI

0       Y       Y       N       N       000     10      6
1       Y       Y       N       N       000     12      6
           etc.
```

There is nothing very fancy about this display (the "sophisticated programmers" who appear here and there throughout the Heath documentation will no doubt see at least three ways to improve it), but it does serve the dual purpose of reminding me of the options available and also of their current state, and it is produced by the following code, which checks the unit table and puts appropriate values in the message.

```
HELP0   LXI     H,CP.UTAB-CP.UTEL
        PUSH    H
        DW      POPIX               IX pts. to source of values (unit table)
        LXI     H,UMSG              HL points to destination (message)
        MVI     B,UNITS             B = nr. of units (nr. of times through
                                        the routine
HELP1   PUSH    B                   save it
        INX     H
        INX     H                   HL points to first flag location
        LXI     B,CP.UTEL           BC = length of unit data
        DW      DADXB               IX points to next unit data in CP.UTAB
        MVI     B,FLAGS             set B up to count flags
        DW      LDAIX
        DB      I.UTFLG             flag byte to A

HELP2   RAR                         move flag to Carry bit
        MVI     M,'Y'               assume flag is set
        DB      JRC,HELP3-*-1       if it is, jump to next
        MVI     M,'N'               otherwise put 'N' in message
HELP3   INX     H
        INX     H                   advance HL to next flag location in msg.
        DB      DJNZ,HELP2-*-1      loop for all flags
        DW      LDCIX
        DB      I.UTLNP             Lines / Page to BC
        MVI     B,0
        MVI     A,3                 3 digits in result
        CALL    $UDD                convert to decimal, using routine in ROM
        INX     H                   point to next message location
        DW      LDCIX
        DB      I.UTPIT             Characters / Inch to BC
        MVI     A,2                 2 digits
        CALL    $UDD                convert it
        INX     H                   point to next message location
        DW      LDCIX
        DB      I.UTLPI             Lines / Inch to BC
        MVI     A,1                 1 digit
        CALL    $UDD                convert it
        INX     H                   point to next message location
        POP     B                   get unit counter back
        DB      DJNZ,HELP1-*-1      loop for all units
        CALL    $TYPTX              print message
SETMSG  EQU     *
        DB      NL,NL,'UNIT',HT,'TABX',HT,'AUTO-CR',HT,'SETUP',HT
        DB      'PS',HT,'PAGE',HT,'PITCH',HT,'LPI',NL,NL
UMSG    DB      '0',HT              Unit nr.
        DB      'Y',HT              Tabx
        DB      'Y',HT              Auto-cr
        DB      'N',HT              Setup
        DB      'N',HT              Form
        DB      '000',HT            Page
        DB      '10',HT             Pitch
        DB      '6',NL              Lines/Inch

        DB      '1',HT
        DB      'Y',HT
        DB      'Y',HT
        DB      'N',HT
        DB      'N',HT
        DB      '000',HT
        DB      '12',HT
        DB      '6',NL

...     etc.   ...    ...    etc.   ...

        DB      '3',HT              unit 3 data
        DB      'Y',HT
        DB      'Y',HT
        DB      'N',HT
        DB      'N',HT
        DB      '000',HT
        DB      '15',HT
        DB      '8',ENL

HELP    XRA     A
        RET
```

One effect of printing the HELP message after every use of the SET command is that the request 'SET CP: HELP' itself just returns, taking no action. This is because the HELP command is actually executed after every option. Otherwise the HELP message would be printed twice, which would be at the least unaesthetic.

This is all well and good, but how does our multi-unit driver know which unit is being called? What good would all these units be if we couldn't call (write to) any one particular unit? The answer to this question (buried in the HDOS documentation) is that when any Input or Output operation is taking place (including OPENing a file), the unit number requested is in the memory location 'AIO.UNI' (041.061), and all we have to do is get it from there. As before, we have to dig the data for the specific unit we want out of the table, and then somehow get the necessary data to the printer. The initialization routine (called by 'Open for Write') seemed like a good place to do this, and the source looks like this.

(See the program listing on following page.)

From here, the initialization routine goes on to make the modifications to the output routine 'CPOUT' required by the flags which are set for the unit we have called (including or skipping the code which processes tabs, for example), and it calculates the codes to be sent to the printer to set the desired character pitch and line spacing, using the values it finds in the unit table.

These changes are all that is necessary to convert a standard Heath driver into a multi-unit driver which is much more convenient to use, even if you only have an H-14. You'll never have to swear because you forgot to re-SET your LP driver before printing a listing, and the printer is fighting ASM for control of the page breaks. There are hundreds of possible refinements, any of which would be an improvement for a particular situation, but this works for me and has stood up well under a number of unforeseen pressures. I hope some of you find it a reasonable framework to build on.

```
INITCP  EQU     *
        LDA     AIO.UNI                 A = unit nr.
        LXI     D,CP.UTEL               DE = unit table element length
        CALL    $MU86                   HL = offset to unit data
        LXI     D,CP.UTAB               DE = table base
        DAD     D                       HL points to unit data we want
        PUSH    H
        DW      POPIX                   now IX points to unit data
        DW      LDLIX
        DB      I.UTFLG                 L = unit flag
        DW      LDHIX
        DB      I.UTLNP                 H = unit lines / page
        SHLD    CP.FLAG                 store where CPOUT can find them
        ERRNZ   CP.LNPG-CP.FLAG-1       make sure we haven't goofed
                  etc.
```

The Winner of the 1983 Article of the Year Award is David Warnick

Dave has provided HUG members, through REMark, an enjoyable and informative series of articles. A Heathkit weather station Model ID4001 will soon be on its way to Dave.

Congratulations Dave!

Controlled Data Recording Systems Inc.

ANNOUNCING THE FDC-H8

DOUBLE DENSITY 8" AND 5.25" CONTROLLER FOR THE H8 COMPUTER

Has all of the capabilities of our popular FDC-880H controller, with the added features of;

- Direct memory access (DMA) data transfer.
- Hard sectored controller (H17) incorporated on the board.
- Runs with the standard 8080 CPU card and with Z80 CPU upgrades.
- Accesses both hard sectored disk formats and soft sectored disk formats through the same drives attatched to the FDC-H8 without hardware additions. **Price $495.00**

NEW PRODUCTS FOR THE FDC-880H

DM-1 DUAL BOARD MODIFICATION KIT **$29.95**
Allows for both the FDC-880H and the H88-4 controller cards to interface with the same 5.25" drives. Drives will run as both hard sectored format and soft sectored format depending upon the logical drive letter.

CDR BIOS by Livingston Logic Labs **$60.00**
Enhanced version of Heath/Zenith CP/M 2.203 BIOS with ZCPR. Supports all Heath/Zenith disk formats through the FDC-880H and the H17 controllers.

CDR DVD by Livingston Logic Labs. **$40.00**
HDOS driver for running double density HDOS through the FDC-880H

Shugart Slimline 5.25" 40 track double sided drives **$275.00**

Shugart Slimline 8" double sided drives **$525.00**

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

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

Using the ANGEL Print Buffer

William M. Adney
P. O. Box 1477
La Mirada, CA 90638-1477

When you get home after a hard days work and want to spend some time working with your computer, it's frustrating to have to wait for printing to be completed before you can use the computer again. Although some word processors, including WordStar, have built-in print spoolers, I personally prefer the Zenith LIST command for program listings. It prints the file name and page numbers at the beginning of each page plus I also use the "D" parameter to include the date that the listing was printed.

Print spoolers have been around a long time, but they have at least one serious disadvantage. In many cases, they "go to sleep" during the execution of some application programs. For example, if you want to print a file with Digital Research's DESPOOL and also want to run the Microsoft BASIC interpreter at the same time, you will find that printing stops until you return to the CP/M command prompt. Although there are a number of technical reasons for this, the bottom line is that you still have not saved a lot of time by using a print spooler. I concluded that the solution to this problem was a hardware print spooler which contains its own "computer" and essentially operates independently of the main unit.

Reviewing Print Buffers

I spent a considerable amount of time looking at various print buffers by comparing all of the features versus the price of the unit. In order to do this, I requested literature from each of the manufacturers for comparison purposes.

At that point, I had a good understanding of the available features, so I was ready to define my own requirements for a print buffer. My basic requirements included page and copy reprinting plus the maximum amount of memory for the dollars spent. Although I currently have an H-25 with a serial interface, it seemed like a good idea to have a buffer that would also handle a parallel interface since I am planning on getting a letter quality printer, and I've noticed that the parallel interface units are cheaper.

As a result of this comparison, I found a rather remarkable unit which satisfies all of my requirements: the ANGEL Print Buffer from Ligo Research. For its list price of $295.00, it has some capabilities that no other single unit has at any price!

ANGEL Standard Features

Although the ANGEL is a small unit, its size is deceptive in terms of its capabilities. You get a memory size of 64K, baud rates which range from 150 to 19,200 baud, various serial and parallel interface combinations, plus all of the normal page reprint and copy functions. Note that the 64K memory size is a standard feature at no extra cost. All serial/parallel combinations are available in one unit so you don't have to worry about a future problem with computer/printer interfaces. That means that there is no extra cost for another buffer if you decide to change any of your hardware interfaces.

There are a number of command functions available with the ANGEL. Eight membrane key switches provide a rather impressive array of commands which extend to about 19 actual command functions. The basic commands, sufficient for most work, include Pause, Hold, Clear, Copy, Page Skip, and Page Reprint.

Pause suspends the data transmission to the printer although the H-25 doesn't stop immediately because it has a 2K internal buffer. The Hold function suspends the reception of data from the computer. Pressing either key a second time releases the pause or hold command.

Clear does the obvious...it clears all data in the buffer. When the Copy key is pressed, you can reprint all data that is currently in the buffer. When I want to make multiple copies of letters, articles, etc., I press the Clear key before I begin printing so that I don't have any extra data in the buffer. Then I use WordStar, LIST, or PIP to print the file one time. Extra copies are generated by simply pressing the Copy key. Isn't that neat? And it really does save an incredible amount of time!

The Page Skip feature allows you to skip the current page and print from the beginning of the next page. Page Reprint will reprint from the beginning of the current page, or if you're at the beginning of a page, it will reprint the previous page. A Page-Pause mode is also available which allows you to stop printing at the end of each page for single sheet feeding into the printer. Most word processors also have that feature, but I found that this is a lot easier and more convenient.

A number of additional commands provide versatility that I think is very unique. For example, you can print multiple copies in magnitudes of tens or hundreds. The unit also provides for self-test diagnostics which can be very helpful.

If that isn't enough, it also provides a capability to perform hex dumps in both hex and ASCII formats for programmers. In short, you get a lot of "bang for the buck" with the ANGEL.

These are the features which are advertised, but what about the ones that are not? The ANGEL also has a "page compression" mode which effectively increases its memory capacity beyond the advertised 64K. It appears that it could provide about 128K effective capacity, but I couldn't find any way to accurately measure that. However, I was able to print a 106K file in about 40% of the normal print time using this mode.

Six light emitting diodes (LED's) are used to provide the status of the unit. One is a "power-on" light, and the other five are used to indicate modes, status, and provide information on the test diagnostics.

Setting It Up

It seems that each printer manufacturer has to come up with their own ways of interfacing with a computer. Heath is no exception to this. For those of you who don't know it, there is nothing standard about the RS-232 interface except the signal types and the physical connectors. For some strange reason, standardization of pins for the signals is totally non-existent between manufacturers. As a matter of fact, some manufacturers (e.g. Kaypro) don't use the same pins on different models of their computers. That kind of interface problem can drive anyone nuts after a while.

Be that as it may, I did have a slight problem connecting the ANGEL to my H-25 because of that standardization problem. Fortunately, I have a "cable matcher" which allows me to reconfigure cables by just moving a jumper. With some help from the nice people at Ligo Research and my trusty cable matcher, that problem was solved in short order. And it turns out that the same cable will work with the H-89 or the H-100. See Figure 1 for the required cable configuration between the ANGEL and the H/Z- 25. Other than that, a standard male to male cable is required to connect the H-100 or H-89 to the ANGEL. By the way, custom cables are available from Ligo Research for about $40.00 each.

```
***********************************************************
*    Figure 1                                             *
*                                                         *
*    From ANGEL              To H/Z-25                    *
*    ----------              ---------                    *
*        1                       1                        *
*        2                       2                        *
*        3                       3                        *
*        20                      4                        *
*                                                         *
***********************************************************
```

The documentation is detailed and certainly provides all of the needed information. I have a review copy of the revised documentation, and it is clearly being improved. Although it may appear somewhat complicated when you first see it, remember that this is a sophisticated piece of hardware. It has a tremendous potential for saving a lot of your valuable time so that any time investment now will be more than repaid.

I think that the ANGEL is a terrific value for the purchase price of $295.00 plus $4.00 shipping (add 6% sales tax for Illinois residents). You can order the ANGEL by using the toll free number 800-323-3304 and charging it to your Visa or MasterCard. Mail orders for the ANGEL should be addressed to:

Ligo Research, Inc.
396 E. 159th St.
Harvey, Illinois 60426

By the way, you can use their toll free number if you need technical support after you buy the ANGEL. A one-year warranty is included for repair or replacement if necessary. In working with them to develop this review, I found them to be very pleasant and helpful. If you are looking for ways to save time, the ANGEL is a good way to improve productivity.

As a side note, I thought you might be interested to know that I received the first ANGEL evaluation unit provided to any magazine. My thanks to Linda Clifford of Ligo Research for her assistance in this area.

Notes On Previous Columns

It seems that there is some difference in opinion regarding the trade-in policy that I mentioned in the December column. While I was in one of the Heathkit stores, I noticed that one of the customers was arguing with the manager regarding the trade-in policy based on my comments in the article. It seems that the customer thought that the Heath stores were required to take trade-ins. That is absolutely not true; it's up to the manager's discretion as to whether or not to accept anything as a trade-in. If that store already has three or four H-widgets on the shelf, it is very unlikely that the manager will want another one. Or if the H-widget has a lot of physical defects, who would want to buy it? Also remember that the maximum trade-in is 50%, and that it may be less than that.

News From Zenith

Since this is a user magazine, I've always thought that it was unusual that new software from Zenith Data Systems (ZDS) was not announced here. Some rather exciting things are in the works, so I thought that I would include them.

ZDS attended ComDex in Las Vegas last November, and there was information that new operating systems would be available for the H/Z-100 series. The same information has been published in a recent ZDS newsletter, Newsdate, but release dates have not been pinpointed yet. If you've seen the current catalog, you probably know that CP/M-86 is already listed. Concurrent CP/M (CCP/M), ZLAN (local area networking), MP/M and Z-DOS 2.0 were also discussed. New math crunching capabilities were announced in the form of an 8087 microprocessor add-on card. As were the new MicroSoft Windows (and mice) which will be supported by the new Z-DOS. PC-DOS 2.0 includes some nice directory commands (e.g. PATH, MKDIR, etc.) as well as some "UNIX like" utilities such as sort. I hope that they're included in the new version of Z- DOS, but please don't call Heath and/or Zenith to find out any information. I guess that we'll just have to wait and see.

Another interesting tidbit is that Zenith has won a military contract for over 29 million dollars worth of Z-100 equipment and software. Why is that important to us? It means that you can expect a lot of support to be forthcoming for the H/Z-100 series. Now if we could just get some RAM disk software...

In The Mill...

I have written to or been contacted by a number of vendors who provide software and/or hardware for Heath/Zenith. In future columns, you can expect to see the results of some of this work, but I thought it might be interesting to share some of the future ideas for this column.

The president of Floppy Disk Services tells me that there are a number of exciting projects that will be of great interest to us. As you know, they are one of the more consistent advertiser in REMark, and I have been impressed with their reasonable prices on disk drives. By the time you read this, I should have their new hard disk drives for the H-89 and H-100 series, and I will be reporting on them in a future column. Equally impressive are the new H-89 TWOET systems (48 and 96 TPI half height drives), and I will also be looking at the installation and performance for the H-89's.

I received my update card for CP/M 2.2.04 from Zenith for the old H-89. My sources tell me that a number of bugs have been fixed, but I'm not sure what they were. In addition, I understand that the documentation has also been improved, and it now resembles the CP/M-85 and Z-DOS format. Is it worth the $35.00 cost of the update? Probably, but more on that in a future column.

Based on a question about my reviewing games in this column, I have contacted Software Toolworks, and yes, I will be reviewing some games. I have to confess that I'm not a game player as far as my H-100 is concerned since I use that for business purposes. If I want to play some games, I use the Atari 2600 since the sound and graphics is much better. In my opinion, Heath/Zenith systems were not designed for game playing, and as such, they do not do that as well as some other computers. I may be surprised at the quality of games which are available for our systems.

Another exciting project is that I expect to be reviewing some RAM disk hardware and software. I've been talking with one vendor for several months, and with a little luck, I should have a unit by the time you read this. It is supposed to work with CP/M, but I understand that they don't have it set up to work with Z-DOS. I have to admit that I am puzzled by the fact that Zenith has a 256K expansion board available for the H-100, but as far as I know, there is no CP/M-85 or Z-DOS RAM disk software to support it. That seems like a natural link to me. Maybe one of our HUG software experts will develop it.

Reviews of spelling proofreaders and such have also been requested. In particular, The Word Plus and Punctuation & Style by Oasis Systems. I'm working on that.

Although it hasn't come up yet, I don't ever plan to review any of the so-called "copy protected" software in this column. It is a waste of my time, and I just don't believe in it. At this point, I really am not aware of any which would be of interest to you, but I think that any software which does that is not worth my time. From my perspective, it is absolutely ridiculous to make software that can not be copied. Sure, there is a problem with piracy of software, but that's not the way to eliminate it. Affordable and reasonable software prices are the best ways to cope with the problem. Vendors who have tried to protect their software have found that they have only spawned a very good business for those who develop programs to "break" the copy-protect feature. And besides, floppy disks are not perfect, and the copy-protected distribution disk is absolutely guaranteed to fail at the worst possible time.

Mainframe software vendors have also tried to cope with that problem too. They have even tried to place the serial number of the CPU in their programs with little success. Why? Companies do not want to fool around with that sort of thing when they upgrade their equipment to a new/larger/faster processor. It also can create backup problems. But, enough of that.

In The Mail

I've received several letters on previous columns, and I appreciate the time that you spend in letting me know your interests. If you have any specific questions about one of my columns, please type them on a separate piece of paper, and enclose a stamped, self-addressed envelope. That will help me give you a speedy answer, and I'll promise to answer all mail that follows this process. If the questions seem to be of general interest, I'll also include them in the column.

One of the more interesting comments about this column was that it wasn't long enough. So let me know what you would like to see! As far as I know, there are no REMark restrictions on the length of the column...it is strictly based on the amount of time that I have for researching and writing about the subjects. I do expect that the columns will get longer as time goes on, but that also depends on the kind of information that you want to see plus the cooperation that I get from various hardware and software vendors who support the Heath/Zenith line. When I get a lot of things to write about, the column will naturally expand as I attempt to keep up with everything.

Although I'll generally resist the temptation to talk about programming, I am looking at some compilers and other programming tools. The Software Toolworks C compiler is an amazing piece of work, and I'm using it to learn the C language. For $49.95, it's still one of the best software investments that I have ever made.

Ed Percy of Micro-Systems Software tells me that they will be coming out with a new word processor, MSCRIPT, which will sell for less than $100.00 and should be available by the end of March. If it's as user friendly as MTERM, I think that they'll have another winner. More on that in a future column.

Hints and Kinks

Useful time and money saving hints seem to be of general interest, so I'll be including them from time to time. If you have any good ideas, shortcuts, etc., let me know. I will include your name in the column in order to acknowledge credit for the idea.

Speaking of ideas, how about a way to renew nylon ribbons for your H-25 (or any other printer) for a few cents? Read about how to do it next month.

Late Flash

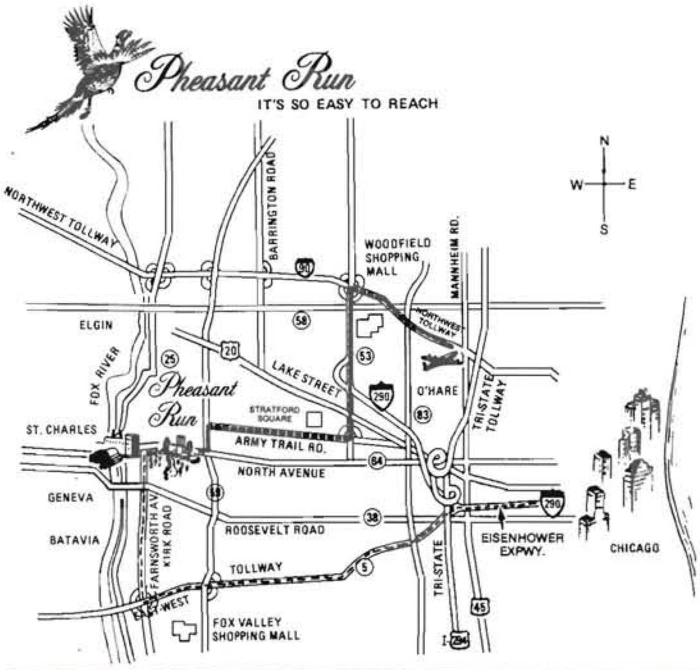
Just as I was wrapping up this column, I got my copy of CP/M-86 for the H-100. And it looks like ZDS really outdid themselves on this. The most spectacular feature is that it will run most 8-bit software, and it seems to do it a lot faster. Even faster than Z- DOS! There are, of course, some modest restrictions on running 8- bit software. My preliminary testing indicates that WordStar, Magic Wand, SuperCalc, and dBase II work just fine. More about the new CP/M-86 next month.

Next Month

When you have a lot of files on a disk, it gets to be very difficult to keep track of which file contains what, and when it was last updated. And wouldn't it be nice if a software package could be found which not only provides a file description of your choice, but includes the last date that the file was changed? And provides some CP/M functions like rename, file copy and backup, and erase? Autodex provides all of these capabilities, and is one of the most useful software "shells" that I have ever seen for CP/M. I use it all the time to keep track of files for my books and magazine articles, and we'll take a look at Autodex next month.

Conference Information

Margaret Bacon
HUG Secretary



As promised last month, this article contains information about Pheasant Run Convention Resort Hotel in Saint Charles, Illinois where we will be holding the International HUG Conference the weekend of July 27, 28, and 29, 1984.

Pheasant Run offers you a variety of accommodations. The Main Lodge and the Tower have single and double occupancy rooms available. There are also bi-level rooms with occupancy for four. When we were looking over this new and exciting location for the International HUG Conference last fall, Pheasant Run was just completing remodeling of rooms in the Main Lodge. The Hotel requires Friday and Saturday night reservations. This is a weekend package. Room Rates are as follows:

	Single	**Double**
Standard	$52.00	$60.00
Deluxe	$65.00	$75.00
Tower	$75.00	$85.00
Bi-level Family Quad	$90.00 (occupancy set at 4)	

The Hotel requires a one night deposit with your reservation. You may bill your deposit to your American Express Card (72 hour cancellation notice required).

As a Convention Resort Hotel, Pheasant Run is geared to helping you occupy your time when you are not busy with the events of the International HUG Conference. The 'Midwest's Largest Health Club' is free to all registered guests. There are both indoor and outdoor swimming pools, an 18 hole golf course, and a large under-roof tennis facility. There are self-improvement programs available during the weekend. You can leave the kids in the Game Room for ping-pong or video games. They have a Special Activities Desk where you can check into other facilities and events either at Pheasant Run or within a few minutes of the Resort.

The Dinner Theater is open for week night performances at 6:30, two performances on Saturday evening and a Sunday Matinee luncheon as well as a dinner performance. The El Poco Toro Lounge on Bourbon Street and the Baker's Wife Lounge both offer live entertainment in the evenings. The Tower Lobby and Lounge along with the Smuggler's Cove are quiet corners available for your relaxation after the planned activities of the Conference.

Pheasant Run is located on Illinois Route 64, 50 minutes from O'Hare International Airport (see the map). Pheasant Run has many years of experience in getting their guests from O'Hare to their facility. When you receive your Conference Ticket, you will be receiving a card to be mailed to Pheasant Run. This card will have space for flight information, as well as hotel accommodations. If you are flying - PLEASE - fill out this card and mail it to Pheasant Run. If we cooperate with them, our arrival will be smooth. In case of an emergency, please contact the Pheasant Run Transportation Department on your arrival to Chicago.

For those of you who will be flying to the Conference, Pheasant Run staff members will be at O'Hare International Airport at United Airlines or American Airlines baggage claim areas by the curb for the purpose of directing you to buses or vans during the times you have told them on the card you mailed to them. Depending on the number of you arriving in the same time period, a charter bus, van, or car will be waiting to take you to Pheasant Run. The most economical transportation is provided by the charter buses or vans ($10.00). However, Pheasant Run's Transportation Director has indicated that the maximum fee to the Resort is $24.00 for the Limousine (single occupancy) from the airport. (Those of you in a real hurry can arrange for a helicopter.) If you make a change in your schedule, they have asked that you inform the Transportation Department at Pheasant Run.

Checklist for Z-100 Computer Users:

✓ 1. Are you getting the best prices?

If you put two computer dealers in the same room, and they BOTH tell you that "they won't be undersold", then one thing you can be sure of is that at least one of them is full of it! We won't make such a silly claim, but it is true that our prices are consistently amongst the lowest — and very often ARE the lowest around!

✓ 2. Does your dealer solve problems, or create them?

When you are just learning a computer system, one thing you DON'T need is a dealer that you have to TRAIN! On our staff we have no one with a background in computer sales, but we DO have people with backgrounds in computer science and engineering! In place of hard-sell, you'll get solutions. If it doesn't sound like much, ask yourself if you'd trust your accountant to recommend a good compatible printer to you!

✓ 3. Do you receive merchandise on time?

Wizardry normally can (and WILL) ship stock items within 24 hours, and most items ARE in stock. In most cases we ship standard UPS, but we also can ship UPS Blue or even Federal Express when you're really in a pinch, for delivery within a mere 2 days to your door!

✓ 4. Is your supplier a proven performer?

We're proud of our past record, and our reputation. Many of the same people now developing products for the Z-100 got their start in a computer from us. These guys know where the best single-source for pricing, support, and service is — and they keep coming back for the same reason. Our support is legendary, and we've literally won awards for end-user support to our customers.

✓ 5. What are your dealer's references?

If we listed just the major firms and institutions that purchase their products from us, we'd fill this page with just their names. Scratch a knowledgeable computer consumer, and you're apt to find a Software Wizardry customer there.

✓ 6. Isn't $1,000 worth a few minutes of your time?

That's how much we can save you on an average system sale! All it takes is a letter or phone call to receive our FREE catalog. In it you'll find over 450 compatible Z-100 products, probably many you didn't even know existed! We LOVE the Z-100 computer, and think the Zenith users are the best in the world! That's why we know our future is secure, the best users sooner or later find their way to the best dealers. We wouldn't have it ANY other way!

Software Wizardry
THE MAGIC TOUCH

ZENITH | data systems
AUTHORIZED SALES & SERVICE

MasterCard VISA

122 Yankee Drive
St. Charles, MO 63301
(314) 946-1968

HUG NEW PRODUCTS

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

885-3009-37 ZDOS ZBASIC Dungeons & Dragons (DND) $20.00

Introduction: DND is the HUG version of the popular game "Dungeons and Dragons", played in real-time. This version displays graphic representation on the screen of the rooms, halls, and doors in the area which the player is in. The object of DND is to find the lord master of the Dungeon.

Requirements: This game requires the ZDOS operating system on an H/Z-100 computer. DND requires the ZBASIC interpreter. Only one drive is required.

These programs have been compiled with the ZBASIC compiler. To use the compiled version, the timing loops in the program will need to be increased.

The following programs and files are included on the HUG P/N 885-3009-37 ZDOS DND disk:

```
README   .DOC
MENUBW   .BAS
MENUC    .BAS
STARTBW  .BAS
STARTC   .BAS
DNDBW    .BAS
DNDC     .BAS
DND      .DAT
DND      .DOC
INSTALL  .BAT
```

Note: The BW files indicate the programs are written for Black and White monitors verses the C files which are for Color monitors. Details are contained in the README.DOC file.

Author: Robert E. Wild
Modified for ZDOS ZBASIC by Richard Evers

Program Content: The search for the lord master of the Dungeon is made by exploring the 50 level dungeon, with the search beginning on level 1. DND is played in real-time, which means the program waits only a short time for a response and will continue playing, without a response.

At the starting point there is a roadside tavern with many other taverns located on level one. It is at these taverns that treasures are cashed in for experience points. An accumulation of experience points will allow the player to become a higher level character, which will increase the chance of survival as the user moves deeper into the dungeon.

During the quest for the lord master, many obstacles will be encountered. The obstacles will include monsters which may try to steal any treasures which the player may have. Sometimes the monster may attack first. The player may fight, cast a spell, or evade. The player must watch his Hit Points and Spell Units to determine which option is best. These units can be refreshed by returning to a tavern or worshiping at an alter.

The user will find objects along the way which may be of help with the search. As easily as the objects appear, they may disappear.

The lord of the dungeon will be found in a HEATHKIT VAULT. As the game progresses, the combination to the safe will be given. The lord master may not be in that vault as there are many vaults throughout the dungeon.

The DND.DOC file contains information on some of the areas of the game. It is recommended that a hardcopy of the documentation be made for future reference. There are useful tables that will be nice to reference while playing the game.

There are many aspects of the game that must be learned while exploring the dungeon in DND.

Comments: DND is an excellent adventure game with the added feature of the graphic display with real-time mode. A "Dungeons and Dragons" master player informed HUG that this version is the best computer implementation of the game he has seen.

TABLE C Rating: (1),(2),(5),(7),(10)

885-4700 HUG Bulletin Board Handbook $5.00

Introduction: This handbook is intended to give the new user a helping hand in using the HUG Bulletin Board (BB) or Special Interest Group (SIG). The HUG SIG is a service provided by CompuServe Inc. of Columbus, Ohio.

Requirements: The handbook may be of use to anyone who has an account (User ID) with CompuServe and has been added to the HUG SIG as a member. It also may be of use to non-users of CompuServe, who would like to pick up some insight into the use of the HUG SIG.

Content: The content of the handbook is a mixture of some information files of the SIG, sprinkled with some new advice and suggestions for learning to use the SIG quickly and efficiently.

Note: CompuServe is continually making changes to the services they provide. These changes may directly or indirectly affect or alter the operation of the SIG. Therefore, SIG options stated in the handbook may become obsolete or altered without notice to HUG and/or the user. The user is responsible for noting changes to the CompuServe service. SIG changes are normally posted within the SIG option "NEW".

Comments: This handbook may be of practical help to users not familiar with the commands and options of the CompuServe SIG service.

TABLE C Rating: (9)

885-8027 HDOS
885-8028-37 ZDOS
SCICALC Scientific Calculator $20.00

Introduction: SCICALC is a program designed to turn the H/Z-89 or H/Z-100 computer into an extremely powerful and easy to use 14 digit scientific calculator. Besides providing six arithmetic operators, SCICALC supports a large variety of transcendental functions, trigonometric functions, statistics functions, U.S. to metric and metric to U.S. conversion formulae, and key mathematical and physical constants. It has also been pre- programmed to solve problems related to geometry (both plane and solid) and statistics (including linear regression and percentile calculations), and can perform arithmetic in base 8 and base 16.

Requirements: SCICALC will run under HDOS and ZDOS.

HDOS: This version of SCICALC requires the HDOS operating system version 2.0 on an H/Z-89 or H8/H17/H19 with 56K of memory. Only one hard-sectored disk drive is required. No printer is required.

The Microsoft BASIC (MBASIC) interpreter version 4.82 is required.

Note: To H8 users, the H19 terminal is required.

ZDOS: This version of SCICALC requires the ZDOS operating system. Only one drive is required. A printer is not required.

This version requires the ZBASIC interpreter release 1, version 1.25.

The following files are contained on the P/N 885-8027 HDOS and P/N 885-8028-37 ZDOS SCICALC disks:

HDOS	**ZDOS**
SCICALC.BAS	**SCICALC.BAS**

Hardcopy documentation is included with both versions.

Author: Brian Downs

Program Content: SCICALC utilizes a four-level stack for arithmetic operations, along with nine registers for saving intermediate results, and a "Last X" storage location for remembering the argument to single parameter functions. Complex arithmetic expressions can be easily and quickly entered without the use of parentheses since SCICALC employs the "Reverse Polish" expression entry technique (also employed by Hewlett- Packard calculators).

All registers, stack and Last X storage locations are continously updated and displayed by SCICALC and in the format designated by the user. The informative display especially makes this powerful calculator easy to learn and use. The current "state" of the calculator (i.e., the values of all stack and register locations) can be saved for use at a later time.

The following is a partial list of the common functions of SCICALC's MAIN menu:

X to the Y power
1/X (inverse)
ln(X)
log(X)
e to the X power
Square Root of X
Cube Root of X
X squared
sin(X)
cos(X)
tan(X)
cot(X)
sec(X)
csc(X)
Degree/Radian conversion
Arcsin(X)
Arccos(X)
Arctan(X)
X! (X factorial)
Change percent
%+
%-
logY(X)
Base 10
Base 8
Base 16
Change sign
sum X
Int(X)
Frac(X)
Round(X)

The following is a list of the conversion functions:

in to cm	cm to in
ft to m	m to ft
mi to km	km to mi
lb to kg	kg to lb
oz to g	g to oz
qt to l	l to qt
deg to rad	rad to deg
floz to ml	ml to floz
lb/in to atm	atm to lb/in
atm to mmHg	mmHg to atm
BTU to cal	cal to BTU
J to cal	cal to J
Fahr to Celc	Celc to Fahr
Kelv to Celc	Celc to Kelv
km to light yr	light yr to km
ft↑2 to acres	acres to ft↑2
acres to hectares	hectares to acres
hp to watts	watts to hp

The following is a list of the constants from the CONSTANTS Menu: Menu:

Pi
e
Velocity of light
Velocity of sound
Acceleration due to gravity
Radius of the earth
Circumference of the earth
Distance to the moon
Distance to the sun
Density of water
Heat of fusion of water
Heat of vaporization of water
Volume of one mole of gas
Gas constant
Mechanical equivalent of heat
Charge of one electron
Charge/mass ratio
Atomic mass unit
Mass of electron
Mass of neutron
Mass of proton

Ordering Information

For Visa and MasterCard phone orders; telephone Heath Company Parts Department at (616) 982-3571. Have the part number(s), description, and quantity ready for quick processing. By mail; send order, plus 10% postage and handling, up to a maximum of $3.50 to Heath Company Parts Department, Hilltop Road, St. Joseph, MI 49085. Visa and MasterCard require minimum $10.00 order.

Any questions or problems regarding HUG software or REMark magazine should be directed to HUG at (616) 982-3463. REMEMBER - Heath Company Parts Department is NOT capable of answering questions regarding software or REMark.

Avagadro's number
Bohr radius
Boltzman's constant
Euler's constant
Faraday's constant
Planck's constant
Rydberg's constant
Golden Ratio
Angstrom
Micron
Multiplication factor for milli-
Multiplication factor for micro-
Multiplication factor for nano-
Multiplication factor for pico-

The following is a list of calculations of some of the advanced functions of the MATH Menu:

Circumference of a circle with a given radius.
Area of a circle with a given radius.
Surface area of a sphere with a given radius.
Volume of a sphere with a given radius.
Area of a cylinder of a given radius and height.
Volume of a cylinder of a given radius and height.
Area of a cone of given height and radius of base.
Volume of a cone of a given height and radius of base.
Approximate the circumference of an ellipse.
Area of an ellipse.
Length of the third side of a triangle.
Length of the second side of a right triangle.
Area of a triangle given all three sides.
Area of a triangle given the base and height.
Distance between two Cartesian coordinates.
Equation of a line given two points on the line.
Equation of a line given one point and the slope of the line.
Intersection of two lines given the line equations.
Prime of X.
Hyperbolic sine, cosine, and tangent.

Convert split octal to base 10 and base 10 to split octal.
Convert Cartesian coordinates to polar coordinates.
Convert polar coordinates to Cartesian coordinates.

Calculate the accumulation of a money market funds, given number of months, yearly interest, initial principal, and monthly deposit.

The following are statistical functions of SCICALC:

Calculate the probability that a normally distributed random variable of a given mean and standard deviation will equal X.

Calculate the probability that a Poisson distributed random variable with a given mean will equal X.

Calculate the probability of observing x occurrences out of n trials of some event that has probability p of occurring.

Calculate the number of combinations of m things can be chosen out of n (different) things (without replacement).

Calculate the number of permutations of m things can be chosen out of n (different) things.

SCICALC performs all internal arithmetic using double precision variables. This affords approximately 16 digits of accuracy. However, math functions provided by most BASIC interpreters, including MBASIC 4.82 and ZBASIC 1.0, return only single precision values. These functions are what are used directly to provide many of the MAIN menu functions, and indirectly, to provide many of the ADV MATH menu functions.

SCICALC provides for error checking if the user tries either an illegal programming option or mathematical function.

Comments: This is an excellent package, well done! The documentation leads the user through simple arithmetic functions to familiarize the reader with how SCICALC works. Then the user is introduced to the other functions and options.

TABLE C Rating: (0),(1),(3),(10)

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.

Volume 3, 1983 REMark now available, part #885-4004 and priced at $20.00.

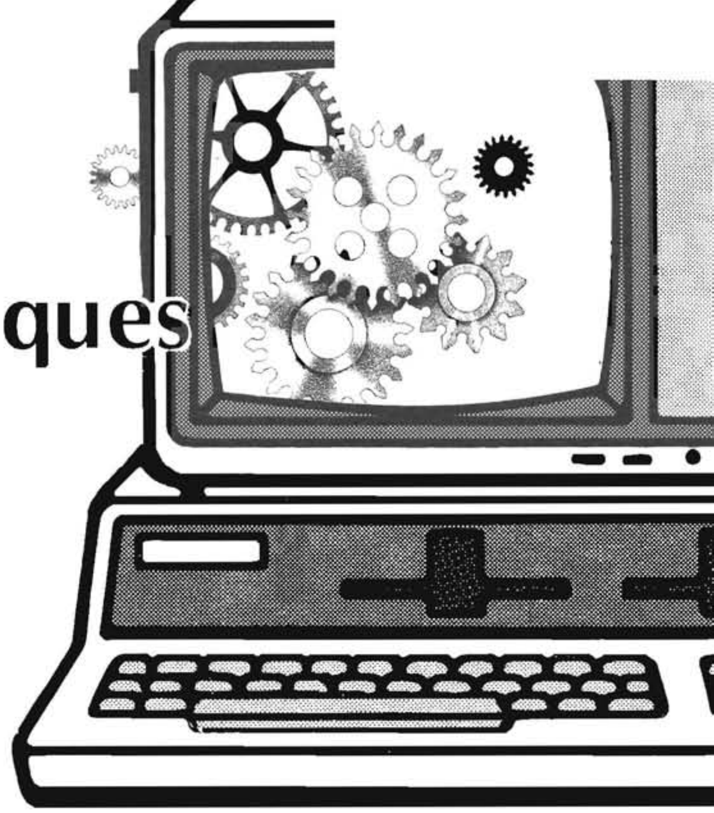
Part Number	Decription of Product	Selling Price	Volume - Issue
HDOS			
885-1030[-37]	Disk III, Games II	$18.00	5-2
885-1096[-37]	MBASIC Action Games	$20.00	5-2
885-8026	Space Drop	$16.00	5-2
CP/M			
885-1234[-37]	CP/M Ham Help	$16.00	5-2
885-8025-37	CP/M 85/86 FAST EDDY	$20.00	5-2
MISCELLANEOUS			
885-0004	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	
885-4004	REMark Volume 4, Issues 36-47	$20.00	

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

Thinking about your next vacation?

Plan now to attend the Third International HUG Conference!

ZBASIC Simulation Techniques



Arthur C. Smith
11685 Shangrila Avenue
Hesperia, CA 92345

Computer simulation of events and processes have become one of the most powerful engineering and research tools ever to fall into the hands of Man. From the watchmaker's bench to the edges of our solar system, this versatile art has found useful applications. Whatever the task, computers have proven their worth with printout and picture. Now this splendid potential to see what is not, to manipulate what may never be, and to develop what no man could ever afford has fallen within my reach and yours.

All of us who have found it necessary to make models or mock-ups as a step toward something greater, can appreciate their value. Many problems may be discovered and retired before the main project is ever begun. This is a time-honored approach to getting things done right the first time. Still, the cost and time spent on a reasonable model can be quite significant. That is when a computer simulation should be considered.

Anyone can apply simulation techniques. All that is needed is a good computer, an adequate language, and a few scraps of scratch paper. ZBASIC was used on Heath's H-100 for the two programs presented here. If any kind reader is concerned that an interpreted BASIC will be too slow for good dynamic graphics, he may put his concerns aside. Here is why. Fast graphics are very important for action games and we can agree that high-speed simulations may be useful tools. But when we need a system that is both dynamic and interactive, it is best to put speed aside. We want to be able to inspect the progress of events in easily understood steps. And when it is time to stop the action, we want to be able to stop at a specific point and we should not need the finely tuned reflexes of a video arcade superhero to do it.

In our system, everything will begin and end this side of infinity and everything will repeat upon command. The processes employed in the simulation can be interrogated at will and the information can be saved for reference and analysis. BASIC is fast enough. It is, in fact, an excellent choice for most users.

Before we get busy with specifics, let's take a look at what kinds of things might offer applications for simulation programs. I'm sure you will find one that interests you.

1) **Hobbies.** Boat and plane modeling and R/C. Amateur radio antenna structures. Astronomical optics and drive systems.

2) **Practical.** Design of gear, belt, chain, and hydraulically driven systems. Analysis of forces and limits in fluid and mechanical systems. Marine and aeronautical navigation and surveying. Ballistics and artillery.

3) **Education.** Anything that moves or is moveable can be a topic for simulation. Simple programs can offer new insights to inquiring young minds.

Both of the programs presented here are simple in concept and in execution. Nevertheless, they demonstrate the fundamentals that make a simulation work. The main program is a wheel driven by a piston and the assumed input is compressed air. The second program will present a simple fluid problem.

Figure 1 is a sketch of the proposed engine. Air is to be valved to the cylinder but the valve system is not important so we will use arrows to indicate which side of the piston is getting the push. It is best to keep the screen free of most non-essentials. This idea helps to speed up getting the job done. And the results are easier to read. We must couple the linear motion of the piston to the rotary motion of the flywheel. A connecting rod makes this link. The burning question for the programmer is centered on how to make the parts move together.

Before we start, a few observations might be useful. 1) The connecting rod attaches at the rim of the wheel and must be at least as long as the diameter of the wheel. When you run the program you can see what happens if the rod is shorter - that's what the program is for. 2) The piston's travel equals the diameter of the wheel. 3) The working width of our screen must be more than three times the diameter of the wheel. Saying all of this helps us get a sense of scale. My screen has a horizontal resolution of 640 pixels. I have treated one pixel as one mathematical unit in my program. This arbitrary choice is convenient. If true dimensions were needed, I would let the program make the conversions. Two other things should be considered before the writing begins. A method for introducing new parameters should be woven into the program. Also, because we will be drawing pictures with objects of varying sizes, there will be times when the picture seems a bit off-center unless we have a method for automatically selecting the correct starting point for each set of parameters.

This simulation under ZBASIC is made very simple by the fact that each line or figure drawn may begin where the last one left off. This can be made to happen by default. It's a bit like a map that says,

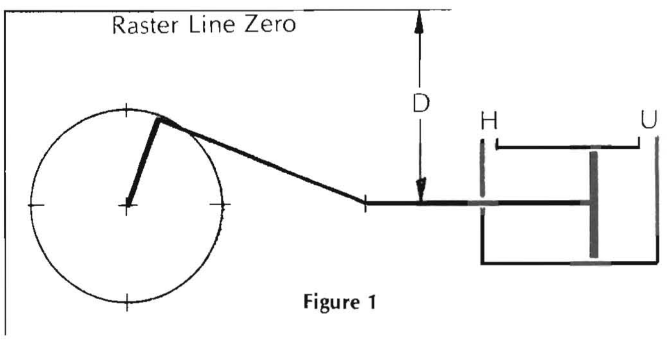


Figure 1

"Walk 30 paces north from point A to point B, thence 50 paces west to point C". When there is a string of elements that must move together, it is necessary to specify the beginning coordinates for only the first point and even the undignified thrashings of the connecting rod can be managed perfectly in this way.

There were some good reasons for selecting the center of the wheel as the reference point for the screen images. If they are not obvious, they will be soon. After the wheel position is chosen, the correct place for the cylinder can be calculated. These two objects will be stationary on the screen. The first moving part to be drawn will be the spoke. For clarity, just one spoke is used. We will glue one end of the connecting rod to it. The other end of the connecting rod must somehow be made to fall on the center-line common to the wheel and the cylinder. When that is done, the piston assembly can be drawn and that will complete the first image.

You have looked at the engine program by now and have seen that much of what goes on is obscured by the substitution of variables for equations in some of the steps. This was done to make it simpler for you to use the program if you wish. It really isn't important that anyone understand just what it does. However, I have already decided to explain some of it and nothing is going to stop me now.

When I decided to do this article, my first problem was to find a good "something" to simulate. A steam engine came to mind very quickly because it fits so well. We all understand the idea, etc. So, off I ran to the computer and started drawing pictures...a circle, a box. Hot dog! We're really rippin'. Then I came to that spoke. Just one little line to make on the wheel but I could not merely command it into being. I had to compute it into being. If you have not been through this before, please take notes. To draw an angled line from some known point to an unknown point you must first calculate that point's position with some elementary trig. There are two tiny problems, though, and you must take them into account. First problem: you and I are accustomed to angles given in degrees but BASIC only understands radians. No Problem. Just don't forget it. Second problem: trig solutions just naturally assume x and y axes will be scaled alike. Your screen was not planned with trig in mind.

The trig part is easy. Add the procedures to your console notebook for future reference. To convert degrees to radians, just multiply degrees by 2 x pi and divide by 360. Or RADIANS = DEGREES x 6.28318 / 360 and DEGREES = RADIANS x 360 / 6.28318. After an angle is converted into radians, BASIC can use it for trig problems. The trig problem we have in mind is that spoke. Figure 2 illustrates the problem. The spoke is the line labelled M. The lines labelled M, S, and T form the triangle which must be solved. When the computer understands where on the screen lines M and T intersect, it can draw line M. For now all we can tell it is that the spoke goes to a point E + S on the x-axis and D - T on the y-axis. We can convert this to numbers by solving the triangle. S = M x sin J. You can see where this is done in the program on line 390 but it doesn't look right. That is because we don't use the angle J directly. Instead of J we have used the sine of J and it is stored in a table under the name B(F). T is solved on the same line using the cosine of J from another table. The equation for T is T = M x cos J.

```
10 '(engine.pic)  Version 10-28-83  ZBASIC
    by Arthur C. Smith
20 CLS:KEY 12,"GOTO 120"+CHR$(13):COLOR 2:
   KEY OFF:DIM A#(7):DIM B#(7)
30 PSET (4,8):DRAW "c4h4r2u4r4d4r2g4":
   GET (0,0)-(8,8),A#        'Red arrow
40 PSET (4,8):DRAW "c1h4r2u4r4d4r2g4":
   GET (0,0)-(8,8),B#        'Blue arrow
50 C=.44                    'Screen X-Y correction factor
60 D=105:K=78:P=83:Q=127 'Some y-axis reference points
70 DIM A(360):DIM B(360):G=6.28318/360
80 CLS:LOCATE 5,30:PRINT"STANDBY FOR 20 SECONDS"
90 FOR F=1 TO 360           'Make a sin and cos table
100 A(F)=COS(F*G):B(F)=SIN(F*G)   'Radians, please!
110 NEXT F
120 CLS:LINE (0,0)-(640,225),1,B
130 PRINT"HOW MANY DEGREES PER STEP <"J">"
140 INPUT;J
150 PRINT:PRINT"WHAT IS THE RADIUS OF THE WHEEL <"M">"
160 INPUT"(limit 100)";M
170 PRINT"HOW LONG IS THE LINK? <"N">"
180 PRINT"(range is"2*M "to"614-4*M")"
190 INPUT;N
200 PRINT:PRINT"HOW MUCH DELAY <"B">"
210 INPUT;B
220 Q$="c0u2d4u2r15r"+STR$(2*M)+"u21d42"
    'Piston erase pattern
230 P$="c4u2d4u2r15r"+STR$(2*M)+"u21d42bd3c4d1"
    'Piston pattern
250 E=(614-4*M-N)/2+M:H=E+M+N+5:U=H+2*M+20:F=0
    'E centers image
260 CLS:LINE (0,0)-(640,225),1,B
270 PRINT"Radius="M:PRINT"Link ="N:PRINT"Deg/step="J
290 LOCATE 25,20:
    PRINT"CTRL-C THEN F12 TO SELECT NEW PARAMETERS":
    LOCATE 20,1
300 CIRCLE (E,D),M,2:CIRCLE (E,D),2,2
    'Draw the wheel and hub
310 LINE (H,K)-(H,Q),2:LINE -(U,Q),2          ' /
320 LINE -(U,K),2:LINE (U-10,K)-(U-10,P),2
    '( Draw the cylinder
330 LINE -(H+10,P),2:LINE -(H+10,K),2         ' \
340 F=F+J:IF F>360 THEN F=F-360
    'Simulation begins here
350 IF A(F)<0 THEN 380
360 PUT (H+1,67),A#,PSET:PUT (U-9,67),B#,PSET
    'Air into left side
370 GOTO 390
380 PUT (H+1,67),B#,PSET:PUT (U-9,67),A#,PSET
    'Air into right side
390 S=(M*B(F)):T=(M*A(F))
    'Dimensions S and T from figure 2
400 V=SQR(N↑2-T↑2)+E+S
410 LINE (E,D)-(S1+E,T1*C+D),0      ' /Erase all
420 LINE -(V1,D),0                  '( of the
430 DRAW Q$                         ' \moving parts
440 CIRCLE (E,D),M,2:CIRCLE (E,D),2,2
    'Rim gets damaged as link passes over it
     so redraw it each time through the loop
450 S1=S:T1=T:V1=V
460 LINE (E,D)-(S+E,T*C+D),4
    'Spoke from hub to rim - just one for clarity
470 LINE -(V,D),4     'Link from rim to piston assy
480 DRAW P$           'Piston assy
490 FOR BB=0 TO B:NEXT BB:GOTO 340
```

Listing 1

With the radians and triangles safely behind us, we can tackle the question about the screen ratios. Under ZBASIC, a radius or other line is specified in terms that are just fine on the x-axis, but a line of

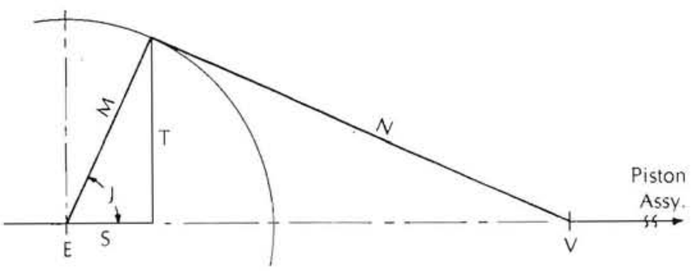


Figure 2

300 pixels that does not reach center screen on the x- axis is longer than the entire vertical screen. Something will have to be done. Every vertical dimension that is calculated from a horizontal dimension must be multiplied by about 0.438 before being drawn or the proportions of things will be interesting but wrong. A word of caution - be sure to apply this correction after all calculating is done; the scale factor must not be included in calculations that pass values to other formulas. In our program, C is the screen factor and we have used 0.44. That is close enough.

Each time the program steps the wheel (Yes! It is easier to solve this system if the wheel drives the piston, I'm glad you noticed.), the spoke triangle is solved and a line is painted from the center of the wheel to some point on its rim. ZBASIC will remember that point while we solve the next triangle.

This is the last triangle we need to solve. It is formed by lines T and N plus a line from V to the bottom of T. We know the lengths of T and N already and must find the length of the remaining side. There are two good ways to do this. The easiest way does everything we need so it is the logical choice. The missing side = the square root of N squared - T squared. You recall that our objective was to determine where the connecting rod should meet the center-line. The missing side + S + E is the answer. See line 400 of the program. Now a line can be drawn from the spoke to V and it will have the correct length. Finally, the piston assembly may be drawn at the end of the connecting rod. No calculations are required for that.

One picture has been drawn and the next one must show the moving parts in new positions. It will be confusing if the old positions are not erased. I choose to erase the old at the last possible moment prior to the painting of the new. This has the advantage of presenting each image for the maximum possible time during each cycle but it does mean that we get another small problem to resolve. Images are erased by drawing them again using all the same coordinates but changing their colors to whatever the background might be. The hangup is that we want to leave the picture on the screen while we compute the points for the next screen. There goes all our data for erasing the old one! So line 450 saves the three crucial data under new names and we will do our erasing using these aliases.

The sequence during simulation is Calculate, Alias, Erase, and Paint. It does not matter on the first pass through that you will be erasing a non-existent image. And that fairly sums it up. This is just a skeleton program, though. A useful program would continue after the Paint operation with an additional calculation interval. At that part of the program comes the payoff. Sometimes it is sufficient to see the picture but most often one wants the numbers that come with it.

Now I will try to explain some of the obscure parts of the program and justify my decisions. 20) These are H-100 set-ups up to the DIM statement. There is no exit routine to restore the screen or key assignments. The DIM statements are for the arrows in the next two lines. They will be stored in arrays defined by these statements. 80-110) Here we create a pair of tables. It takes about 18 seconds to create these tables but only about three seconds to read them out to variables. They make the show roll along more briskly. 130-210) These lines prompt the user for necessary variables. But they do more than that. Some of them calculate and display the range of useful or legal values. All of them tell the user what values he chose for the last run. 220) Notice the STR$(2*M) in the middle of Q$. Q$ will be used by the DRAW command. Perhaps you were unaware that this could be done. 250) This line defines several starting points for subsequent LINE and DRAW commands. 270-290) User information is present during simulation. 380) PUT takes the arrow out of storage in the A# and paints it on the screen.

```
10 '(Flow.pic)  version 10-28-83  ZBASIC
   by Arthur C. Smith
20 'Left  side volume=75000   (150 X 500 X k)
30 'Right side volume=7500    (150 X 50 X k)
40 CLS:LINE (555,40)-(560,50),4,BF:
   LINE (610,40)-(615,50),4,BF:
   LINE (534,40)-(530,50),4,BF:
   LINE (25,50)-(30,40),4,BF
50 DRAW "C4D160R580U160BL50D155R50L60U40BL10D40L10U155"
60 LINE (31,50)-(529,50),4    'Left side fluid level
70 LINE (541,164)-(549,199),4,BF     'Valve closed
80 PAINT (40,195),3,4                'Fill 'er up!
90 LINE (31,50)-(529,50),0   'Erase top PAINT boundary
100 PRINT"            RATIO OF VOLUMES IS 10:
    1            PRESS F5 TO OPEN VALVE"
110 END
120 LINE (541,195)-(549,199),7,B    'Valve open
130 PAINT (40,195),3,4:R=200:L=50
    'Flood area just right of the valve
140 R=R-1:LINE (561,R)-(609,R),3
    'Water level is rising
150 S=S+1:IF S<10 THEN 170
160 S=0:L=L+1:LINE (31,L)-(529,L),0
170 IF R<=L+1 THEN END
180 GOTO 140
```

Listing 2

A program such as this is fun to write and amusing to run, but as a tool it is quite weak unless it is enhanced a bit. The most elementary additions might be screen displays of the calculated angle where the connecting rod and piston assembly meet or, perhaps, of piston travel increments. But the whole thing really begins to bloom when we make assumptions about speed and assign each of the elements some mass.

The second short program might amuse you if you care to try it. A full reservoir connects to a smaller, empty one. When the valve between them is opened, the water fills the small reservoir and the new fluid level is appropriate for the ratio of the volumes. In this example the levels are written into the program, not calculated. As the program runs you will see that there are two red lines that serve as boundaries for the PAINT command and they are erased when no longer needed.

Graphics for charts and games is great but there is so much more. If you put a strong calculations program and a simulator together, you have a powerful tool and a superb teaching aid. Where models and mock-ups have been used in the past, a simulator might be used in the future. The potential for savings in time, labor, and materials is certainly great.

I do hope you will be inspired to try some simulations programs of your own. You will find them useful and fun.

Now you can program HERO 1™ from your computer.

It's a breakthrough by the crack robotics team at Virtual Devices! Meet MENOS I: the control system that brings HERO to life.

MENOS I's software makes HERO more fun to operate.

It's as fast and easy as programming your PC. Because MENOS takes full advantage of a computer's keyboard, memory, and disks—to save you months of time and frustration.

MENOS even performs a diagnostic check every time you start operating. And when you write programs in our high-level robotic language, MENOS prompts you and checks your syntax. You can edit quickly, then download programs to HERO.

A split window display of motor positions and sensor readings lets you monitor HERO's execution of your program. Or you can simulate the execution right on your computer screen. So you don't even need a robot to learn robotics!

MENOS I's hardware gives HERO a mind of its own.

Besides a more powerful microprocessor and an RS-232 port, you get 34K of memory, with 3 operating systems in ROM—including an upgrade of HERO's own. Plus 4 expansion slots for more breakthroughs.

Our fee is very reasonable.

Especially now, when the introductory price is just $395. (After March 31, it's $595.) And you get a 30-day money-back guarantee, plus a full year's warranty on parts and labor.

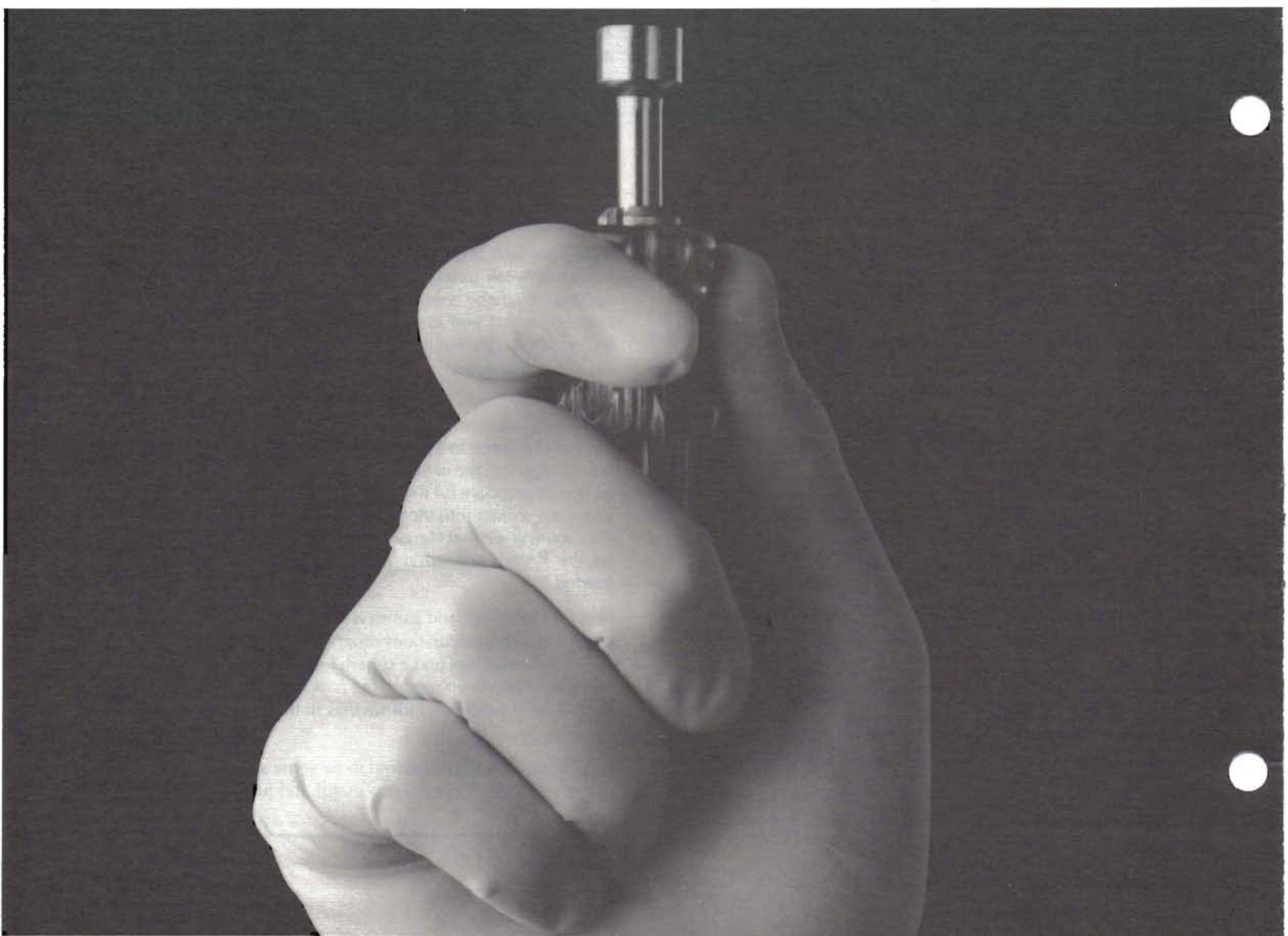
To find out more, call us or write us at P.O. Box 30440, Bethesda, MD 20814. Better still, order today. And get ready to perform a heroic feat of surgery.

Call toll-free: 1-800-762-ROBO.

In Maryland, 301-986-9327.

VIRTUAL DEVICES INC.

WORLD'S FIRST BRAIN TRANSPLANT PERFORMED ON ROBOT.



THE QUALITY SOFTWARE SOLUTIONS YOU NEED

SOLUTION #1: CATALOG-MASTER

If trying to find the disk file you need right now is a little like looking for a needle in a haystack, you need *CATALOG-MASTER.*

You will realize the following benefits:

- you can catalog your disks quickly and easily
- you can catalog all your disks, even different types
- you can find the disk files you need quickly
- you can get more information by using file descriptions

Find that file you need quickly and easily with ***CATALOG-MASTER***

Requires 2-drives and 48K RAM
Specify HDOS or CP/M when ordering.
Regular Price . . . $29.95
Special HUG price . . . only **$24.95!**
(~~First~~ Next 100 orders!)

Get both the HDOS and CP/M versions for only $45.00

LIMITED TIME OFFER — CALL YOUR ORDER IN TODAY

SOLUTION #2: FUND-MA$TER

Now for ZDOS!

If you would like to know not only when to buy or sell stock but also how much to buy or sell, you need ***FUND-MA$TER.***

You will realize the following benefits:

- you can know when to buy or sell your shares
- you get timely help on how much to buy or sell
- you get the current status of your investments
- you get a complete history of your stock transactions
- you do not have to be a stock market expert to benefit

Get the tool that provides effective information for the management of stocks and mutual funds — get ***FUND-MA$TER.***

Requires 48K RAM
Specify HDOS, CP/M or ZDOS when ordering.
Regular Price . . . $39.95
Special HUG price . . . only **$34.95!**
(First 100 orders!)

LIMITED TIME OFFER — CALL YOUR ORDER IN TODAY

SOFTWARE PRODUCTS FOR YOUR H8, H89, Z90, and H/Z100 COMPUTER

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"**!

All products are available in HDOS or CP/M format, except where noted. Add $2 for shipping. Michigan residents please add 4% for sales tax. PAYMENT IN U.S. FUNDS PLEASE.
Call or write for more information and our **FREE** catalog.

For Faster Delivery
Call 906-249-9801

MasterCard VISA

GENERIC SOFTWARE
P.O. Box 790 - Dept. 384R
MARQUETTE, MI 49855
906-249-9801
10 AM - 5 PM EST

FLOPPY DISK CONTROLLER

Controls Any Combination Of Up To Four
8" and 5¼" Drives

This easy to install plug in board can control any combination of single or double sided, single or double density drives.

Designed especially for H88/H89 users.

- Fully compatible Bios supplied for your CP/M 2.2 operating system
- Easy to follow instructions
- Contains controller board with boot prom
- Order cables for connection $15 (HFDC-110)
- ***Introductory Offer $395,*** *Order HFDC-100*

NORTH COAST INTELLIGENCE INC.
1201 Cherokee Trail
Willoughby, Ohio 44094
Phone: 216-946-7756

Check, COD, VISA or MC — 90 Day Warranty

ARE YOU PREPARED FOR TAX TIME?

TAXTYME will assist you in preparing and calculating your 1983 Federal Income Tax forms. **TAXTYME** simplifies filling out the forms by leading you through a series of formatted questionnaires. **TAXTYME** includes formatted questionnaires for IRS Forms 1040, Schedules A, B, and C. Schedule C is processed separately so that several Schedule C business activities can be calculated and combined on Form 1040. Information from Schedules D, E, F, and the many numbered Forms are calculated separately and entered on the questionnaires. Forms G, W, and 2210 are calculated automatically when required. **TAXTYME** makes decisions on Income Averaging, Marital Deduction, and penalty Form 2210 if necessary. **TAXTYME** then processes the information you enter, performs the necessary calculations, and produces a computer printout for direct transfer onto IRS forms. Requires a printer.

ONLY $59.95!

CP/M version for H89, H8, requires CP/M and MBASIC
ZDOS version for Z-100, requires ZDOS and ZBASIC

Newline Software

P.O. Box 402, Littleton, MA 01460 (617) 486-8535

ORDER NOW . . . SHIPMENTS BEGIN JANUARY 15, 1984

NAME ______________________
STREET ______________________
CITY ______________________
STATE ______________ ZIP ________

CHECK ONE
☐ H89, H8, CP/M
☐ Z-100, ZDOS

Send me _____ TAXTYME program(s) at $59.95 each.
Check one: ☐ payment enclosed ☐ send COD (add $3.00)
Send order to:
NEWLINE SOFTWARE, P.O. BOX 402, LITTLETON, MA 01460
Foreign orders: add $3.00 Airmail, $10.00 for non-U.S. checks

CP/M is a trademark of Digital Research, Inc.
MBASIC is a trademark of Microsoft, Inc.
ZDOS, Z-100 are trademarks of Zenith Data Systems, Inc.

The Heath Advanced Microprocessor Course

D. C. Shoemaker
HQ USEUCOM Box 897
APO NY 09128

Heath's reputation is founded on several skills unique in the hobby electronics industry. High quality, reliability and reasonable price means that they have no competition. Equally famous is the high quality of Heath's documentation, both for the kits themselves and for their software. Finally, Heath's training courses are known industry-wide as about the best you can get for home instruction. Heath was one of the first to introduce home computer and solid state electronics courses, and the ET-3400 8-bit trainer has won at least one international award for excellence. Now there's a 16-bit follow-on to their 6800-based trainer, the new Advanced Microprocessor Course. If you own or contemplate owning a 16-bit microcomputer such as the H/Z-100, and you want to get the most from it by programming in 8088 assembly language, then this course is worth a closer look.

Like the ET-3400, the Advanced Microprocessor Course actually has two main parts. The first is the course itself, EE-8088, of which more later. The second, and perhaps more attention-getting part, is the ET-100 computer trainer that is intended to accompany the course. A quick look at the trainer will help place the course in a clearer context.

The ET-100 trainer is based on Intel's 8088 microprocessor unit (MPU), now known as the iAPX88/10, the current "processor of choice" for a great many microcomputer makers. Closely related to the 8086, the 8088 shares the same instruction set and specifications except for buss width (8 bits for the 8088, 16 bits for the 8086) and the size of the instruction fetch buffer. From a practical standpoint, the differences can be reduced to a question of ease of combining 8-bit and 16-bit microprocessors into one system. Use of the 8088 makes the job easier, and the combination of the 8-bit 8085 with the 8088 is relatively simple and inexpensive due to the "family" nature of the power supplies, timing, and other factors. When Heath decided to design a 16-bit system, the natural choice was the 8088 paired with the 8085 to allow maximum use of the wide range of existing 8-bit software while we all eagerly await the flood of 16-bit software that's "just around the corner".

In designing the ET-100 trainer, Heath tried to maintain a high degree of compatibility with the H/Z-100 systems, for two primary reasons. First is the need for the trainer to be a natural lead- in to their main computer, the H/Z-100. This compatibility was not the case with the earlier ET-3400 and the H8 and H/Z-89 computers, which used the 8080 and the Z80, respectively. This was often cited as the main short-coming of an otherwise excellent trainer. Second, given the number of computers using the 8086/8088 family of chips, using any other in a trainer would greatly reduce the usefulness of the training gained.

The ET-100 starts out as a minimum-configuration computer using a television for a display and a tape recorder to store your programs. Later, it can be made to "grow" into a full-blown 16-bit system that's compatible with the H/Z-100 in all respects but one. The expanded trainer can include two disk drives and a high resolution color monitor. The keyboard is similar to the H/Z-100, but is detached (giving rise to speculation that the next H/Z-100's will be so equipped). The main difference is the complete lack of the H/Z-100's 8-bit processor. For the person who primarily wants to learn 16-bit processor techniques, this is not a problem. Some users may find the lack of the 8-bit processor to be a drawback, but as the next two years or so go by and more 16-bit software becomes available, that lack will probably seem less and less important.

The trainer, however, is not the main object of this article. The main theme is the usefulness of the course as a teaching tool for learning to program the 16-bit microprocessors. First, a word of caution. The H/Z-100 was primarily intended to run Microsoft's MS-DOS (or ZDOS in it's Heath/Zenith release). The EE-8088 and the ET-100 trainer use the assembler from Digital Research's CP/M-86, ASM-86, as the assembler you will use to create the programs for the course. A simple line editor, debugger, and this assembler are provided in Read-Only Memory (ROM), making it easy to use since it's immediately available, but it would be difficult to replace it with some other assembler such as Microsoft's Macro-assembler (MASM). CP/M-86 is only now (October 83) becoming available for the H/Z-100, and is considerably more expensive than ZDOS. Most 16-bit computers use a version of MS- DOS, so you can expect to incur some additional work in transferring what you learned using ASM-86. Notwithstanding this drawback, I feel that the course could be used by anyone desiring an introduction to 8086/8088 programming, no matter on what brand of computer. Making the proper allowances, a user of an IBM PC could use the course just as well as a user of an H/Z-100.

This one shortcoming aside, the course as a whole is extremely well thought out. While not expressly intended for the rank beginner, I believe that someone who wanted to apply the necessary extra work could use the course as a basis for a general introduction to assembly language programming. That's important, since good books on 16-bit assembly language programming aren't exactly thick on the ground. Heath does, however, recommend that those persons who have had no experience with digital electronics might profit from taking their "Digital Techniques" courses first. The choice is an individual one and a lot would depend on your ultimate goal.

If you are contemplating using the EE-8088 course with the 16-bit trainer, you'll require the package of electronic components that come with the course. These include a handful of resistors, eight push-button switches with color-coded caps, a seven-segment LED, 19 various integrated circuits, and assorted wire and hardware. These parts allow you to make a simple light pen, and perform a series of hardware interfacing experiments to illustrate programming techniques. Many of the lessons you'll derive from the course depend on some of these experiments, and if you are using the course in conjunction with the H/Z-100 computer rather than the trainer computer, you'll lose a little. My feeling is that these losses aren't serious enough to outweigh the advantages of the instruction. Lacking an ET-100, I just added the parts to my spares collection.

The heart of the course is the instructional material itself, and this comes in the form of about a thousand pages of text, a set of dividers, and two of Heath's new style light tan D-ring binders. You get to assemble the two text books, and it's a tight fit. You also get a programming guide in the form of a plastic coated full-page card, printed on both sides with instructions for using the ROM-based firmware of the ET-100 computer. It also contains a summary of the escape sequences that the computer and display will respond to, and a decimal-hex- ASCII conversion chart that includes the graphics characters that the trainer will produce (the standard H19/H89 graphics.) The card will be of limited usefulness unless you're using the trainer, since the commands apply only to the trainer's editor, debugger, and assembler.

The course itself consists of ten sections and five appendices. The first section is Computer Basics, a general introduction to the 8088 MPU. It covers the basic terminology of microprocessors, the main components of the microprocessor unit itself, block diagrams for the functional areas of the chip, and an introduction to machine-level programming. There are several sets of review questions, with answers, to check your progress. These review questions are a feature of all ten sections. If you answer the questions honestly, you may occasionally find that your understanding of the section's material was incomplete. You'll have a chance to go back and fix that before you get stuck later. This is serious work, and you'll be lost soon if you don't keep up with it as you go. Finally, Section One offers you your first experiment with the trainer. It's an introduction to machine language, the 8088's registers and addressing modes, and some hands-on work with some of the instructions described in the text.

The experiments, while geared to the trainer computer, are worth the time to examine, even if you're using the H/Z-100 computer. All of the experiments for the first six sections are located in Section Nine, the first section of Book Two. The first set of six experiments are all software-related and involve using the built-in line editor to write a short (or sometimes not so short) program, assembling it, and using the debugger and the ET-100's display to observe the effects of the program on the various registers and flags of the 8088 MPU. In many cases, there will be some modifications required to suit the computer you're using, since a number of built-in features of the ET-100 are involved. The experiments are well thought out, described and presented, and while some of the hardware interfacing can't be done without the trainer, the basic principles are clear enough that you'll profit from going through the material. Later experiments stress the hardware more than the software, so the benefits decrease, and you won't be able to do the "big one", the light pen, but there is still plenty to be learned.

Section Two begins with an introduction to assembly language. The great advantage of assembly language over the machine language of the previous section is the relative ease of creating the flow of instructions in a language that is somewhat more English-like. You'll be introduced to the format of an assembly language statement using ASM-86. You'll find out about assembler directives (those instructions that the MPU ignores but the assembler uses to manage the creation of the program), and you'll get to work with a few of the most common assembly language instructions. Flowcharting is covered, but briefly. You'll get to do another experiment, too. This one deals with assembly language format, the use of the ASM-86 assembler, and a chance to program with the arithmetic operators and branching instructions you saw in the text.

Section Three introduces different kinds of stacks, looping instructions, subroutines, and addressing modes. The 8088 uses a much more sophisticated stack structure than the 8080 and 8085. Heath spends a lot of time going over the proper way to go about setting up and using the stack, and there are a number of examples. The illustrations are also a big help. Experiment Three reinforces the main points with a series of programs to be entered, run, and the results examined. I found that even without the trainer, the use of Microsoft's DEBUG utility allows you to see what's going on in the stack and in the areas of memory you're working with. Remember that sometimes little will appear on the screen of your computer; the trainer relies on it's own ROM firmware to drive it's display, so you won't always get the ET-100 display the manual tells you to expect. This can usually be worked around, so shouldn't put you off.

Section Four goes into detail on expanded addressing, one of the most important differences between the 8088 and other micro processor chips. The 8088 can directly address far more memory than the 8080, 8085, or Z80 can, and the technique of doing this is central to making the most of the 16-bit computers. You'll get to work with some more arithmetic instructions and be exposed to the bit manipulation instructions. The corresponding experiment deals with Based Index addressing, and the arithmetic instructions and the bit manipulation instructions you saw in the main text. The experiments are somewhat more difficult here, but if you've done your "homework" and followed the sequence of the course, there probably won't be any insurmountable problems.

Section Five and the fifth experiment deal with memory segmentation. In my judgement, this was the most difficult section since the concepts were completely new. It's essential material, because segmentation is the key to getting to that megabyte of memory that the 8086 and 8088 can address.

Section Six presents data handling. This includes memory-mapped input/output, I/O addressing, I/O ports, interrupts, and string I/O. Memory-mapped I/O treats a peripheral device as if it were a memory location, and is a technique that's useful for interfacing various devices to the computer. For instance, the television terminal or monitor and the light pen are memory- mapped peripherals in the sense that the computer handles data from these devices as if it were data in a particular memory location. This section will set you up for the hardware interfacing experiments that follow. Experiment six is the last one in Section Nine. You'll be exposed to some new MPU flags and the various interrupt routines available with the 8086 and the 8088.

Section Seven marked a dividing point in the two text books. Together with Section Eight, it deals with hardware interfacing, and is specifically geared to the ET-100 trainer. If you don't have it, you can still read along and gain some insights into the process. Most of what's presented can be applied to the H/Z-100 computers, but will require some translation to allow for the hardware differences. In fact, if your primary goal is to learn interfacing for some specific application, you might seriously consider the purchase of the trainer even if you already have a full-scale computer.

Section Seven will give you hands-on experience with the control lines of the 8088 MPU, the clock generator, memory limit determination, and the differences between multiplexed memory and demultiplexed memory. Section Eight continues with coverage of types of memory (ROM and RAM), peripheral interfacing using the Programmable Peripheral Interface (PPI) chip, parallel/serial data conversion, and serial data communication. The experiments for the interfacing instruction are all contained in Section Ten. By the time you're finished with Section Eight, you will have had a chance to install a simple working light pen and test it out. While it's applications are limited, you won't be able to directly apply the pen to the H/Z-100, you will receive a good insight into the workings of the system, and may have some ideas for applications to other areas.

The four appendices at the end of Book Two include a 25 page review of numbering systems for the benefit of those who may not have been exposed to binary, hexadecimal, binary coded decimal, and the conversion of one to another system. There's also a brief review of the ASCII alphanumeric codes.

Appendix B is a listing of the 8088's machine coding instructions. These are the instructions to which the MPU responds directly. The appendix gives you the bit pattern for the instructions and the equivalent hex code. If you ever have to do work at the machine (binary) level, this set of tables will be useful.

Appendix C is a useful summary of computer arithmetic, how the MPU does its computations, and how you can hand-execute the arithmetic in the same manner as the MPU so that you can check on what's going on "inside". It's a bit of basic instruction, like Appendix A, and it illustrates Heath's intent to provide a complete course of instruction, one which won't require the user to go searching for some necessary piece of information needed to understand part of the course.

Appendix D is a copy of the Intel instructions as recognized by ASM-86. This is the "bible" for writing assembly language programs using the ET-100's built-in assembler. The listing gives the instruction mnemonic, its full name, what the operation does, how the MPU flags are affected, and a brief description of what goes on in the computer when that instruction is executed. The bit pattern of the instruction is shown, along with a few examples. The fact that this is the largest section of the two books indicates the vast number of instructions contained in the 8086/8088 instruction set. It's somewhat daunting when you first look at it and realize that these are the instructions you must master to be able to program the 8088 in your trainer or computer. It's also exhilarating when you finally complete the course and are able to conjure with these same instructions.

Finally, Appendix E contains the Intel data sheets for the 8088 MPU. If you have a technical bent, these 27 pages will make fascinating reading. Otherwise, just keep them for that one-in- a-million chance that you may have a question on the deep-down workings of the chip. It's all here, but it was written by engineers for engineers, and it will take some digging to get the answers out. Also in Appendix E is a very detailed six page index.

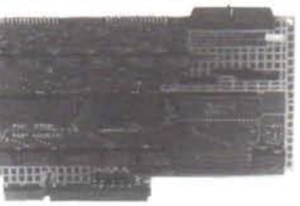
I'd like to add that since I'm not particularly adept at assembly language programming with ANY chip, 8080, 8085, or 8088, I approached the EE-8088 course with a degree of apprehension that might have been reduced had I a better grounding in assembly language for some other MPU. If you feel intimidated by the 8088 and its huge and complex instruction set, you might consider starting out with Heath's 8080/8085 assembly language course (EC-1108). If you have an H/Z-100, it will apply to the 8-bit side and help give you a more complete education.

On the whole, the course strikes me as well-written by someone (or a group of people) who honestly know what they're doing, and feel a genuine interest in getting their knowledge across to the beginning programmer in the least painful and most effective way. Keying the material to a trainer is a proven concept, highly successful with the ET-3400 trainer. There's no reason to assume the EE-8088 and the ET-100 trainer will be any less successful. For $99.00, the course is a good value, and for $995.00, the trainer seems like an ideal companion. The course is certainly as good an approach as buying four or five of the $20.00-25.00 programming books appearing on the market now.

H/Z89 PERIPHERALS from **SECURED COMPUTER SYSTEMS**

2/3 PORT SERIAL — CARD — I/O
PORT PARALLEL

"... not your typical vanilla-flavored serial and parallel interface ..."



Features:

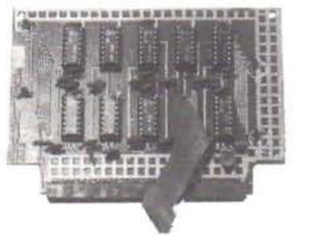
Chip independent design • Reduces computer data buss loading from 3 to 1 • Choice of Centronics or Epson parallel drivers for HDOS or CP/M • Complete documentation and installation instruction.

• 2 Serial Ports
• Supports: Ring Input, External Clock, Auto Dialer
• 3 Port Parallel with 2 Level Interrupt Control
• Fully compatible with all models of H/Z 88, 89, 90 using CP/M or HDOS.
• Fully tested, 90 day warranty, two serial cables and a parallel cable (internal to computer) and software driver.

PRICE $199.00
Shipping & Handling $10.00

16K RAM EXPANSION CARD

Expands your H/Z89 RAM Memory capacity to a FULL 64K!



Fully compatible with:
H/Z 89 • H/Z 88 • Magnolia Microsystems CP/M and disk drive I/O interface cards

NOW INCLUDING SUPPORT MOUNTING BRACKET

Featuring:
Complete installation instruction • 90 day Warranty
Field reliability record now entering its 21st month

Now Only $65.00

HDOS is a registered trademark of Heath Company
CP/M is a registered trademark of Digital Research

Shipping & Handling $5.00

PRICES ARE LESS SHIPPING AND TAX IF RESIDENT OF CALIFORNIA
MAIL ORDER: 12011 ACLARE, CERRITOS, CA 90701 (213) 924-6741
TECHNICAL INFO/HELP:
8575 KNOTT AVE., SUITE D, BUENA PARK, CA 90620 (714) 952-3930
Terms and specifications subject to change without notice.

ZENITH data systems
SERVICE CENTER

Hey Boys and Girls, What Time Is It?!!

(A Review of the H891-A Multi-Function Utility Board From Micro Widget Works, Inc.)

Pat Swayne
Software Engineer

The H891-A Multi-Function Utility Board is an add-on board that adds six functions to your H/Z-89 or Z-90 computer. The functions are a clock/calendar with battery backup, a Centronics compatible parallel printer port, three user configurable parallel ports, provision for a math processor chip, EPROM programmer support, and I/O bus expansion. The manufacturer, Micro Widget Works, also produces similar boards for H8 computers and for S-100 bus computers, including the H/Z-100. However, only the H89 version will be reviewed in this article.

Assembling the H891-A

The H891-A is available in both kit (reviewed in this article) and assembled form. The kit version comes packaged in a small mailable box with the board well padded. The board itself is a first quality 4 layer PC board with part labels and solder masking (the green "paint" you see on many PC boards). The power and ground connections are on two planes sandwiched inside the board and all other connections are on its outer surfaces. This kind of board construction is normally used when higher part density is needed, and when noise suppression is a consideration.

The solder supplied with the kit deserves mentioning because it is a new kind, with a water soluble flux core. After you finish soldering all parts into the board, you can wash all of the flux off of the back with plain tap water. The result is a clean board with a factory-built appearance (assuming you are a good solderer). The water does no harm to the board as long as you dry it off before applying the power. Many commercial operations are turning to water soluble flux these days to avoid using environment-fouling solvents, but this kit was the first home use of it I have seen.

The instructions for building the kit are clear and easy to follow, with check-off steps Heathkit style. There are not nearly as many illustrations as with a Heathkit, but where there might be misunderstanding without them, they are provided. One area of possible trouble for new kit builders is the way resistors are labeled on the PC board. On Heathkit boards, there is usually an outline of each part drawn on the board, but on this board there is only a line between the mounting holes for each resistor. Since the board is dense, and there are feed through holes very close to some of the mounting holes, an inexperienced builder could possibly mount a resistor improperly. If you are inexperienced and decide to build the kit version of the H891-A, be sure to watch out for that.

There are several places for jumpers on the board for setting up the various ports and devices. The instructions say to solder wire jumpers where they are needed, but I chose instead to solder pin connectors into all of the jumper holes, and use jumper plugs on the pins where jumpers are needed. Pin connectors are available from the Heath parts department as part no. 132-1075 for a 25-pin strip (can be cut to length desired). The jumper plugs are part no. 132-1041.

There was one part of the wrong type supplied with my kit, but Micro Widget Works mailed me the correct part (first class mail) when I called them. All of the other parts were OK, and the board worked without any trouble shooting required when I completed it. They also sent me a printer cable so I could test the parallel port on my Epson. I had a cable which at first I thought would work, because it had the correct connector at each end (a Centronics connector at one end, and pin connector at the other end). The cable was for a ZTX-10, and it was "straight through", that is, the pins at one end corresponded directly to the pins at the other end by position (though not by pin number, because the numbering conventions for Centronics and pin connectors are different). However, the connections on the H891- A are not in the correct order for straight through connection, so I had to get a special cable from Micro Widget Works.

The H891-A decodes its own ports for all of its devices, so it requires additional signal lines to those supplied on the right hand H89 bus pins. To get those lines, a small flat cable with dip connectors is provided which plugs into the CPU board, into the socket normally occupied by U550. The other end plugs into the H891-A, and the U550 IC removed from the CPU also plugs into the H891-A. The H891-A can then be plugged into any right side slot except the rightmost one. Photo 1 shows the H891-A (left board on the right side)

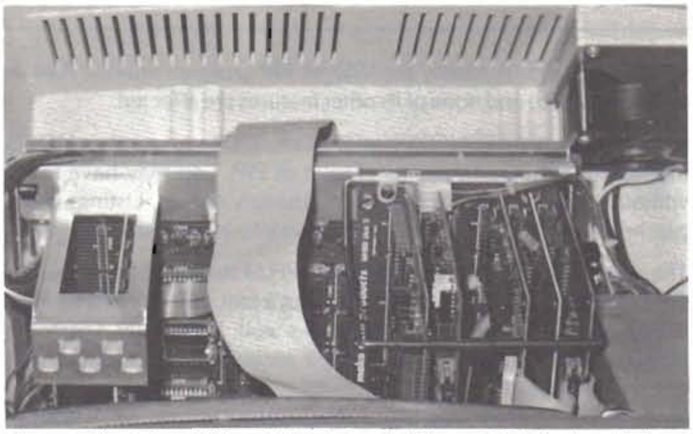


Photo 1. The Micro Widget H891-A installed in my H89 (the board closest to the center). The pins at the top of the board are for a parallel printer.

plugged into my system along with 4 other boards. To accomodate all of these boards, I use the Mako MH89 Plus 3 extender, reviewed in REMark #34. This board also re-routes U550 for port decoding, so I was a little concerned about loading on the devices supplying it, but everything seems to work OK.

Software Support

My H891-A came with three 5.25-inch hard sector disks containing support software. The software is also available on soft sector disks. Two of the disks were HDOS, and contained a utility for setting the clock/calendar, a device driver for the printer port, a sample BASIC program that accesses the clock/calendar, and modified HDOS system files. One of the HDOS disks is bootable, and is used to produce your own modified HDOS system disks. When you boot a disk with their modifications, it prints out the time and the day of the week, and when it gets to the part where you would normally enter the date, it just prints out the correct date itself and goes on. PIP is modified so that the time as well as the date is stored in the directory, and when you type CAT or DIR, you get a listing of files that shows a time as well as a date beside each file name. The Widget mods use the S.TIME cell that the HDOS designers provided for storing the time, but never used. They also used one of the unused bytes in each HDOS directory entry for time storage. A special device driver they supply maintains the correct time in S.TIME so programs can get the time there instead of having to access the H891-A directly.

The only bad thing about all of this is that I had a patch of my own in PIP (from REMark #27) that caused PIP to list files as it copied them, and their PIP did not do that. So I worked out the following patch for the Micro Widget version of PIP, which can be done with the PATCH program supplied with HDOS.

```
>PATCH                                     051244 = 101/43
PATCH Issue #50.06.00                      051245 = 124/176
File Name? PIP                             051246 = 110/377
Patch ID? IFOJIC                           051247 = 040/2
Prerequisite Code? IFBEIADPGEFFCF          051250 = 124/43
Address? 56275                             051251 = 105/247
056275 = 052/315                           051252 = 103/302
056276 = 030/236                           051253 = 110/245
056277 = 064/51                            051254 = 115/51
056300 = 021/^D     (type control-D)       051255 = 111/76
Address? 51231                             051256 = 103/12
051231 = 040/240                           051257 = 101/377
051232 = 103/76                            051260 = 114/2
051233 = 117/1                             051261 = 012/52
051234 = 116/377                           051262 = 103/30
051235 = 124/0                             051263 = 117/64
051236 = 101/52                            051264 = 122/311
051237 = 103/315                           051265 = 122/^D
051240 = 124/56                            Address? ^D
051241 = 040/43                            Patch Check Code? KIKFONJE
051242 = 110/43                            PATCH Issue #50.06.00
051243 = 105/43                            File Name? ^D
```

When this patch is made, the Widget PIP program will list files as they are copied, and none of its other features are affected.

If you use Super SYSMOD (see REMark #37), or another HDOS modification that relies on its version of PIP, you may have to do without having the time recorded in directory entries. Listings of the patches made to HDOS by Micro Widget Works are not supplied.

The printer device driver supplied is LPH24 modified to work with the Centronics parallel interface. It is not a fancy device driver, having only a few SET options, but it works well. Personally, I prefer to interface a printer serially, because the required cable is smaller and can be made much longer. If a parallel cable is more than a few feet long, "noise" can get into it and affect printer output. On the other hand, if your printer only has a parallel interface, or if all of your serial ports are used, the H891-A printer port is a good way to go. It decodes its own port for the printer, so there is no conflict with any of the standard ports in your computer.

The CP/M software supplied includes a utility to set the clock/calendar, a utility to test the board, a BASIC program that demonstrates using the clock/calendar, .LIB files and a SUBMIT file for modifying the BIOS, and a pre-assembled BIOS with modifications installed for those using hard sector disks only. I assume that if you get soft sector disks, the assembled BIOS would be for them. I found a "bug" in the file H1ABIOS4.LIB, which is one of the BIOS modification files. Just after the ENDIF at the beginning of the file, you should insert the line:

IF H1APP

Without this line, a portion of the parallel printer driver will be assembled into the code even if you don't want it. If the CP/M assembler was as "picky" about the correct use of IF's and ENDIF's as the HDOS assembler is, an error message would have been generated when the BIOS was assembled. Maybe someone should translate the HDOS assembler to CP/M.

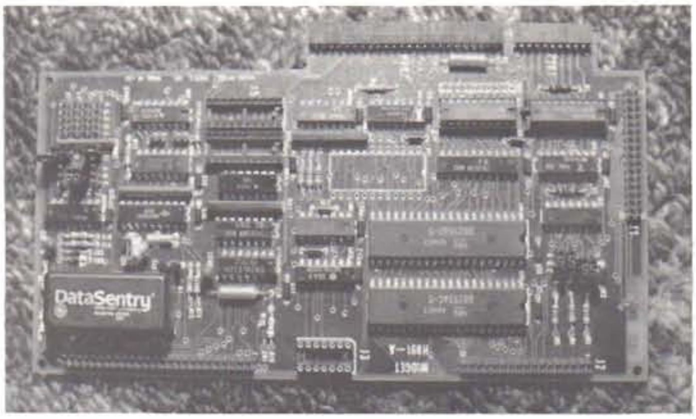


Photo 2. A close-up of the H891-A. The large device marked "Data Sentry" is a rechargeable battery. Above it are several jumper pins, which I used instead of wire jumpers for configuring the board.

The SUBMIT file uses the CP/M ED editor to modify the Heath/Zenith CP/M 2.2.03 BIOS.ASM with the .LIB files. The result is a BIOS with code for making use of the clock/calendar and the parallel printer port. Conditionals allow you to select or de-select the new code. The clock calendar code makes use of the "TOD" (Time Of Day) code that is already in the BIOS, but is normally unused. The time and date are maintained in memory cells that are in a fixed place in reference to the clock interrupt service routine (that is, the computer clock, not the H891-A clock). Therefore, a program can locate and access the time and date by loading the address at the clock interrupt vector. All of this provides a CP/M equivalent of the HDOS S.TIME and S.DATE cells.

The instructions for installing the software, both HDOS and CP/M, occupy a large portion of the instruction manual, and are very complete. The HDOS instructions seem a little "dangerous" though, because they require you to remove the write protect tab from one of the Widget-supplied disks and delete some files from it after copying them to another disk. I would recommend that you DUPlicate the disks before you start, and use the duplicate copies. If you do not have an HDOS disk duplication program (available from HUG and other sources), you can use the CP/M DUP program to duplicate an HDOS hard sector disk, as long as the volume number is zero. The Widget disk on which you must delete files just happens to be volume zero.

Untested Features

The H891-A provides an interface to the AM9511A or AM9512

Math Processing Unit (MPU), and a socket is provided on the board for one. The H891-A decodes a port for the MPU that does not interfere with any standard ports. The MPU chip is quite expensive, and is not supplied with the board, but is available at extra cost.

The H891-A also provides three user configurable parallel ports, driven by 8255 Programmable Peripheral Interface chips, and an expansion bus that includes all data lines and the lower 8 address lines for I/O port use. I did not test the expansion bus or the configurable ports.

The H891-A also provides a direct connection to the Optimal Technology model EP-2A-79 EPROM programmer. Since I do not have the programmer, I was unable to test the interface for it.

Documentation

The documentation supplied with the H891-A kit consists of 79 pages of construction and installation instructions and some data sheets on the Centronics interface, the clock calendar chip, the 8255's used for the parallel ports, and brief data sheets on the AM9511A, the AM9512, and the EP-2A-79 programmer. The parts of the documentation produced by Micro Widget Works appeared to have been prepared using word processing software and a daisy wheel printer. All of the documentation was punched for a three ring binder, but was not bound. Schematics in large blue print form (three pages) are also supplied.

Performance

As I stated before, the H891-A worked perfectly as soon as I assembled it, with no trouble shooting required. However, when I tried to use it with my 4 MHz modification (see REMark Issue #34), the system would sometimes "lock up" during booting, when the speed was switched from 2MHz to 4MHz. Apparently, some kind of "noise glitch" was occurring when the speed was switched. I fixed the problem by replacing some resistor networks (RN2 and RN3) that are on the outputs of the data and address line buffers. The original networks connected two resistors to each output: 220 ohms to +5 volts, and 330 ohms to ground. That would cause the quiescent voltage to be around 3 volts. I am not a hardware engineer, but it seems to me that having the lines at 3 volts would not do much for noise immunity. The new networks I used are Heath p/n 9-118, and connect each line to +5 volts only through 1K ohms. To install them, you should cut off pin 10 of each network first. Then install each one with the cut off pin over the pin one hole on the board (in other words, install them backwards). You must also install a 9-118 resistor network at RN1 in the same way, which was left empty even though the schematic shows a network there.

The comments in the source codes for the clock/calendar utilities warn about strictly observing certain time delays when you access the clock/calendar chip, but it seems to work perfectly at 4MHz with no modifications to the utilities.

Summary

Overall, I was quite pleased with the board and the software supplied. This board and the modifications provided bring HDOS right up to date with the newest operating systems, with time as well as date stamping for files. The S-100 version should be an asset for Z-DOS users, because Z-DOS software is supplied which presumably would automatically update the time and date like the HDOS software does. Since the clock/calendar has a back-up battery, it always keeps time even while the power is off.

I was displeased with some inconsistencies in the software between HDOS and CP/M. For CP/M, you get complete source code for all modifications, but for HDOS, listings of the Micro Widget patches to the system files are not supplied. The clock/calendar set utilities for each operating system work differently, with different command syntax. I see no reason why they could not be identical. The board test utility supplied on the CP/M disk should be supplied in an HDOS version also.

The documentation was complete with regard to construction and installation instructions, but could use more discussion of how the board works. Specifically, there should be a chart of what each jumper on the board does, and there should be more discussion of how the software works with the hardware. As it is, the only way you can figure out the jumpers is to study the schematic.

In conclusion, I would recommend the Micro Widget Works H891-A for any H/Z-89 or Z90 owner who needs a clock/calendar and a Centronics compatible parallel port. If you just need the port, there are other boards that are less expensive, but the combination of features on the H891-A are hard to beat. The other things on the board beside the clock/calendar and printer port are "icing on the cake". The H891-A is priced at $239.95 for the assembled and tested version, $190.00 for the kit version, and $110.00 for the bare board. H8 and S-100 versions are slightly less expensive (probably due to not having to use a multi-layer board). For more information, contact Micro Widget Works, Inc., P. O. Box 15185, Santa Ana, CA 92705, phone (714) 544-8252.

The ILLUSTRATOR

The **ILLUSTRATOR** is a full-featured graphics drawing program for use with the Z-100 pixel graphics (w/wo color) or the H89/H19 IMAGINATOR pixel graphics option. No need for a lightpen.

Features Include:

• Turtle Graphics	• Area Fill	• 64 Colors
• Rubber Banding	• Box Fill	• Color Define
• Line Drawing	• Circle Fill	• Color Pattern
• Box Drawing	• Diamond Fill	• Screen Save
• Circle Drawing	• Copy Area	• Screen Restore
• Ellipse Drawing	• Erase Area	• Area Save
• Diamond Drawing	• Text Mode	• Area Restore
• Screen Print	• Invert Mode	• Compacted Files
• Dot Cursor Control	• Help Display	• more. . .

ONLY $89.95!

HDOS version for H89, H8, requires IMAGINATOR
ZDOS version for Z-100, color memory optional
Supports many dot graphics printers. Call for info.

NEWLINE SOFTWARE
P.O. Box 402, Littleton, MA 01460 (617) 486-8535

NAME ______ CHECK ONE
STREET ______ ☐ H89, H8/H19, HDOS
CITY ______
STATE ______ ZIP ______ ☐ Z-100, ZDOS
Send me ______ "The ILLUSTRATOR" program(s) at $89.95 each.
Check one: ☐ payment enclosed ☐ send COD (add $4.00)
Send order to:
NEWLINE SOFTWARE, P.O. BOX 402, LITTLETON, MA 01460
Foreign orders: add $3.00 Airmail, $10.00 for non-U.S. checks

HDOS is a trademark of Heath Company
ZDOS, Z-100 are trademarks of Zenith Data Systems, Inc.
IMAGINATOR is a trademark of Cleveland Codonics, Inc.

NOTICE

Heath/Zenith Support Vendor

This could have been your advertising space to reach over 20,000 HUG members!

Call Donna at HUG today for more information (616-982-3794).

WATCHWORD

The word processor and full screen editor for the Z100 and ZDOS.

FRIENDLY - FAST - FLEXIBLE

- *See subscripts, superscripts, underlining and boldface directly on the screen*
- *Create your own fonts and special characters*
- *Remap any key and configure to your printer*
- *Written in native 8088 assembly language for fast response*

Other features: centering, formatting, microjustification, arbitrary line length, automatic horizontal scrolling, split screen, macros, and color.

Demo available at all Heathkit Electronics Centers.

For additional details, send a SASE.
Demo Disk: $3.00
WatchWord with Manual: $100.00

S & K Technology, Inc.
4610 Spotted Oak Woods
San Antonio, TX 78249
512-492-3384
ATTN: Steve Robbins

USE ALL YOUR SPECIAL FUNCTION KEYS

With WORDSTAR™

WSKEY™: Now you can take the mystery out of WordStar with SKILL DATA's WordStar enhancement, which implements all twenty-one of the H/Z89-19 function/pad keys or all twenty-three Z-100 labeled key commands.
Function key commands are labeled by a twenty-fifth line banner, which can be toggled on and off by you during your session.

With dBASE II™, ZIP™, SuperCalc™

DBKEY™, ZPKEY™, and SCKEY™: Just type your favorite SKILL DATA key command. All function key commands are labeled by a twenty-fifth line banner, which can be toggled on and off by you during your session. Pad keys also function and send multiple key inputs with a single stroke. All previous command key sequences are still available for the old and painful ways. (See the review of DBKEY and ZPKEY in September 1983 *dNEWS*.)

Just $29.95 for each program (plus shipping charges shown below).
H/Z89-19 diskettes are 5.25" 10HS, require CP/M 2.2.03.
Z-100 diskettes are 5.25" SS; specify CP/M-85 or Z-DOS.

Complete and mail this order form with your check or money order today!
Mail to SKILL DATA, P.O. Box 1943, Olympia, WA 98507.

Yes, I want to use all my special function keys! Send me:

☐ WSKEY—keys to WordStar: ☐ for H/Z89-19 ☐ for Z-100	$29.95
☐ SCKEY—keys to SuperCalc (for CP/M only)	$29.95
☐ DBKEY and ZPKEY—keys to dBASE II: ☐ for H/Z89 ☐ for Z-100	$29.95

For Z-100 diskettes, specify operating system: ☐ CP/M-85 ☐ Z-DOS

Name ____________
Street ____________
City ____________ State ____________
Zip ____________ Phone ____________
Visa/MC Card # ____________ Expires ____________

All orders shipped by first class mail. Include $3.00 per order for shipping and handling. $5.00 for overseas orders. Payment by check, money order, or Visa/MasterCard. Allow 2-4 weeks for delivery. 206/352-0669 (evenings).

SKILL DATA, P.O. Box 1943, Olympia, WA 98507

H/Z-100 COLOR GRAPHICS SOFTWARE

Want to explore **Z-BASIC** color graphics the easy way? Try these packages from **MICROSERVICES.** All you need is a color configured computer with 128K memory, a color monitor, and **Z-BASIC.**

ZANIMATE will help you rapidly draw and paint each frame in an animation sequence. Select frame sizes, vary location and time between each frame. Draw the background too, with the graphics editor **$64.95**

ZPALETTE, the original H/Z-100 palette program, contains a graphics editor for point-plotting images which can be painted in 92 hues! File your images for access by other **Z-BASIC** programs, and transfer imagery from one file to another.
NEW PRICE **$59.75**

ZPATTERN helps you determine the quality of your color monitor and if it needs maintenance.
NEW PRICE **$24.95**

Include 3% handling/postage ($2.50 minimum) — California add 6½% tax. Send check or money order to:

MICROSERVICES
P.O. BOX 7093
MENLO PARK, CA 94026
PHONE: (415) 851-3414

FRAZZLED • HYSTERICAL • FRUSTRATED • TENSE •

DON'T PANIC

PROTECT YOURSELF FROM DATA LOSS FOREVER
with
RESCUE & REPAIR

Did you press the wrong key. . .accidentally hit "Reset". . ."quit" without saving to disk. . .find your "disk full". . .or have your computer "lock up?" Hours of "lost" work can be saved with **RESCUE & REPAIR.**

- **RESCUE** will get it from memory and put it on disk with a one line command.
- **REPAIR** will restore the file in a few minutes using simple but powerful editorial commands.

If you program, word process, or do extended data entry, you need **RESCUE & REPAIR.**

It will pay for itself the first time you use it.

Available for most computers using CP/M 2.2.

21st CENTURY DATA, INC., P.O. Box 1139 • Dept. B2 • Solana Beach, CA 92075 • (619) 755-6218

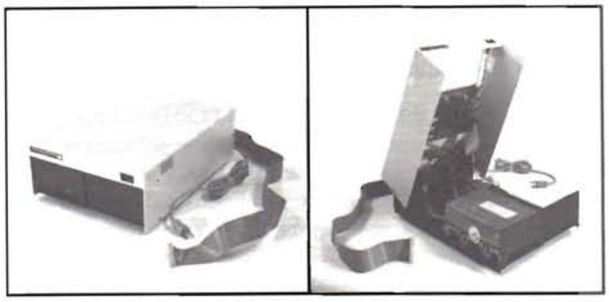
TO ORDER: RESCUE & REPAIR disk† and manual $45.00

† Specify computer, make, model and drive type

Shipping postpaid in the U.S., Canada and Mexico. Other countries please add $5.00 per disk for shipping. California residents add 6% sales tax.

CP/M is a registered trademark of Digital Research.

NOW 12 MEGABYTE
(CDR-10M) $2995



WINCHESTER SYSTEM
For the Heath/Zenith Computer

Systems complete with software, case, power supply, signal cable and interface.
Runs with CP/M, on the H/Z89 & H8 (with Z80 card).

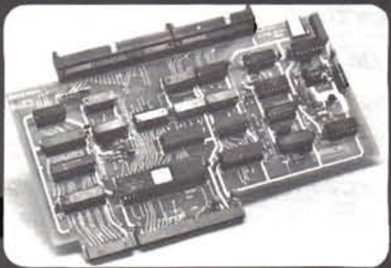
- Switching power supply
- Expansion for backup installations
- Auto attach BIOS
- Hard disk utilities
- Formatting program
- 1 year parts & workmanship warranty

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

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

NEW PRICE $495

Includes controller board CP/M boot prom, I/O decoder prom, hardware/software manuals BIOS source listing. HDOS 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

File Handling With Compiled Programs In CP/M

B. L. McFarland
17175 Gunther Street
Granada Hills, CA 91344

After a BASIC program has been debugged and used for a while, the program usually seems to operate too slowly, and it is desired to speed things up. We then use BASCOM to obtain a compiled version, or we write a FORTRAN program to do the same job. This usually makes a marked improvement in the operational speed of the program. However, the program is still not of commercial quality. One of the superior features of commercial programs or programs written in assembly where the writer has "complete control" as compared with our home brew versions written in BASIC or FORTRAN is the ability to optionally include a data file name in the execute statement. An example of this is the familiar input from the system level of:

```
A>MBASIC CHEAPCALC
```

The Microsoft compilers (BASCOM and F80) do not have the ability to include this capability as a default condition in the compiled programs. Both compilers produce executable programs that read the input buffer up to the first illegal file character, and thus cannot detect a second file name. In addition, the FORTRAN compiled programs have their own unique way of handling file addressing that gives the new user trouble.

This article presents one way to include optional file addressing in your BASIC and FORTRAN compiled programs as well as a way to allow conventional file naming in FORTRAN.

CP/M Default Buffer Area

Before discussing the subroutines themselves, we need to discuss briefly the CP/M Default Buffer Area set aside by CP/M for input. The interested user should read the discussion in the CP/M 2.0 Interface Guide starting on page 5 for more details then I will give here. This area starts at $80 (128 decimal) and is automatically used to store the users keyboard commands that are issued from the system level (e.g. A> prompt visible on the screen) among other uses. The area is 256 bytes in size ($80- $100 where the program starts), although only the first 32 bytes are of interest here.

The first byte (at location $80) is a counter for the number of characters still in the input buffer. Thus $80 is 0 if the buffer is empty and greater than zero if there are characters left from the input entry. Therefore a PEEK (128) from BASIC or a CALL PEEK (128) from FORTRAN can be used to determine if there is usable input left in the buffer. This forms the principle behind the subroutines discussed in this article.

File Naming Conventions

From the system level or from BASIC, addressing a disk file has the form:

```
B:FILE.EXT or FILE.EXT or even FILE .EXT
```

If a colon (:) is present in the input,the character to the left specifies which drive to address and must be within the limits established by the CONFIGUR program furnished with the CP/M system. The filename itself can be up to 8 characters, but can be as few as a single character while the optional file type designation is automatically the next 3 characters following a period (.) in the input. Thus the file name can be as long as 14 characters, or as short as a single character as defined by the user.

From FORTRAN, the conventions are different. A specific filename format must be followed when input during program operation by a READ instruction such as:

```
    LUN=5
    WRITE(1,5) 5 FORMAT(1H ,'Enter File Name ')
    READ(1,10) FILE
 10 FORMAT(11A1)
    CALL OPEN(LUN,FILE,0)
```

The file name input must be entered as follows:

```
FILE    EXT
```

The filename itself must be 8 characters (e.g. fill the name with blanks to 8 characters), there is no period, and the disk drive cannot be defined. The example uses the default drive as the input device. Thus the input is considerably different from the conventions the user is familiar with. The filename is always 11 characters long. Any deviation from this format will cause a program error and an irritated user.

Subroutine Descriptions

It was to get around the last problem in FORTRAN programs that caused the development of the file handling subroutines that are discussed here. Consequently, the FORTRAN subroutine will be considered first. As written, the subroutine is called from the main program with the logical unit number (LUN) as an integer argument in the call statement.

```
CALL FOPEN(5)
```

If the compiled FORTRAN program (called PROG here) is entered without a second file name at the system level

```
A>  PROG
```

then the user is the prompted on the screen with

```
Enter data file name:
```

The data file name conventions for input are the same as in BASIC, namely B:FILE.EXT.

If the compiled FORTRAN program is executed by entering a second file name after the executable program name,then

```
A> PROG B:FILE.EXT or
A> PROG FILE.EXT or
A> PROG FILE
```

The input file name is converted to the required 11 characters, and the alphabetic drive designation (if entered) is converted to a numeric representation.

```
      SUBROUTINE FOPEN(LUN)
      BYTE INP(16),NF(11),BL,CM,PER,D1,D2,D3,COLON,D4,D5,D6
      DATA COLON,BL,CM,PER,D1,D2,D3/':',' ',',','.','I','N','P'/
      IPER=0
      ICOLON=0
      IDRIVE=0
     IN=MIN0(16,PEEK(128))
     IF (IN.LE.0) GOTO 55
     CALL POKE(128,0)
     DO 10 I=1,IN
     INP(I)=MAX0(32,PEEK(129+I))
10   CONTINUE
20   IF(INP(1).GT.32) GOTO 105
     DO 30 I=1,15
30   INP(I)=INP(I+1)
     GOTO 20
45   FORMAT(1H ,16A1,6I6,' ')
 50   FORMAT(1H ,3I6)
 55   WRITE(3,100)
100   FORMAT(' ENTER DATA FILE NAME: ')
      READ(1,102) INP
102   FORMAT(16A1)
105   ICM=16
      IDRIVE=0
      IPER=ICM
      IBL=ICM
      ICOLON=1
      DO 110 J=1,16
      IF(INP(J).LT.PER) IBL=MIN0(IBL,J)
      IF(INP(J).EQ.COLON) ICOLON=J+1
110   IF(INP(J).EQ.PER) IPER=MIN0(IPER,J)
      IDO=MIN0(IPER,IBL,ICOLON+8)-1
      IF(ICOLON.LT.3) GOTO 115
      IDRIVE=MAX0(INP(ICOLON-2)-64,0)
      IDRIVE=MIN0(IDRIVE,2)
115   DO 120 J=ICOLON,IDO
      JJ=J-ICOLON+1
      NF(JJ)=INP(J)
      IF(NF(JJ).LT.BL) NF(JJ)=BL
120   CONTINUE
      IDO=IDO+1
      DO 130 J=IDO,16
      JJ=J-ICOLON+1
      IF(JJ.LT.12)NF(JJ)=BL
130   CONTINUE
      IF(IPER.LT.16) GOTO 250
200   NF(9)=D1
      NF(10)=D2
      NF(11)=D3
      GOTO 275
250   DO 270 J=1,3
      IND=J+IPER
270   NF(J+8)=INP(IND)
275   CALL OPEN(LUN,NF,IDRIVE)
      RETURN
      END
```

Table 1. FORTRAN Subroutine Listing

Now let us discuss the logic used in the FORTRAN subroutine listed in Table 1. The subroutine must determine the following:

1. If a file name has been entered.
2. A drive has been designated.
3. An file type has been used (this is optional, but I like to use the file type to aid PIP transfers).

The program lines up to statement 30 interrogate the input buffer for content. If location $80 contains a zero, then the code jumps to statement 55 and asks for a file name. If $80 is greater then zero, zero is POKEd into location $80 to indicate a clear buffer. Then the DO loop at statement 10 transfers the buffer contents (up to 16 characters) into the INP array while deleting all control characters. The code between statements 20 and 30 deletes all leading blanks. This is needed since the user may enter several blanks between the executable program name and the data file name. After deleting all leading blanks, program control is passed to statement 105 where the file name is searched for a colon (:) and a period (.) by the DO loop at statement 110. THe file name is then transferred from the INP array to the NF array by the DO loop at statement 120. If a period has been found in the name signifying that the user entered an extension, control passes to statement 250 otherwise a default extension of INP is added to the file name (this can be deleted by defining D1,D2,D3 as blanks). At statement 275 the OPEN subroutine in FORLIB.REL is called to open the file.

BASIC Subroutine

The operation of the BASIC subroutine is similiar, but the code is much simpler since the file name does not have to be re-formatted to be acceptable to the computer. The BASIC subroutine is listed in Table 2. In this case the main program lines in a compiled BASIC program (also called PROG) will look something like this:

```
100 GOSUB 1200:IF LEN(FL$)>0 THEN 120
110 LINE INPUT "ENTER FILE NAME: ";FL$
120 OPEN "I",1,FL$
```

As with the FORTRAN program, the user of the compiled BASIC program has the option to either enter the data file name at execution time, or to enter it later from the program. Thus the command line to execute the compiled BASIC program could be any of the following:

```
A> PROG B:FILE.EXT
A> PROG FILE.EXT
A> PROG FILE
A> PROG
```

```
1200 IN=PEEK(128):IF IN>16 THEN IN=16
1210 FOR I=1 TO IN:J=PEEK(129+I):IF J<32 THEN J=32
1220 FL$=FL$+CHR$(J):NEXT:POKE 128,0
1230 IF J>32 THEN FL$=FL$+CHR$(J)
1240 X=INSTR(FL$,"."):IF X=0 THEN FL$=FL$+".DAT"
1250 RETURN
```

Table 2. BASIC Subroutine Listing

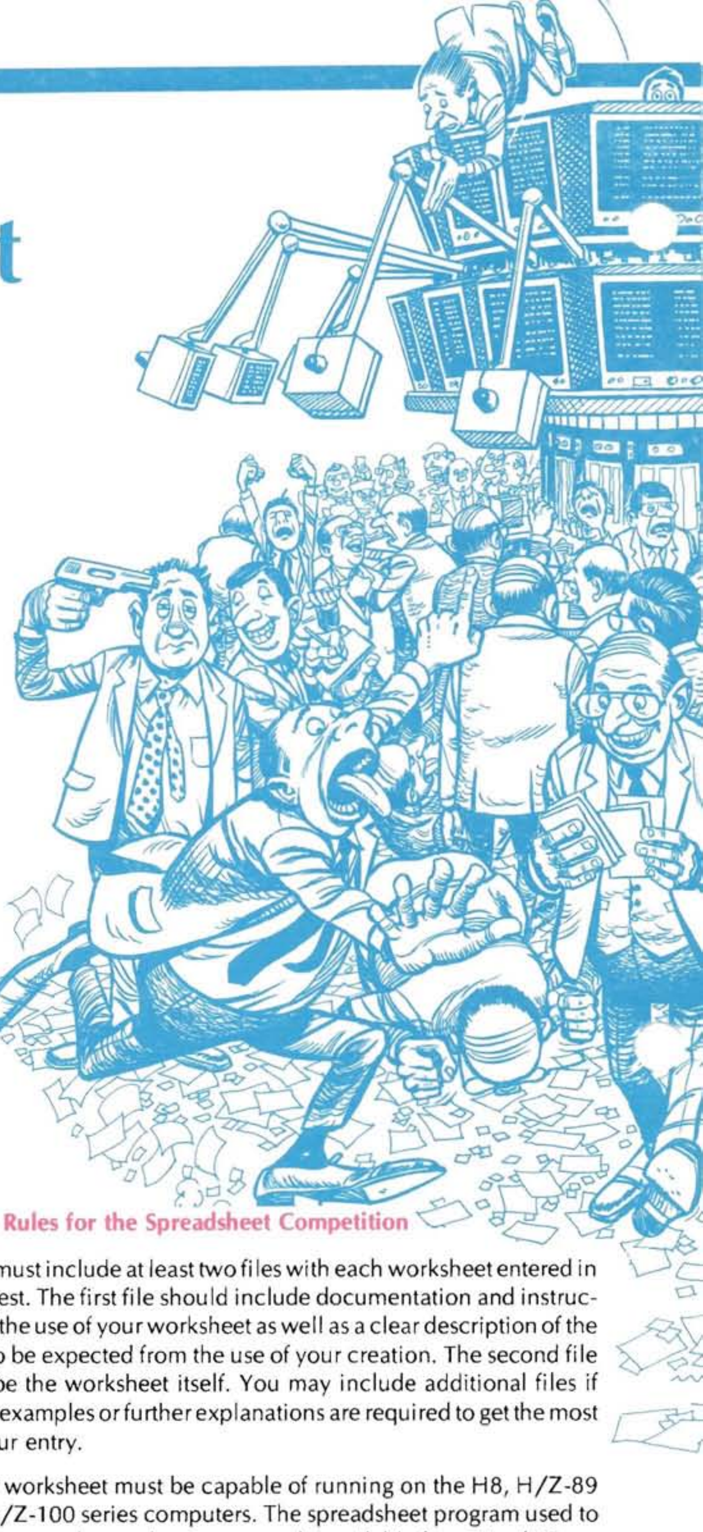
Unfortunately, the BASIC subroutine does not work unless the BASIC program has been compiled, however these two subroutines give the user of the compiled programs the option to either specify a data file at execution or to wait until prompted from the program itself as the commercial codes allow.

Changing your address?

Let us know. We don't want you to miss a single issue of **REMark**, send your change of address to:

Heath Users' Group
Hilltop Road
St. Joseph, MI 49085

It's Contest Time At The Heath/Zenith Users' Group



Bob Ellerton
HUG Manager

Are you sitting there staring at a blinking cursor wondering what to do with your spare computer time?

Have you created a really neat spreadsheet that you feel could be useful to other members of the Heath/Zenith Users' Group?

Have you created a slick game for yourself or the kids that's the greatest thing since PAC-something?

Are you interested in picking up an extra $1000.00 Gift Certificate for Heath or Zenith Data Systems products absolutely FREE?

If you have answered yes to at least one of these questions, read on!

The Heath/Zenith Users' Group will be sponsoring not one, but two software contests beginning April 1, 1984 and ending July 1, 1984. You may enter both contests if you wish. And, you may enter these contests as many times as you like.

Heath/Zenith Users' Group Spreadsheet Competition

The first of the two contests will be based on currently available spreadsheet programs from Heath/Zenith (e.g. SuperCalc, Multiplan, etc.). Entries to this category should be worksheets that are composed using one of the major spreadsheet programs. Any topic for your worksheet will be accepted (e.g. Tax Calculations, Payroll, General Ledger, Inventory, etc.).

Specific Rules for the Spreadsheet Competition

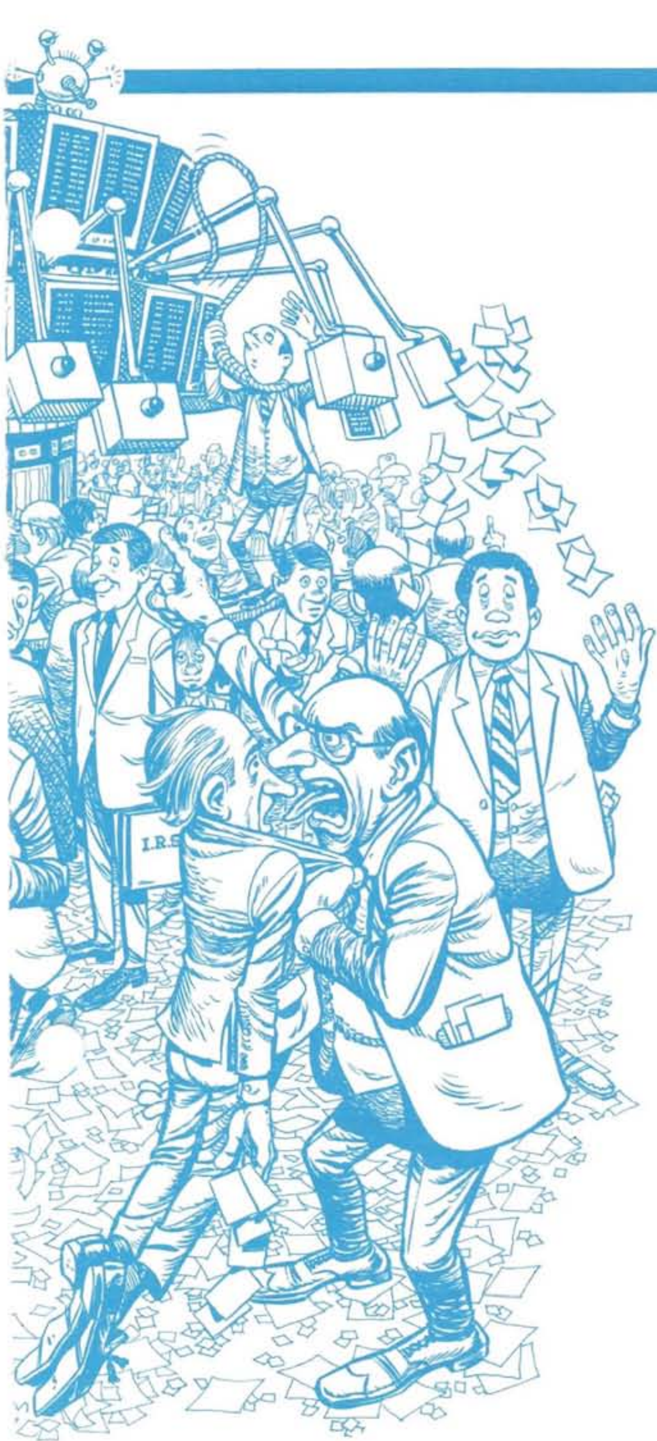
1. You must include at least two files with each worksheet entered in the contest. The first file should include documentation and instructions on the use of your worksheet as well as a clear description of the results to be expected from the use of your creation. The second file should be the worksheet itself. You may include additional files if you feel examples or further explanations are required to get the most from your entry.

2. Your worksheet must be capable of running on the H8, H/Z-89 or the H/Z-100 series computers. The spreadsheet program used to create your work must be one currently available from Heath Company or Zenith Data Systems (described in the Heathkit Catalog).

3. Entries to the Spreadsheet Competition must be sent to:

Heath/Zenith Users' Group Spreadsheet Competition
Hilltop Road
Saint Joseph, MI 49085

The second contest will concentrate on your ability to use the graphics facilities of your computer to build a game. This competition will be open to all languages currently available from Heath/Zenith or described in the Heathkit Catalog. Further, you may use languages from other sources providing that the finished software will run without having the user purchase software not found in the Heathkit Catalog.



General Rules:

1. All entries to either the Spreadsheet Competition or the Graphics Game Competition become the property of the Heath/Zenith Users' Group Software Library.

2. Each entry must be accompanied by the Program Submittal and Agreement Form found on page 27 of the January 1984 Issue of REMark. The form must be completed by you.

3. All entries must be submitted on disk and be accompanied by suitable documentation describing the purpose of the entry. Necessary information on setup and operation must be included for the reviewer.

4. Your entry must be clearly marked with the following words: **"Heath/Zenith Users' Group Contest Entry"**

If possible, these words should be included in your documentation file to ensure the proper handling of your contest entry.

Selecting the Winners:

1. The contest for both the Spreadsheet Competition and the Graphics Game Competition will be divided into two parts. During the first round, HUG Staff members will select those programs or worksheets that are thoroughly documented and perform as described by the author. These programs will then be placed on one or more disks containing similar games or worksheets.

2. Authors of programs selected to appear on a HUG Disk will then be informed that their work has been placed in the final competition with other similar programs.

3. AS A BONUS, authors selected for the final competition will receive any piece of Heath/Zenith or HUG software FREE along with a copy of the disk containing their work.

4. Final judging will be provided by the members of the Heath-/Zenith Users' Group via a postcard sent with each of the disks ordered from the HUG Library. The worksheet and graphics game receiving the most votes before November 1, 1984, will be chosen as the Grand Prize Winners in each of the two categories.

Grand Prize Winners

Two winners will be selected by popular vote, one from each category to receive a $1000.00 Gift Certificate from the Heath/Zenith Users' Group which can be used to purchase a variety of products available at any of your local Heathkit Electronics Centers or through Heathkit Mail Order Catalog. The two winners will be announced in the January 1985 issue of REMark.

Specific Rules for the Graphics Game Competition

1. Entries must include at least two files on the disk. One file should be the game itself. The remaining file must contain the necessary start-up instructions and documentation to allow proper operation of your game. Additional files may be included should you feel they are necessary for the end user to get the most from your creation.

2. Your entry must be capable of running on an H8, H/Z-89 or the H/Z-100 series computer using the various graphics modes available to each computer. Each game must be of your design and not a translation from another available computer video game.

3. Entries to the Graphics Game Competition must be sent to:

Heath/Zenith Users' Group Graphics Game Competition
Hilltop Road
Saint Joseph, MI 49085

7th Anniversary Sale!

Right now, you can start a subscription to *Buss: The Independent Newsletter of Heath Co. Computers* at the Special Introductory Price of $7 for 7 issues!

—A Savings of 45% Off the Usual Introductory Price—

Since April 1977, *Buss* newsletter has been dedicated to bringing system-specific news to users of Heath® and Zenith computer systems. During the past seven years, *Buss* has often been the first to bring its readers news of events in the Heath/Zenith community and future product plans. As *Buss* celebrates the completion of its 7th year of publication, we'd like to invite you to join the information exchange which is *Buss*—for only $7.

In each and every issue, you'll find out who the independent suppliers supporting Heath/Zenith are, what they're doing, what other users are doing, and where to get the products and information you need—all in one newsletter which comes to you twenty times a year, by first class mail.

As a subscriber to *Buss,* you can learn what other users and independent suppliers have to say about:

☐ H8	☐ printers	☐ business applications	☐ utilities
☐ H/Z89	☐ terminals	☐ home/hobby applications	☐ Z-DOS
☐ Z100	☐ disks and drives	☐ games	☐ HDOS
☐ compatible hardware	☐ compatible software	☐ graphics	☐ CP/M

Twenty times a year, you'll get the newsletter which helps you:

- Learn what new products are being developed and where to get them, in *Buss*'s new product announcements.
- Benefit from the advice of users when you write in to *Buss*'s Assistance Wanted column.
- Save money as you improve your system! *Buss* has a free "For Sale" listing for Heath/Zenith users.
- Avoid bugs in your system as you read how other users have solved problems they've found.
- Find out the news in the Heath/Zenith community which affects you and your system.
- Discover other users in your area through the Local Club Notes section of *Buss*.
- Get unbiased coverage on your system from the oldest independent newsletter for Heath/Zenith users. *Buss* has grown with the Heath/Zenith community, and has always been committed to supplying information particularly for the Heath/Zenith *user*.

Hurry to get your introductory subscription to* Buss *at a savings of 45%. This offer is good only during March and April of 1984.

Order your subscription today, and get in touch with the rest of the Heath/Zenith community!

Call Toll Free: Datatel™ **800/341-1522** (M-F 8 a.m.-9 p.m. Eastern Time) **and place your order today!**

Or clip this coupon and mail your order to:

Buss Dept. R34, 716 E Street S.E., Washington, DC 20003 • 202/544-0900

Yes! Sign me up for* Buss *at the low introductory price of $7 for 7 issues!

☐ I've enclosed my payment of $7. (Checks must be in U.S. dollars payable on a U.S. bank.)
☐ Bill me.
☐ Charge my: ☐ Visa ☐ MasterCard

Card # ____________________

Expiration date ______ (MC Code # ________)

Name ____________________

Address ____________________

____

____

This offer is valid only in the U.S., Canada, and Mexico. To start your overseas subscription to *Buss*, send $24 (U.S.) for 10 issues. (Price includes air mail postage.)

Your satisfaction with Buss *is completely guaranteed. If at any time you are not satisfied, your money will be refunded—in full.*

Buss, Dept. R34, 716 E Street S.E., Washington, DC 20003

I/O Port Baud Rate Programmer

Rick Swenton
19 Allen Street
Bristol, CT 06010

Back in October of 1983 I had performed Pat Swayne's 4MHz modifications which appeared in REMark issues 34 and 38. Having been impressed with the new operating speed of my H89A, I decided to try operating the H19 terminal portion at 19200 baud. These two factors when combined produced an outstanding increase in operating speed. Some programs such as basic sorting routines that output listings to the console now run at lightning speed! Of course, the Heath documentation does not guarantee that the H-19 will operate at 19200 baud without dropping characters, I decided to try it anyway. My system seems to work fine at 4MHz and 19200 baud. I was using CONFIGUR to change the baud rate of the computer and then switching the H-19 portion offline and entering "Escape rM" on the keyboard, and then going online again. I did not want to change the settings of the switches on the CPU and Terminal circuit boards for the reasons Pat Swayne mentioned in his 4MHz article - to keep the hardware compatible with obsolete systems which you cannot (or may not want to) modify.

Now the problems. Some programs send a master reset to the terminal (Escape z) for many reasons. (It's probably the easiest way to erase the 25th line.) This will reset the baud rate to that chosen by the dip switch on the terminal logic board. If your CPU is running at 19200 baud and an Escape z is sent to the terminal, you can no longer communicate with the CPU unless you switch the terminal offline and type "Escape rM", then switch back online. This escape sequence will restore the terminal baud rate to 19200. What you need is a utility program to conveniently set the baud rate of both the CPU and terminal, so that you can run at the standard 9600 baud when you wish, and then switch to 19200 baud for those long text editing sessions which used to seem to take forever to page back and forth. There is a good reason for running a slow terminal. Try listing a program in MBASIC with the console running at 19200 baud and you will find that it is hard to stop the listing at the right time to view the desired line numbers. At this baud

```
;
;       SETBAUD.ASM    I/O PORT BAUD RATE PROGRAMMER
;       BY RICK SWENTON  18-OCT-83
;
; THIS PROGRAM WILL CHANGE THE BAUD RATE OF ANY OF THE FOUR
; H-89 SERIAL I/O PORTS BY PROGRAMMING THE NECESSARY
; INFORMATION INTO THE 8250 ASYNCHRONOUS COMMUNICATIONS ELEMENT
; FOR THE SELECTED PORT.  IF THE CONSOLE PORT IS BEING CHANGED,
; THE PROGRAM WILL ALSO SEND THE CORRESPONDING ESCAPE SEQUENCE
; TO THE H-19 TERMINAL.  THE PROGRAM IS MENU DRIVEN AND WILL
; PROMPT THE OPERATOR FOR TWO SELECTIONS:
;
;       1 - THE DESIRED PORT            2 - THE DESIRED BAUD RATE
;
;
;
;
ESC     EQU     27      ;ESCAPE
CR      EQU     13      ;CARRIAGE RETURN
LF      EQU     10      ;LINE FEED
;
CRT     EQU     00E8H   ;CONSOLE PORT
B110    EQU     0417H   ;BAUD RATE DIVISOR FOR 110 BAUD
;
;
;
;       BDOS SYSTEM CALLS
;
BDOS    EQU     5       ;BDOS JUMP VECTOR
PSTRF   EQU     9       ;PRINT STRING FUNCTION
RCONF   EQU     1       ;READ CONSOLE CHARACTER FUNCTION
TPA     EQU     100H    ;TRANSIENT PROGRAM AREA STARTING ADDRESS
;
;
;
;
        ASEG            ;REQUIRED FOR M80 MACRO ASSEMBLER
        ORG     TPA
;
START:  DI                      ;DISABLE INTERRUPTS
;
; SAVE OLD STACK AND SET UP NEW ONE
;
        LXI     H,0             ;CLEAR H,L
        DAD     SP              ;GET STACK POINTER
        SHLD    OLDSP           ;SAVE IT
        LXI     SP,STACK        ;SET UP NEW STACK
;
SENDEV: LXI     D,DEVM          ;D,E POINT TO DEVICE SELECTION MESSAGE
        CALL    OUTS            ;PRINT MESSAGE
        CALL    INC             ;GET THE SELECTION FROM THE OPERATOR
        SUI     '1'             ;SUBTRACT ASCII BIAS
        JC      SENDEV          ;BAD ENTRY
        CPI     '4'-'1'+1       ;GREATER THAN 4?
       JNC     SENDEV           ;BAD ENTRY
        MOV     B,A             ;TEMP STORAGE
        LXI     H,BPT           ;POINT TO START OF BASE PORT TABLE
        LXI     D,0             ;CLEAR D,E
        MOV     E,A             ;STORE A IN E
        DAD     D               ;RAISE H,L BY A
        MOV     A,M             ;GET PORT VALUE FROM THE TABLE
        STA     BPTM            ;SAVE IT
        LXI     H,DMEST-9       ;POINT TO START OF DEV MESSAGE TABLE-9
        LXI     D,9
```

rate, they whiz right by you! You need the ability to change baud rates at will. Heath provides a program to set the baud rate of the printer. It is called SETLP.COM. This program allows the operator to change the baud rate of the printer port by typing a console command. This would be great to change the baud rate of the console port with one major exception: you need to change the terminal baud rate to match the new CPU baud rate. So I decided to write a program which can change the baud rate of the console port of the CPU as well as the terminal in one operation. While I was writing the program, it became evident that it would be easy to include the option of changing the baud rates of any of the four serial I/O ports and only update the terminal baud rate if the operator has selected the option to do so.

About the Program - SETBAUD.ASM

This program is written in 8080 assembly language using the standard format recognized by the CP/M Assembler (ASM.COM) and Microsoft's Macro-80 Assembler (M80.COM) and others. It uses system calls to the CP/M operating system to print messages to the console terminal, read input from the console keyboard, and send the Escape sequence to the terminal to update its baud rate if required. It can update any of the four serial I/O ports of the H89 computer: Console - 350Q (0E8H), Printer - 340Q (0E0H), TTY - 320Q (0D0H), and Modem - 330Q (0D8H). If the operator selects to modify the baud rate of the console, the appropriate Escape sequence is sent to the terminal to modify the baud rate of the terminal before the baud rate of the CPU is modified.

I have used comment statements extensively in the program to help others to understand its operation. The first part of the program saves the stack pointer and sets up a new stack so that we can return directly to CP/M after execution without re- booting. Then we print the menu of selections and prompt the operator to enter the selections. The dialogue between the computer and operator looks like this:

```
I/O Port Baud Rate Programmer Version 1.0
by Rick Swenton      18-Oct-83

Which I/O Device do you wish to set?

1 - CONSOLE          2 - PRINTER
PORT 0E8H            PORT 0E0H

3 - TTY              4 - MODEM
PORT 0D0H            PORT 0D8H
```

(Operator now makes selection:)

```
        1
```

```
        MOV     A,B             ;GET THE SELECTION
        INR     A
;
FINDEV: DAD     D               ;RAISE H,L BY 9
        DCR     A               ;DECREMENT A
        JNZ     FINDEV          ;LOOP UNTIL DEVICE MESSAGE FOUND
        SHLD    DMESTM          ;SAVE DEV MESSAGE LOCATION
;
SENDB:  LXI     D,BAUDM         ;D,E POINT TO BAUD RATE SELECTION MESSAGE
        CALL    OUTS            ;PRINT MESSAHE
        CALL    INC             ;GET THE SELECTION FROM THE OPERATOR
        SUI     'A'             ;SUBTRACT ASCII BIAS
        JC      SENDB           ;BAD ENTRY
        CPI     'I'-'A'+1       ;GREATER THAN I?
        JNC     SENDB
        MOV     B,A             ;TEMP STORAGE
        INR     A
        LXI     H,BRDVT-2       ;POINT TO BAUD RATE DIVISOR TABLE-2
;
FINDD:  INX     H               ;RAISE H,L TWICE
        INX     H
        DCR     A               ;DECREMENT A
        JNZ     FINDD           ;LOOP UNTIL A=0 THEN DIVISOR FOUND
        MOV     E,M             ;GET THE BAUD RATE DIVISOR FROM
        INX     H               ;MEMORY AND PLACE IN
        MOV     D,M             ;REGISTERS D,E
        XCHG                    ;SWAP H,L - D,E
        SHLD    BRDVTM          ;SAVE BAUD RATE DIVISOR
        LXI     H,ESEQT-4       ;SET H,L TO ESCAPE SEQUENCE TABLE-4
        LXI     D,4
        MOV     A,B             ;GET SELECTION
        INR     A               ;INCREMENT
;
FINDE:  DAD     D               ;RAISE H,L BY 4
        DCR     A               ;DECREMENT A
        JNZ     FINDE           ;LOOP UNTIL ESCAPE SEQUENCE FOUND
        SHLD    ESEQTM          ;SAVE LOC OF ESACPE SEQUENCE
        LXI     H,BMEST-6       ;SET H,L TO BAUD RATE MESSAGE TABLE-6
        LXI     D,6
        MOV     A,B             ;GET SELECTION
        INR     A               ;INCREMENT A
;
FINDP:  DAD     D               ;RAISE H,L BY 8
        DCR     A               ;DECREMENT A
        JNZ     FINDP           ;LOOP UNTIL PRINT MESSAGE FOUND
        SHLD    BMESTM          ;SAVE LOCATION OF MESSAGE
;
;
SETUP:  CALL    CRLF            ;NEW LINE
        LHLD    DMESTM          ;GET LOC OF DEVICE MESSAGE
        XCHG                    ;SWAP H,L - D,E
        CALL    OUTS            ;PRINT IT
        LXI     D,BMESS         ;POINT TO BAUD RATE MESSAGE
        CALL    OUTS            ;PRINT THE MESSAGE
        LHLD    BMESTM          ;GET LOC OF BAUD RATE MESSAGE
        XCHG                    ;SWAP H,L - D,E
        CALL    OUTS            ;FINISH THE MESSAGE BY PRINTING NEW BAUD RATE
        LHLD    ESEQTM          ;GET LOC OF ESCAPE SEQUENCE
        XCHG                    ;SWAP H,L - D,E
        LHLD    BRDVTM          ;GET BAUD RATE DIVISOR INTO H,L
        LDA     BPTM            ;GET THE BASE PORT VALUE INTO A
        CPI     CRT             ;SEE IF THE BASE PORT IS CONSOLE
        CZ      OUTS            ;SEND H-19 ESCAPE SEQUENCE IF CONSOLE
        LXI     H,200H          ;PLACE TIMING CONSTANT IN H,L
        CALL    DELAY           ;AND WAIT FOR THE DUST TO SETTLE
        LHLD    BRDVTM          ;GET BAUD RATE DIVISOR
        LDA     BPTM            ;GET THE BASE PORT VALUE INTO A
        CALL    INITACE         ;UPDATE THE ACE
        CALL    CRLF            ;NEW LINE
        EI                      ;ENABLE INTERRUPTS
        LHLD    OLDSP           ;GET THE OLD STACK POINTER
        SPHL                    ;RESTORE IT
        RET                     ;DONE, RETURN TO CCP
;
;
;       CONSOLE INPUT/OUTPUT ROUTINES PERFORMED
;       THROUGH CP/M SYSTEM CALLS
;
;       READ A CHARACTER FROM THE CONSOLE
```

```
Select the new Baud Rate:

A - 110  D - 1200    G - 4800
B - 300  E - 2400    H - 9600
C - 600  F - 3600    I - 19200

(Operator now makes selection:)

        I

Console Baud Rate now set to 19200
```

The program printed all of the above dialogue at the old baud rate. Then the program determined which baud rate and port were selected and sent "Escape rM" to the terminal. It then called the INITACE subroutine to initialize the port. The H89 and H89A use an Asynchronous Communications Element IC to do the parallel to serial conversion. The INITACE subroutine is very similar to the routines used in Heath's SETLP.COM and the BIOS for CP/M to program or reprogram the ACE. I have increased a delay so that the program would run at both 2 and 4 MHz without any changes. After the ACE is programmed, the program will return to the CP/M operating system by restoring the old stack and returning to the Console Command Processor.

There is one precaution when typing this program. There are several "look-up" tables at the end of the program which have a fixed number of bytes in the tables. These have the labels:

DMEST (Device Message Table)
BMEST(Baud rate Message Table)
BRDVT(Baud Rate Divisor Message Table)
ESEQT(Escape Sequence Message Table)
BPT(Base Port Table)

These tables MUST by typed exactly as shown or the program will run improperly or not at all.

This program can be modified for specialized applications. However, it will not run on the Heath H-8 computer because of the different port assignments and the different IC's used. (I believe the H-8 uses the 8251 instead of the 8250 IC. The differences are drastic.)

To copy this program on disk, you need a text editor. Type the program carefully and name the program: SETBAUD.ASM. Then invoke the CP/M Assembler by typing: ASM SETBAUD.abc where "a" is the disk drive which has the source file (SETBAUD.ASM), "b" is the disk drive where the assembler will save the .HEX file, and "c" is the disk drive where the assembler will save the Print Listing (use a "Z" here to suppress the Print Listing). After assembly, use the LOAD.COM program to convert the .HEX file into a .COM file. Type: LOAD n:SETBAUD where "n" is

```
;
INC:    PUSH    H               ;SAVE THE ENVIRONMENT
        PUSH    D
        PUSH    B
        MVI     C,RCONF         ;CP/M FUNCTION NUMBER IN C
        CALL    BDOS            ;BDOS RETURNS THE CONSOLE CHAR IN A
        POP     B
        POP     D
        POP     H               ;RESTORE THE ENVIRONMENT
        RET
;
;
;       SEND CARRIAGE RETURN AND TWO LINEFEEDS TO THE CONSOLE
;
CRLF:   LXI     D,CRLFT         ;SET D,E TO CR,LF,LF MESSAGE
;
;
;       PRINT A CHARACTER STRING ON THE CONSOLE
;
; PRE-REQUISITE:  D,E HAS THE LOCATION OF THE STRING
;
OUTS:   PUSH    H               ;SAVE THE ENVIRONMENT
        PUSH    D
        PUSH    B
        MVI     C,PSTRF         ;CP/M FUNCTION NUMBER IN C
        CALL    BDOS            ;BDOS PRINTS THE STRING UNTIL '$'
        POP     B
        POP     D
        POP     H               ;RESTORE THE ENVIRONMENT
        RET
;
;
;       INITACE - INITIALIZE AN 8250 ACE
;
; PREREQUISITES:   H,L CONTAINS THE BAUD RATE DIVISOR WORD
;                  A CONTAINS THE BASE PORT
;
INITACE:MOV     B,A             ;SAVE BASE PORT VALUE IN B
        XCHG                    ;SAVE BAUD RATE DIVISOR IN D,E
        LXI     H,OUTACE+1      ;SET H,L TO LOCATION TO PLACE PORT VAL
        MVI     A,3             ;SET BAUD RATE ACCESS BIT ON BASE+3 PORT
        ADD     B               ;GET ACTUAL PORT
        MOV     C,A             ;SAVE IN C FOR LATER
        MOV     M,A             ;AND MODIFY THE PORT VALUE WHICH
                                ;FOLLOWS THE OUT INSTRUCTION
        MVI     A,083H          ;SET DIVISOR LATCH ACCESS BIT
        CALL    OUTACE          ;SEND IT TO THE ACE
        INR     M               ;RAISE THE BASE PORT VALUE BY ONE
                                ;TO MODIFY THE MODEM CONTROL REGISTER
        MVI     A,00FH          ;AND SET DSR & CTS HIGH FOR DIABLO
        CALL    OUTACE          ;AND OTHER PRINTERS WHICH REQUIRE THEM
        MOV     M,B             ;SET PORT TO LEAST SIG BYTE
        MOV     A,E             ;
        CALL    OUTACE          ;SEND IT TO THE ACE
        MOV     A,D             ;SET PORT TO MOST SIG BYTE
        ANI     00FH            ;AND TURN OFF CONTROL FLAGS
        INR     M               ;INCREMENT THE PORT NUMBER
        CALL    OUTACE          ;SEND IT TO THE ACE
        MOV     M,C             ;RESET PORT TO DIVISOR LATCH ACCESS
        CPI     B110 SHR 8      ;IF SET FOR GREATER THAN 110 BAUD
        MVI     A,3             ;THEN SET NO PARITY, 8 BIT WORDS, 1 STOP BIT
        JC      ACE1            ;IF 110 THEN JUMP TO ACE1
        ORI     4               ;IF NOT 110 THEN SET TWO STOP BITS
ACE1:   CALL    OUTACE          ;SEND IT TO THE ACE
        DCR     M               ;NOW SET PORT FOR INTERRUPT CONTROL
        DCR     M               ;BY DECREMENTING THE PORT NUMBER TWICE
        XRA     A               ;DISABLE ALL DEVICE INTERRUPTS
        CALL    OUTACE          ;DISABLE INTERRUPTS
;
;       DELAY FOR APPROXIMATELY FOUR CHARACTER TIMES
;       (DELAY WILL BE TWO CHARACTER TIMES IF RUNNING AT 4 MHZ)
;
        XCHG                    ;PUT BAUD RATE DIVISOR IN H,L
DELAY:  DAD     H               ;MULTIPLY BY 32 TO GET DELAY
        DAD     H               ;
        DAD     H               ;
        DAD     H               ;
        DAD     H               ;
        DAD     H               ;
;
LOOP:   DCX     H               ;DECREMENT THE COUNT
```

the disk drive which has the file SETBAUD.HEX. The loader will create the file: SETBAUD.COM on specified disk drive. SETBAUD.COM can now be run like any other .COM file.

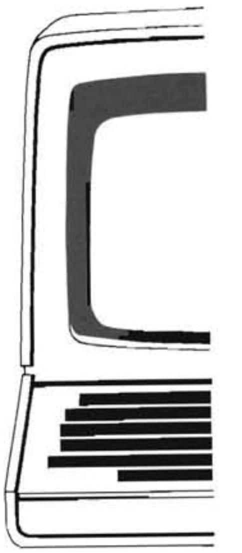
Have fun!

About the Author:

Rick Swenton *is an Equipment Service Representative with Eastman Kodak Company in Hartford, Ct. His work centers around Computer Output Microfilmers and Computerized Microfilm Retrieval Systems. His first computer was a home built 8080 system with over 8K of machine code entered through an EPROM Programmer. He will NEVER do that again! That's why he bought the H89! Rick is also Amateur Radio operator WA1LMV.*

```
        MOV     A,L             ;GET THE LEAST SIG BYTE
        ORA     H               ;OR IN MOST SIG BYTE
        JNZ     LOOP            ;LOOP BACK IF NON-ZERO RESULT
        RET                     ;
;
;       SELF-MODIFYING OUT INSTRUCTION USED BY INITACE
;
OUTACE: OUT     000H            ;THE PORT VALUE IS ALTERED
        RET                     ;BY PLACING THE VALUE IN
                                ;LOCATION OUTACE+1
;
;
;       FIRST PART OF CONFIRMATION MESSAGE
;       SELECTED DEVICE MESSAGE TABLE
;
; ONE OF THESE MESSAGES ARE SENT TO THE CONSOLE
; TO CONFIRM THE SELECTED I/O DEVICE
; THERE MUST BE NINE ASCII CHARACTERS IN EACH MESSAGE
; LEADING CHARACTERS ARE ASCII NULS
; TRAILING CHARACTER IS ASCII SPACE
;
DMEST:  DB      'Console $'
        DB      'Printer $'
        DB      0,0,0,0,'TTY $'
        DB      0,0,'Modem $'
;
;       SECOND PART OF CONFIRMATION MESSAGE
;
BMESS:  DB      'Baud Rate now set to $'
;
;
;       THIRD PART OF CONFIRMATION MESSAGE
;       BAUD RATE MESSAGE TABLE
;
; ONE OF THESE MESSAGES ARE SENT TO THE CONSOLE
; FOLLOWING THE ABOVE PRINT MESSAGE
; TO CONFIRM THE BAUD RATE SELECTION
; THERE MUST BE SIX ASCII CHARACTERS IN EACH MESSAGE
; INCLUDING TRAILING SPACES
;
BMEST:  DB      '110  $'
        DB      '300  $'
        DB      '600  $'
        DB      '1200 $'
        DB      '2400 $'
        DB      '3600 $'
        DB      '4800 $'
        DB      '9600 $'
        DB      '19200$'
;
;
;       BAUD RATE DIVISOR TABLE FOR 8250 ACE
;
BRDVT:  DW      0417H           ;110 BAUD
        DW      0180H           ;300 BAUD
        DW      00C0H           ;600 BAUD
        DW      0060H           ;1200 BAUD
        DW      0030H           ;2400 BAUD
        DW      0024H           ;3600 BAUD
        DW      0018H           ;4800 BAUD
        DW      000CH           ;9600 BAUD
        DW      0006H           ;19200 BAUD
;
;
;       BAUD RATE ESCAPE SEQUENCE TABLE FOR H-19
;
; IF THE OPERATOR HAS SELECTED TO MODIFY THE BAUD RATE OF
; THE CONSOLE, THEN ONE OF THE FOLLOWING ESCAPE SEQUENCES
; WILL BE SENT TO THE H-19 TERMINAL
; THERE MUST BE FOUR ASCII CHARACTERS IN EACH MESSAGE
;
ESEQT:  DB      ESC,'rA$'       ;110  BAUD
        DB      ESC,'rC$'       ;300  BAUD
        DB      ESC,'rD$'       ;600  BAUD
        DB      ESC,'rE$'       ;1200 BAUD
        DB      ESC,'rH$'       ;2400 BAUD
        DB      ESC,'rI$'       ;3600 BAUD
        DB      ESC,'rJ$'       ;4800 BAUD
        DB      ESC,'rL$'       ;9600 BAUD
        DB      ESC,'rM$'       ;19200 BAUD
;
;
;       I/O BASE PORT ASSIGNMENT TABLE
;
BPT:    DB      0E8H            ;CONSOLE PORT 350Q
        DB      0E0H            ;PRINTER PORT 340Q
        DB      0D0H            ;TTY     PORT 320Q
        DB      0D8H            ;MODEM   PORT 330Q
;
;
;       SIGN-ON MESSAGE
;
; THIS MESSAGE IS PRINTED AT THE CONSOLE TO PROMPT
; THE OPERATOR TO SELECT THE DESIRED I/O DEVICE
; TO BE MODIFIED
;
DEVM:   DB      ESC,'EI/O Port Baud Rate Programmer Version 1.0',CR,LF
        DB          '      by Rick Swenton  18-OCT-83',CR,LF,LF
        DB          'Which I/O Device do you wish to set?',CR,LF,LF
        DB          '1 - CONSOLE     2 - PRINTER      3 - TTY     4 - MODEM'
        DB          CR,LF
        DB          '  PORT 0E8H       PORT 0E0H       PORT 0D0H       PORT 0D8H'
        DB          CR,LF,LF,7,'$'
```

```
;
;       BAUD RATE PRINT MESSAGE
;
; THIS MESSAGE IS PRINTED AT THE CONSOLE TO PROMPT
; THE OPERATOR TO SELECT THE DESIRED BAUD RATE
;
BAUDM:  DB      CR,LF,LF,'Select the new Baud Rate: ',CR,LF,LF
        DB      'A - 110                D - 1200            G - 4800',CR,LF
        DB      'B - 300                E - 2400            H - 9600',CR,LF
        DB      'C - 600                F - 3600            I - 19200',CR,LF
        DB      CR,LF,LF,7,'$'
;
;
CRLFT:  DB      CR,LF,LF,'$'
;
;
        DS      15                      ;SET UP 16 BYTE LOCAL STACK
STACK:  DS      1
;
OLDSP:  DS      2                       ;OLD STACK POINTER SAVED HERE
;
;       STORAGE LOACTIONS
;
BPTM:   DS      1                       ;BASE PORT VALUE (BYTE)
DMESTM: DS      2                       ;DEVICE MESSAGE TABLE POINTER (WORD)
BRDVTM: DS      2                       ;BAUD RATE DIVISOR TABLE POINTER (WORD)
ESEQTM: DS      2                       ;ESCAPE SEQUENCE TABLE POINTER   (WORD)
BMESTM: DS      2                       ;BAUD RATE MESSAGE TABLE POINTER (WORD)
;
;
        END     START
```



H89/Z90's CAN NOW DEAL WITH A FULL DECK!

THE ORIGINAL ALL-IN-ONE ACCESSORY BUS EXPANDER.

MH89+3 doubles expansion capacity. Allow for 6 right-hand type cards instead of the usual 3. Room at last to run those neat accessory boards you've seen advertised!

Piggyback motherboard installs internally with a screwdriver in just minutes – with no modifications! 3 slots exactly duplicate the originals. The 3 added slots occupy unused addresses and eliminate previous conflicts. 100% compatible with all accessory boards!

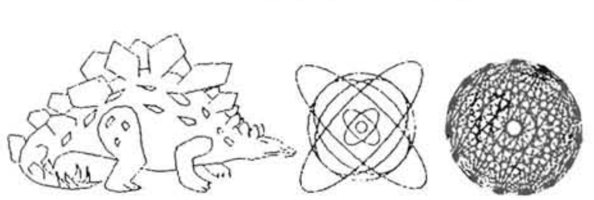
No overheating problems! Simple design draws little power. Leaves plenty of overhead for the minimal load of most accessories. Full technical information provided.

The best news about this "No-hassle" design is the price — **ONLY $150.** About 1/3 the price of other solutions!

Price includes assembled and tested MH89+3 expander, complete instructions and one (1) year warranty. CA residents add 6% tax. USA include $5 shipping. Foreign add $10. Telephone and COD orders accepted.

mako data products
1441-B N. RED GUM, ANAHEIM, CA 92806
PHONE (714) 632-8583

EPSON Printers CAN Do Graphics!



TRUE GRAPHICS FOR EPSON PRINTERS

• **MXGRAPH** will turn your EPSON printer into a graphics plotter for personal or business needs! Make your EPSON plot an unlimited number of artistic sketches, geometric designs, and business graphs, which would not be possible without MXGRAPH. MXGRAPH runs under CP/M or HDOS operating systems ONLY on the H/Z-89 and H/Z-90 computer systems. Requires a serial EPSON printer equipped with bit plot graphics. Specify HDOS or CP/M, hard or soft sectored 5.25" disk. **MXGRAPH is ONLY ~~$59.95~~ $49.97!**

LETTER QUALITY PRINTING FOR THE EPSON

• **MXPRINT** generates almost **LETTER-QUALITY** printing on an EPSON printer equipped with bit plot graphics! MXPRINT sends your files to the printer to be printed at 10 cpi by 6 lpi. Each character is plotted by a 9x24 dot matrix. MXPRINT includes MXFONT for generating your own character fonts. MXPRINT requires **CP/M** Version 2.2.03 or later for operation on an H-8, H/Z-89, or H/Z-90; or CP/M-85 for operation on an H/Z-100. Specify hard or soft sectored 5.25" disk. **MXPRINT is ONLY $19.97!**

WORDSTAR ENHANCER

• **NOVA The WordStar™ Enhancer!** Add full use of ALL your keypad AND special function keys to WordStar. Greatly increases speed and ease of editing text. 25th line prompts you. NOVA REQUIRES HEATH/ZENITH DISTRIBUTED WORDSTAR. NOVA requires **CP/M** Version 2.2.03 or later for operation on an H-8, H/Z-89, or H/Z-90; or CP/M-85 for operation on an H/Z-100. Specify hard or soft sectored 5.25" disk. **NOVA is ONLY ~~$39.95~~ $24.97**

Z-100 ELECTRONIC FILING CABINET

• **EFC - The Electronic Filing Cabinet** is a screen-oriented database management package. EFC is human engineered to allow ANYONE to store and retrieve any and all data as if you were indeed pulling files from a cabinet. Use EFC as a **PERSONAL FILING SYSTEM** for all your personal needs from financial accounts data storage to the household grocery list. Available for **Z-DOS** on the H/Z-100. **Electrinic Filing Cabinet is ONLY $39.97!**

INTERLACED GRAPHICS PACKAGE

• **ILG GRAPHICS - Resident Interlaced Graphics Package!** ILG is a resident 8088 assembly language interlaced graphics package. In interlaced mode pixel resolution is 640x by 480y. Includes a 256 interlaced character font set. Once loaded, ILG remains resident in system memory for easy interface by any high level language or assembly language module. ILG includes many examples to interface to the various languages as well as an interlaced screen-dump to an EPSON or Diablo printer. **ILG interfaces directly with all the MicroSoft compilers (PASCAL, FORTRAN, COBOL, ZBASIC), assembler (MASM-86) and the Z-BASIC interpreter.** 64K video ram chips are required for operation in interlaced mode. Runs under **Z-DOS** on the H/Z-100. **ILG GRAPHICS is ONLY $39.97!**

PRINTING THAT ONLY A DIABLO CAN DO

• **LETTERIT** generates poster quality lettering on any DIABLO printer! Prints proportional about 3 char per inch by 2 lines per inch (Nx24 matrix). Includes DBFONT for generating your own lettering. LETTERIT requires **CP/M** Version 2.2.03 or later for operation on an H-8, H/Z-89, or H/Z-90; or **CP/M-85** for operation on an H/Z-100. Specify hard or soft sectored 5.25" disk. **LETTERIT is ONLY $19.97!**

MICRO INNOVATIONS

2455 Sylvania Avenue
Toledo, Ohio 43613

VISA and MASTERCARD Orders Welcome!

PHONE ORDERS CALL: (419) 471-1245

Vectored from 8

Question: Is it HDOS or HDOS MBASIC which will not let me pass the high order bit? I don't believe that the hardware is at fault because the serial board can be configured for 8 bit operation. Do you have a solution or any patches that can solve this problem? I am getting tired of beating my head against the wall; bad for the old noodle.

Christian L. Wilson
US NAS Bermuda
P. O. Box 1476
FPO N.Y., NY 09560

Corrections To MBASIC Electronic Worksheet in Issue 44

Dear HUG,

Just a couple of lines to tell you that there were a couple of typos in the MBASIC Electronic Worksheet which appeared in the September 1983 Issue 44.

Line 130 of the program should be as follows to ensure enough string space:

```
130 CLEAR (10000):DEFINT B-Z
    ^^^^^^ ^^^^^^^^
```

Line 3720 should read as follows to print the worksheet properly if you are using HDOS MBASIC:

```
3720 PRINT#1,SPACE$(CW(X1)-LEN(B$(Y1,X1)));B$(Y1,X1)
                                  ^  ^
```

I don't see the purpose of line 2950. Could you please enlighten me?

And just another line letting you know that I think you are doing a great job. Thanks for keeping us up with the times!!

Christian L. Wilson
US NAS Bermuda
P. O. Box 1476
FPO N.Y., NY 09560

HUG Payroll Package

Dear HUG,

We recently placed the HUG Payroll Package (CP/M #885-1218) in service and are very pleased with its performance and ease of use by inexperienced operators.

Since we pay employee expense accounts on our paychecks, it appeared at first that we would have to change to separate expense checks. However, by labeling one of the voluntary deduction columns "-Expense" and by entering the employee's expense amount each pay period as a negative voluntary deduction, we were able to generate combined checks without any program modification.

The CP/M version we purchased had the fix from page 27 of REMark #39 installed. However, the FICA module of the PAYCAL.BAS program still didn't work properly.

For the pay period in which an employee's gross year-to-date salary passes the FICA maximum, the program calculates FICA tax on the pay which is in excess of the FICA maximum instead of on the amount of pay up to the maximum.

Line 470 is unchanged. Replacing line 475 with the following causes FICA to be calculated correctly.

```
470 IF CVD(Y$(1))>T2 THEN P3=0:GOTO 480:' Max FICA
475 IF P1+CVD(Y$(1))=>T2 THEN P3=(T2-CVD(Y$(1)))*T1:P3=INT(P3*100+.5)/100
```

Prior to year end, we intend to implement the following improvements to the payroll package:

1. Show employer's share of FICA due, and allow for different employee and employer FICA rates in 1984.

2. Modify the Federal tax payments due, so that the warning is triggered by the combined FICA and Fed. Inc. Tax due, per IRS requirements.

3. Allow multiple trigger amounts for combined Federal taxes due, showing the days allowed for payment per IRS requirements.

We'll provide listings and instructions to HUG for these modifications after testing.

Thanks for a solid piece of software.

Marty Heerschap
Jim Howerzyl
1700 Centennial Drive
Louisville, CO 80027

Enhancement To Tour-700

Dear HUG,

A few months ago I bought a copy of Tour-700, a nice little simulation of an early 20th Century card game developed in Europe. It's particularly good for play by fairly small children (those who have just learned to read) because it doesn't require continuous action like most arcade-type games. The game is distributed by Newline Software, P. O. Box 402, Littleton, MA 01460. Its only flaw is that as the end of the game approaches, the pace becomes maddeningly slow as the random number generator takes increasingly long to find a card to draw out of the 101- deck matrix. Fortunately, the game is written in BASIC and it's fairly easy to fix the problem with the following additional lines of code:

```
692 IF T>5 THEN 700  '  T = Cards still in deck
693 X=8  ' A(8) just precedes "ROLL" cards in matrix
694 X = X+1: IF X>19 THEN X=1 ' 19 kinds of cards in deck
696 IF A(X)=0 THEN 694 ' try again if none of these left
```

This has the effect of bypassing the random number generator whenever there are five or fewer cards remaining in the deck (when things really slow down). Line 693 is optional; it sets the matrix index so that if any "ROLL" cards remain, they will be dealt first. Then it looks for "SPEED" cards. These additional lines simply step through the deck matrix in rising numerical order rather than having the random number generator keep looking into empty elements.

Without this fix, I have seen this game take a full hour to deal one card. I don't believe that the fix interferes with the logic of the game in any other respect.

Lewis M. Phelps
50 Fifth Avenue
San Francisco, CA 94118

Unhappy Peachtree User

Dear HUG,

Readers of the fine article by H. W. Bauman concerning PeachText 5000 (issues 46 and 47) may wish to consider my experience before purchasing this software. The software is not supported by Heath, but by Peachtree, which is unfortunate. Being a novice with Z-DOS at the time, I made a stupid error in attempting to back up the PeachText 5000 system. The result was destroying the contents of the first disk prior to back up. I asked Peachtree to restore the contents of my disk (we license software, not buy it). They wrote saying a $25.00 charge is required for "shipping and handling of the disk". (I paid 88 cents postage and used a 20 cent envelope to send it to them.) My letter claiming this to be unrealistic for home users and requesting either a reasonable charge or the name of the company president was answered after 6 weeks with neither.

One might have second thoughts about dealing with a company which makes outrageous charges to customers whom they have over a barrel. PeachText 5000 will certainly be my last purchase of an MSA product!

Peter N. Bartram
3294 Chadbourne Rd.
Shaker Heights, OH 44120

Remapping Keys and Graphics On the Z-100

Dear HUG,

The ability to remap keys and graphics is one of the great virtues of the Z-100. It's a pity it is so poorly documented in the Z-100, Z-DOS, and CP/M-85 manuals.

Thanks to Marc Aagenas of ZDS Software Consultation for the illuminating article "A Z-100 Font Editor" (Remark No. 42) and the excellent program "FONTED.BAS", which works as advertised!

A hint to those who, like me, expect to make a typing error or two in entering such a long and complex program:

DO NOT enter line 20 (20 ON ERROR GOTO 2790) until the program has been thoroughly debugged and freed of the inevitable syntax errors!

The error trapping routine in line 20 will (except for a "File Not Found" error which it is designed to trap) cause an immediate jump back to the main menu without any visual indication of the existence of an error, much less an indication of what it was, making debugging difficult if not impossible.

John F. Smith
Ford Aerospace
300 Welsh Road
Horsham, PA 19044

Tip For CP/M Users' Of 4MHz Mod

Dear HUG,

Here is a tip for CP/M users' of Pat Swayne's 4 MHz modification. Ever since I received REMark Issue 34, I have used Pat's modification. And ever since then, I have been plagued with select errors on B: after having once used Drive B.

When applying the modification to CP/M 2.2.04, I discovered that something has been added just before RDYH37D:, in the form of a Debounce delay. The constant is FDHDD which is equal to 20. So I changed "MVI A,FDHDD" to "MVI A,40" and presto, no more problem with can't select B: errors.

Robert F. Hassard
3566 Tice Creek Drive, #4
Walnut Creek, CA 94595

A New Pascal Compiler

Dear HUG,

While leafing through my October issue of Microsystem, I ran across an advertisement for another "inexpensive" Pascal Compiler available to Heath Users'. I am surprised that it has not been mentioned before in "REMark", considering what it has to offer the "home type software hacker".

Released by Borland International, 4807 Scotts Valley Drive, Scotts Valley, CA 95066 and selling for a mere $49.95, it runs under CP/M 2 and is available for all Heath disk formats. Its nomenclature..."Turbo" Pascal, and Borland claims it is fast.

Having worked with Polybyte's "Lucidata" Pascal these past couple of years, my biggest gripe was its compilation speed, even using "RUNCOM". Having worked with "Turbo" for the past two months, I can personally attest to its being faassstttt! Programs under Lucidata which take minutes to compile, compile with "Turbo" in seconds. A source program of 400 lines takes approximately "10 seconds".

The speed of compilations is not its only noteworthy feature. It supports an in-line editor. During compilation, the compiler invokes the editor pointing to a source code error. Borland's editor claims compatibility with WordStar although not all WordStar features are implemented.

The "goodies" standard with "Turbo" are too numerous to mention, but if there are people out there working with "the other brand" that are envious of UCSD string functions and procedures, or are tired of adapting "library" functions to individual source programs, "Turbo" has a lot to offer.

There are a number of standard Pascal features not supported. "GET" and "PUT". No "Page". No "packed" variables. Possibly its one weakness, though not for the majority of us hackers, procedures and function can not be passed as parameters.

The supplied disk has a "Terminal Installation" program which lists a Zenith H/Z-19 terminal as already installed. My copy had none of the proper codes installed. After a little difficulty and a phone call, all was set right and hopefully Borland now has the Zenith codes installed on future disks.

Quite frankly, with their speed of compilation and in-line editor, Borland "Turbo" Pascal offers a package for software development almost as convenient as BASIC, which will also compile to an absolute (COM) file.

Donald E. Risher
908 Charlotte Place
Charleston, WV 25314

Corrections To An Assembly Language Program In Issue #47

Dear HUG,

In the December 1983 issue of REMark (Issue #47, page 46), Tom Colman presented a small assembly language program to utilize the "transmit page" function on H/Z-19 and H/Z-89 terminals. I have a few comments which may be of interest to him and others who use this feature.

Mr. Colman's routine will fail if the screen contains graphics or reverse video characters. The reason for this is that he has assumed that the transmit page function will always cause exactly 1920 characters (24 lines by 80 characters per line) to be transmitted. In fact, if the screen contains graphics or reverse characters, the transmit page function will transmit the appropriate ESCape sequences and the final total will be more than 1920. (The sequence is always terminated with a Carriage Return.) One way to handle this would be to increase the buffer size, but we would have to make it very big to be absolutely safe.

A better way is to note that there are exactly 256 printable characters which can be displayed on the H/Z-19/H/Z-89 screen (95 printable ASCII characters plus 33 graphics characters, each in normal or reverse video). In order to take advantage of this fact, one can add a little code to Tom's program to remap the graphics and reverse characters into unused bit patterns in the 1920 byte buffer. I recommend remapping the first 32 graphics characters into the control character space (0 to 31 decimal). For reverse video characters, one can then simply set the 8th bit (i.e. add 128 to the character representation). With this scheme you are guaranteed that one screen's worth of characters will always fit into a 1920 byte buffer space. The price you pay is a little extra computation time in remapping the character representation.

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

A Repeat For MUSIC

Dear HUG,

Interest seems to continue in Frank Clark's MUSIC program ("Changing Your Tone", REMark #38). Those who have been able to make the program run and have come to understand how to write the SCORE code have been rewarded with some very interesting results. Our associate, Bob Horn, managed to encode an extensive score from Startrek II!

Since some considerable labor is required to write the score code, we found a way to cause selected phrases and periods to repeat. This code will either repeat a period once and continue, or will repeat forever if you can stand it. The trick is to embed controlling flag characters in the score that MUSIC is blind to. We selected the following ASCII characters:

'X', to be placed at the point that the repeat returns to;

'Y', to be placed at the end of a period that is to repeat only once;

'Z', to be placed at the end of a period that is to repeat forever.

These characters can be braced in commas for readability but it is not required.

To make the repeat work, a very small amount of code needs to be added to the ASM source code.

(1) In the "MAIN LOOP", find the label "LOOP". On the next instruction (MOV A,M) add a new label, "LOOPB".

(2) 3 lines down, insert the following code, between the instructions JZ QUIT and PUSH H:

```
;
        CPI     'X'      ;is it a repeat marker?
        JZ      LOOPB    ;ignore 'X' as we go by
        CPI     'Y'      ;is it one repeat?
        JZ      REPEAT   ;yes, go do it
        CPI     'Z'      ;repeat forever?
        JZ      REPEAT1  ;yes, 'fraid so
;
;   BACK TO THE ORIGINAL CODE
;
LOOPR   PUSH    H        ;note the new label here
;
```

(3) Now, just ahead of the label "CMD$GOT", and below the instruction "JMP OCT$CHK", insert the following code:

```
;
REPEAT  DCX     H        ;point back to 'Y'
        MVI     M,' '    ;replace 'Y' with space to
;                        ;prevent second repeat
REPEAT1 DCX     H        ;point back one
        MOV     A,M      ;get character
        CPI     'X'      ;back to start yet?
        JNZ     REPEAT1  ;no, loop 'til we get there
        INX     H        ;point to first after 'X'
        MOV     A,M      ;get it
        INX     H        ;point to next
        JMP     LOOPR    ;back to where we jumped out
```

That's all you need to do except assemble and run it. Perhaps your holiday music next year will be as strange as ours was this year.

Charles Horn
Horn Engineering Associates
1714 Patricia Lane
Garland, TX 75042

David Warnick Adds Some Corrections To His "BASIC Computing" Series

Dear HUG,

Now that I've been writing for a few months, several readers and I have discovered some inevitable errors in past articles. Here are the corrections.

REMark #42, page 57, correct 2 lines.

```
5131 PRINT E1$;CHR$(Y+1);CHR$(X);"i";E9$;CHR$(95);
     E10$;"i"
5160 PRINT E1$;CHR$(Y);CHR$(X);E9$;"r";E10$;"p";E9$;
     CHR$(95);E10$
```

REMark #46; page 27, missing parenthesis.

```
5020 P2=INT(25*SIN((360/23)*(P/57.3)))+45
5040 E2=INT(25*SIN((360/28)*(P.57.3)))+45
5060 I2=INT(25*SIN((360/33)*(P/57.3)))+45
```

I'm sorry if these errors caused any grief for the readers. They were my typos as the BASIC Computing is submitted on disk.

For those readers who hate to type but would like to run my programs, I am offering two disks:

Disk #1: From REMark Issues 42, 43, 44, 45. WORD.BAS - BASIC program you can work on. WORDS.COM - the same program compiled.

Disk #2: From REMark Issue 46. BIO.BAS - a BASIC biorhythm chart program.

These disks are available in CP/M only, either hard or soft sectored. Each is $5.00 to cover the cost of the disk and shipping. HUG members only, please.

My thanks to all the readers who have written to comment, suggest, or ask for help.

David E. Warnick
RD #2 Box 2484
Spring Grove, PA 17362

Correction To Buggin' HUG Letter "A Typing Error In the Book The C Primer" Printed In December Issue 47

Dear HUG,

Further correction to the C Program is needed than what was printed in the December Issue 47. Here is the letter with the corrections included:

Readers who are following the illustrative examples in the book "The C Primer" by Les Hancock and Morris Krieger should be aware of a typing error in the palindromia exercise on page 145. The statement:

```
    char *strptr2 = strptr;
```

should not include the asterisk. It is necessary to separate the character declaration from the assignment, thus:

```
    char *strptr2;
    strptr2 = strptr;
```

Also, users of the MX-80 printer with CP/M who are cranking up their MBASIC to change the print mode using an LPRINT CHR$(mode) statement will benefit from the following brief assembly language routines.

To set or reset the compressed mode, the basic form is:

```
      org     100h
      mvi     e,mode   ; mode = 15 sets compressed,
;                        18 resets
      mvi     c,5
      call    5
      ret
      end     100h
```

I named the Set Compressed program SC, its reset RC. To enter those modes that employ escapes, such as the emphasized, which requires ESC E, insert the following lines immediately below the org declaration.

```
      mvi     e,27
      mvi     c,5
      call    5
```

To set the emphasized mode, enter 'E' for mode; to return from the emphasized, enter 'F'. I renamed the Set Emphasized program SE; its return RE.

These routines are real time savers as well as reducing disk drive wear and tear (disk on time with these is ever so much shorter!). While the mode values given apply to the Epson printer, the routines should be applicable to any printer that responds to LPRINT instructions for mode changes.

Clem Pepper
3270-96 Caminito East Bluff
La Jolla, CA 92037

Update On "Personal Checks and Balances"

Dear HUG,

I was interested in using Don Thomas' program on Personal Checks and Balances (July Issue #42). As it was headlined for HDOS, I tried it but it would not work for me. I wrote Mr. Thomas and he was good enough to send me the program for HDOS. The one in the magazine should have been headlined for CP/M.

What I want to say now is that if anyone else like me (amateur hobbyist) is using this program in either HDOS or CP/M, they can add this line:

```
515 LINE INPUT "From. ";B3$
```

This will add to the listing so you can note where the deposit came from, which the program lacked.

My thanks to Mr. Thomas for his help. I'd like to add that this is a good program for anyone who has bookkeeping problems.

Carl Saluzzi
P. O. Box 751
Quartzsite, AZ 85346

Beware!!

Dear HUG,

I am writing this letter to pass on some information which may save someone else from making a very costly mistake.

On Aug. 5, I sent my H-8 computer to Trionyx for modification, using the U.S. Postal APO/FPO system. After a month, the computer was finally located in a U.S. Custom's warehouse, IMPOUNDED! And all because it was not clearly marked " U.S. MADE ITEM "! The H-8 remained impounded until a custom's fee of $125.00 was paid. This was on top of the original $65.00 postage, insurance, and registration fees. As you can see, a very expensive mistake.

Please advise your readers who use the US Postal APO/FPO system to clearly mark on the custom's tag " COMPUTER, U.S. MADE ITEM, RETURNED FOR REPAIRS " to avoid U.S. Custom's impound fees as well as worry about their machines.

Sammy G. Hornsby
PSC Box 4256
APO S.F., CA 96519

A Question On Memory Usage of Z-BASIC On An H-100

Dear HUG,

I have a question about the memory usage of Z-BASIC on the H-100 with expanded memory. When I first got my H-100 on line, I noticed that the Z-BASIC signed on with about 62K of memory available. Some time later I added a 256K memory expansion card, the Z-BASIC still signs on with about 62K. After some experimentation, it appeared that it would accept a good deal more strings than originally but it still showed only 62K to start with, and the FRE(0) command never showed more than 62K.

Is the Z-BASIC using extra memory, and if so, how can I tell how much is available? If the Z-BASIC is not using it, is there any way I can use the extra memory from BASIC?

On another subject, I would like to thank all of you at REMark for a

The Software Toolworks presents: MyCalc

CAR PAYMENTS. BANK STATEMENTS! TAXES... ARRGH!
YO, FREDDIE. STILL GLAD YOU FIRED THE ACCOUNTANT?
BUDGETING. ACCOUNTING. TAXES... ARRGH!
THIS IS MYCALC™, FREDDIE. IT'S A SPREADSHEET. AND FOR SIXTY BUCKS IT'LL SOLVE YOUR PROBLEMS.
COMES WITH TEMPLATES FOR IRS FORMS 1040 & SCH. A
TWENTY THREE HELP SCREENS! THIS IS EASY, I MEAN MY BUDGET'S ALMOST DONE.
NICE WORK, FREDDIE. NOW PLOT MY CATNIP FUTURES.
MYCALC* IS $59.95. IT'S AVAILABLE FOR CP/M® AT MOST HEATH/ZENITH DEALERS. WRITE OR CALL FOR A FREE 41 PRODUCT CATALOG AND NEAREST DEALER.
A SOFT BUY IN A HARD WORLD!

*UPDATED VERSION OF ZENCALC

Full featured MyCalc™ with sort, bar graphs, multiple files and more is ideal for financial planning and budgets. It sells for $59.95 from The Software Toolworks, 15233 Ventura Blvd., Suite 1118, Sherman Oaks, CA 91403 (818) 986-4885.

CP/M is a registered trademark of Digital Research great magazine, my thanks also go to all of the other HUGgies out there for some very useful and informative ideas. I would like to send special thanks to Larry Oberholtzer for his article (REMark Issue 44, page 49) on transferring HDOS files to ZDOS, it has saved me a few years of typing.

Joseph Boutilier
25 Glenwood Ave.
Bloomfield, NJ 07003

How To Turn Off The Z-100 Key Click

Dear HUG,

It seems that the Z-100's key click bothers a number of users. I happen to like the audio feedback it provides. For those who don't, there is a very easy way to send the ESCape sequence to turn it off. The secret is to use a REMark statement in an AUTOEXEC file which includes the necessary characters. The sequence will then be sent to the screen, but otherwise ignored by Z-DOS.

```
REM <ESC>x2
```

EDLIN will provide the <ESC> character by typing the F8 key, although it will be displayed as "↑[".

Other ESCape sequences can be sent using this technique, such as for a non-blinking cursor:

```
REM <ESC>x;
REM <ESC>x2<ESC>x;          to do both functions
```

Richard Hole
23560 13 Mile Road
Big Rapids, MI 49307

Headware's CP/M Public Domain Software

Dear HUG,

I appreciate your mention of Headware's CP/M public domain software for Heath Users' in your issue number 47 on page 44.

The response is encouraging, but I see that my wording on the press release was not precise.

Each volume of the original public domain software came in an 8 inch disk and averaged about 130K bytes. When converted to the 5 1/4 inch format, each volume occupies 1, 2, or 3 diskettes, depending on the format. This suggests that our library of public domain volumes (SIG/M volumes 1 through 114 and CP/MUG volumes 1 through 91) occupies 205 diskettes in the double-sided double-density format.

Which is true. The problem came when some people reading the announcement assumed that the entire library was available for the prices mentioned. No doubt they did not realize how many diskettes were involved.

To set the record straight, the prices given are for a SINGLE VOLUME or for the COMPOSITE DIRECTORY which covers the above volumes. Any help you may give to clarify this with your readers would be very much appreciated.

Dick Riggs, Owner
Headware
2865 Akron Street
East Point, GA 30344

More Modifications to MORSE.BAS

Dear HUG,

Reference the Morse.Bas program modification shown in the October 1983 REMark, this program runs fine in CP/M when using MBASIC version 5.21. Some other modifications are necessary for it to send the proper code characters. The following lines should read:

```
380 L$(K)="" THEN GOTO 465
440 FOR T=1 TO N1: NEXT T
445 NEXT U
460 FOR T=1 TO N3: NEXT T
465 NEXT K
```

Line 410 should also be modified as described by Henry Milam in the October 1983 REMark. Changing line 380 as published in the October issue will not allow the machine to read the proper morse characters, but instead only sends "then goto 465" in cw. By making the above changes, my CP/M version of Bob Horn's program works fine. Hopefully this will help someone else with the 5.21 version of MBASIC.

M. Vern Hajek
1924 Packard Ct.
Concord, CA 94521

Feedback On 'C' Series

Dear HUG,

In trying the sample program "More" in Brian Polk's Part III article on "An Introduction To 'C'", I was frustrated to find that it would not work with version 2.0 of the C/80 compiler. It would display the first screen's worth of data, but would not respond to a carriage return to continue the display. The problem is in Mr. Polk's presumption that the getc function reserves channel 0 for the terminal. In the combination debugging and learning effort, I reviewed the compiler documentation and found no substantiation of that assertion. I corrected the program by issuing an fopen at the head of the program for the con: device, thereby obtaining a valid channel number. Other frustrated pupils of C should be made aware of this problem in the sample program.

Richard Sims
185 Freeman Street Apt. 950
Brookline, MA 02146

More On Peachtext

Dear HUG,

The recent articles in REMark (Issues 46 and 47) by H. W. Bauman gave a good description of the Peachtext 5000™ package from Peachtree Software. The functions it performs under ZDOS on the Z-100 are indeed remarkable. The only part of Peachtext not quite up to modern standards is Peachcalc, which is outshined by Multiplan, Lotus 1-2-3, and the like. In fact, Peachtext has some features which are cleverly hidden in the documentation, such as that by selecting 'TW' (stands for Tab by Word), the cursor jumps by words when the TAB key is depressed, and still tabs by numbers when the line is blank.

A very serious deficiency of Peachtext, which should be carefully considered by any potential purchaser, is the inadequate support given to printers. Only the C.Itoh Starwriter, two Diablo, five NEC spinwriter, and five Qume Sprint printers are supported. Amazingly, there is no support for any EPSON printers, certainly the most popular brand, any IBM printers, or for the Okidata 92, which I happen to own. Both Peachtree and Heath Hot Lines advise configuring the printer as 'DRAFT'. In that mode, such features as superscripting, subscripting, and others available for the devices mentioned, cannot be invoked. Fortunately underlining and boldface do work. Of course 'OUT' commands can be used, but this is preposterous when package appears to support the feature directly.

It is also necessary to enter '\BI OFF' to disable the automatic bidirectional printing of text by the printer. Otherwise, as the manual correctly states, every other line will be printed 'SDRAWKCAB'. According to Peachtree, there is no way to change the defaults for BI OFF, TW, or any other commands.

I would like to know if anyone has figured out how to make Peachtree work with the popular printers. Any potential or current users should consider sending Peachtree a letter requesting support for their printer.

Warren S. Hoffman
208 Jackson Boulevard
Deerhurst
Wilmington, DE 19803

Dear HUG,

I would like to reply to Warren S. Hoffman's letter concerning PeachText 5000.

I am sorry that I did not emphasize the need for a letter quality printer in my article to obtain maximum use of the PeachText Word Processor. I did state that is was closely related to MagicWand! Heath has always stated that they recommend the WH-54 letter quality printer for use with MagicWand! In fact, all of the Word Processing Software that I have worked with have the same requirements.

I agree that some of the new spread sheet software packages have a few ways that they can out-perform PeachCalc. That does not mean that PeachCalc is not a good product. Mr. Hoffman is comparing a higher priced software that WILL NOT do what the PeachText 5000 INTERACTIVE SYSTEM will do!

I do hope that someone can help Mr. Hoffman with his OKIDATA 92 printer. I do not know a thing about this printer.

H. W. Bauman
493 Calle Amigo
San Clemente, CA 92672

A Call For Help To All H8 Owners

Dear HUG,

I would like to hear from other H8 owners about hardware related projects.

Ray Poli
2322 Calumet
Flint, MI 48503

Help Needed On Transferring Files Between H/Z-89 & H/Z- 100

Dear HUG,

I liked the recent article by William Adney "Trade Your H/Z-89 For An H/Z-100". I would like to see an article on how easy or hard it is to transfer files and software from an 89 to the Z- 100. I would like to upgrade to a Z-100, but would like to know how compatible the systems are.

In our area, the major employer (VA TECH) has gone with the IBM PC. Many of us are interested in the real Z-100 compatibility with the IBM PC. I would like to see a future article on this topic.

Jim Johnson
1413 Hillcrest Drive
Blacksburg, VA 24060

ZWAR!

Tired of trying to get your latest Pascal compiler to work? Your spreadsheet won't compute? Sick of syntax errors in line 4658? It's time to take a break and indulge your fantasies of global domination with ZWAR.

ZWAR is a *two player* strategy game of world conquest. The screen displays a map of the world and the Soviet and American cities. You must decide what mix of the nine available weapons systems you will build to outwit your enemy. You can destroy his cities with ICBM's, or attempt to capture them with an invasion, thus gaining their production facilities for your own use. Each weapon has its strength: carriers and destroyers for sea superiority, ABM's for defense against missiles, jets for quick response, satellites for spying on enemy activity. The possible strategies are nearly endless.

So if you're getting tired of humdrum computing, grab a friend and try ZWAR. It's your chance to make history.

ZWAR, $19.95 + $1.50 shipping. For H8-H19, H/Z89 or H/Z100 systems (color graphics on the Z100), 56k, and HDOS, CP/M, or CP/M-85. Specify hard or soft sector disk. Free catalog available.

APOGEE SOFTWARE
Box 15124
Savannah, GA 31416
(912) 925-3765

Z-ZAP

Whether you go absolute sector or by filename you'll find Z-ZAP filled with flexibility and power. ASCII, Hex, Oct or Dec display and/or entry with a 4K internal buffer for swapping, moving and editing your data means power and total control over your disk data.

$29.95 Requires ZDOS (except Version 2.0+)

OTHELLO/WARI

Two games for the price of one! If you like a challenge as well as fine graphics, check this one out. Both games were programmed in ZBASIC and take advantage of some very colorful rou - tines, especially the WARI game.

$16.95 Requires ZBASIC, ZDOS (except V. 2.0)
Color (color monitor recommended!)

Westcomp

ALL PRICES INCLUDE SHIPPING/HANDLING. (USA ONLY)
CA RESIDENTS ADD YOUR PROPER SALES TAX!
WE HAVE MOVED INTO OUR OFFICE (FINALLY) AND OUR NEW ADDRESS IS:

517 N. Mountain Ave. Suite 229
Upland, CA 91786

HEATH ZENITH 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. We are not responsible for errors in descriptions or prices.*

The new year is here (obviously gone 2 full months by the time you read this) and the annual lack of new products have come upon us. For some reason (I haven't figured out why, yet), vendors rush to get all their new releases out by the December issue. Then along comes January and February (and sometimes March and April, too) with little or nothing new to report. So if you were wondering why you didn't see this column last month and this one's so short on new stuff, now you know why.

VEGA-BOUND 1
A REAL TIME/3-D SPACE ADVENTURE
FOR ZENITH H/Z-100 COMPUTERS WITH COLOR OR B/W MONITORS
COPYRIGHT © 1983 BY INTERDISCIPLINE INC.

3-D Space Adventure Game for H/Z-100

Interdiscipline, Inc., has announced a real time space adventure game, VEGA-BOUND I, for the H/Z-100 computers. VEGA-BOUND is a broad adventure game that includes star navigation, encounters with black holes, combat with enemy space ships, and landings on alien worlds. This game allows players to think as well as react and requires 128K memory, either 32K or 64K video memory, version 1.2 firmware (monitor ROM) or later, and an RGB color or monochrome monitor.

Vendor: Interdiscipline, Inc.
403 S. Brandon
Seattle, WA 98108
(206) 763-2099
Price: $44.95

Letterform 1000, The Letter Writer's Reference

PBL Corporation has announced Letterform 1000, a computer disk letter reference that contains over 1000 letters and forms ready to interface with most word processing software. Quality letters can be called to the screen, edited, personalized, and quickly printed. Letters and forms included cover accounting and collection, employees/employers, general business, goodwill and sales, legal, ordering and shipping, personal, school, charity, and organizations. All are categorized and indexed in the documentation. The manual also includes abbreviations; proper headings; titles and forms of address; academic degrees; U.S. federal, state, and Canadian provincial government abbreviations; and complimentary closings. The letters and forms are stored in standard ASCII text files and are available for a wide range of MS-DOS compatible machines, including the H/Z-100 under Z-DOS.

Vendor: PBL Corporation
P.O. Box 559
Wayzata, MN 55391
(612) 473-8998
Price: $95.00

Price Reductions for COMMSOFT Products

COMMSOFT has announced major price reductions for some of its software. ROOTS, a popular genealogy program; and RTTY89, CW89, and CIPHER89, programs for HDOS to communicate using radio teletype (RTTY89), Morse code (CW89), or tune and copy shortwave RTTY and CW signals (CIPHER89), have all been reduced.

Vendor: COMMSOFT
665 Maybell Ave.
Palo Alto, CA 94306
(415) 493-2184
(800) 227-1617, ext. 585
(for placing orders from outside CA)
(800) 772-3545, ext. 585
(for placing orders from inside CA)

Prices:	ROOTS/M (CP/M vers)	$49.95
	ROOTS89 (HDOS vers)	$39.95
	RTTY89 (HDOS only)	$39.95
	CW89 (HDOS only)	$39.95
	CIPHER89 (HDOS only)	$39.95

CP/M for Z-DOS

Two programs have been announced that allow you to run CP/M programs under Z-DOS (both will be reviewed in a future issue). One is from HUG (described in the January, 1984, issue of REMark under "1983 HUG Releases", page 41), the other is from Software Toolworks.

ZP/SIM lets CP/M programs run directly from a Z-DOS disk, and operate on Z-DOS data files without requiring use of CP/M. CP/M programs can be executed in one of two ways: in "command mode" or "convert mode".

In the command mode, ZP/SIM looks like CP/M at the command level. CP/M commands and programs may be executed directly.

In the convert mode, CP/M program files are modified by ZP/SIM to run directly under Z-DOS.

Most CP/M compilers, editors, business programs, applications, and utilities will run under ZP/SIM. Programs that read the CP/M directory, use absolute disk track and sector addressing, bypass CP/M to perform I/O, or use Z-80 opcodes will not work.

Vendor: The Software Toolworks
15233 Ventura Blvd., Ste. 1118
Sherman Oaks, CA 91403
(213) 986-4885
Price: $29.95

Z-100
Word Wiggle Available for Color Systems

Word Wiggle is a word square vocabulary game with color display that displays a four by four square of letters. The player must find as many words as possible within a time limit. His opponent is the computer, playing at one of eleven skill levels within a 30,000 word dictionary.

Vendor: The Software Toolworks
15233 Ventura Blvd.,
Ste. 1118
Sherman Oaks, CA 91403
(213) 986-4885 prior to 1/1/84
(818) 986-4885 after 1/1/84
Price: $29.95

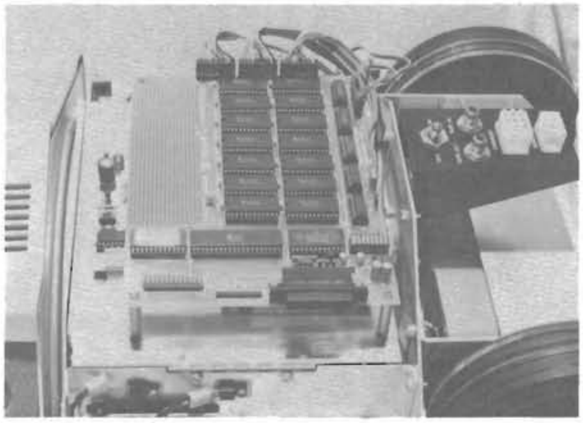
Report Explains Tax Breaks for Computer Owners and Buyers

Tax Breaks for Computer Buyers is a report that explains the tax rules surrounding taking deductions for a personal computer. The author, Vernon Jacobs, a CPA, says that a computer can be deducted by those who use it in a business--whether part time or full time, and by many people who use it to manage their investments, develop programs for sale to the public, and, in some cases, by employees who use it in their vocation. 26 pages.

Vendor: Research Press, Inc.
4500 West 72nd Terrace
Box 8137-P
Prairie Village, KS 66208
(913) 362-9667
Price: $9.00

Hero I Interfaces and Peripherals

Micromation has announced several peripheral items for Hero I. MEMCOM provides a means to develop programs for Hero I using any computer with an RS-232 port. The product includes two 8-bit bidirectional parallel ports (with handshaking), two 16-bit timers, an RS-232 serial port, onboard serial software for up and downloading programs via the serial port, all hardware for mounting the board on the rear door of Hero I and interfacing to the robot's CPU board, and a complete instruction manual and schematics.



VCS (Voice Command System) consists of VOREC (Voice Recognizer) and the VOCOL (Voice Command Language) software. The VCS manual contains a complete description of how to use VOREC under program control with HERO I. The VOREC board mounts next to and interfaces with MEMCOM and includes voice word recognition accuracy of about 98%, 256-word vocabulary, external speaker input (rather than microphone), and complete instructions and installation manual. The software (VOCOL) is similar to high level languages such as BASIC, except that it is Voice-driven through VOREC. When first implemented, the software has Hero I talk to you through a voice training session where you are asked to repeat the words in his command vocabulary three times. A 6808 Source Code for VOCOL is available (on an APPLE DOS 3.3 disk) as an option (not separately) to the VCS system.

Vendor: Micromation, Inc.
9104 Red Branch Road
Columbia, MD 21045
(301) 730-1237

Prices:	MEMCOM	$295.00
	VCS	$595.00
	VOCOL Source Code (only with VCS)	$55.00

Index of Advertisers

21st Century Data, Inc.	*49*
Apogee Software	*65*
BUSS	*54*
William M. Campbell, M.D.	*16*
Cherry Engineering	*19*
Controlled Data Recording Systems Inc.	*28,49*
Floppy Disk Services Inc.	*13*
Generic Software	*41*
Husker Systems of Nebraska, Inc.	*4*
Mako Data Products	*59*
Micro Innovations	*59*
Micromation Inc.	*6*
MicroServices	*48*
Micro-Systems Software, Inc.	*68*
Newline Software	*41,47*
North Coast Intelligence Inc.	*41*
S & K Technology, Inc.	*48*
Secured Computer Systems	*44*
Skill Data	*48*
Software Support, Inc.	*20*
Software Toolworks, The	*63*
Software Wizardry	*33*
Studio Computers, The	*2*
Virtual Devices Inc.	*40*
Westcomp	*65*

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

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 ____________________

REMEMBER - ENCLOSE CHECK OR MONEY ORDER

CHECK THE APPROPRIATE BOX AND RETURN TO HUG

	NEW MEMBERSHIP RATES	RENEWAL RATES
US DOMESTIC	$20 ☐	$17 ☐
CANADA	$22 ☐	$19 ☐ US FUNDS
INTERNAT'L*	$30 ☐	$24 ☐ US FUNDS

* Membership in France and Belgium is acquired through the local distributor at the prevailing rate.

"They shouldn't have touched my MTERM"

"I only left my keyboard for a few minutes ... when I returned, I found Stamitz from accounting and Miss Sashshay from the secretarial pool fondling my MTERM. Now if I've told them once, I've told them a hundred times ... use my coffee cup. Borrow my key to the employee lounge. Bend my paper clips but, leave my MTERM alone!! Did they listen? Nooooo! Well, I guess I lost my head. Both Stami and Sashshay are doing fine. They should be released from the hospital any day now. For me, it's an entirely different story."
Signed, 0076697

Why MTERM creates such fanatically touchy users is not so hard to understand. MTERM turns the Z-100 into something altogether better. MTERM lets your Z-100 work a lot faster, with more accuracy, efficiency and dependability. MTERM also has a lot of features that users such as 0076697 find positively endearing. For instance, the ability to translate any ASCII character to another, and an easy to read operating guide that makes using MTERM as easy as ... well, bending a paper clip.

So to increase productivity, to increase the speed and accuracy of your Z-100. MTERM Only $79.95! Still the best program for the cost!
P.S. "Get your own MTERM" 0076697.

MICRO-SYSTEMS SOFTWARE, INC.
4301-18 Oak Circle, Boca Raton, Florida 33431, Telephone: (305) 983-3390
Toll Free: 1-800-327-8724

Heath/Zenith Users' Group
Hilltop Road
Saint Joseph, Michigan 49085

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

Volume 5, Issue 3

POSTMASTER: If undeliverable, please do not return.

885-2050