REMark®

Volume 6, Issue 6 • June 1985

P/N 885-2065 Issue 65

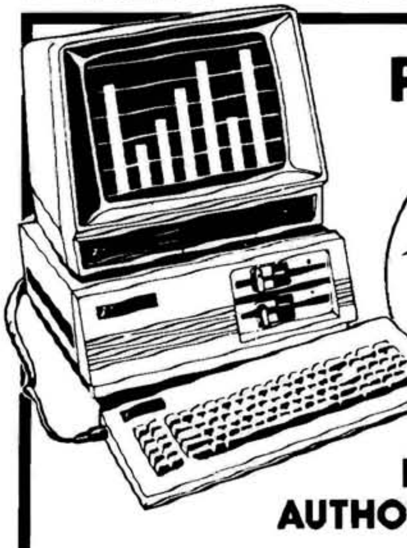
Official magazine for users of HEATH Zenith computer equipment.

$2.50

INTRODUCING THE ZP-150 PC LAPTOP

VALUE BULLETIN

PRICE PERFORMANCE BREAKTHRUS



ZENITH data systems

FACTORY AUTHORIZED SERVICE

ALL SYSTEMS INCLUDE 320K RAM, COLOR GRAPHICS, DOS 2.1
Parallel & Serial Ports Microsoft Word & Multiplan.

COMPUTERS

Z150 **Dual Floppy System**	**$ CALL**
Z150 **10 meg Winchester**	**$ CALL**
Z160 **Portable Dual Drive**	**$ CALL**
Z160 Portable Winchester	**$ CALL**

MONITORS

ZVM122 **Amber Monitor**	**$ CALL**
ZVM123 **Green Monitor**	**$ CALL**
ZVM124 **Hi Res Monchrome**	**$ CALL**
ZVM133 **Hi Res Color**	**$ CALL**

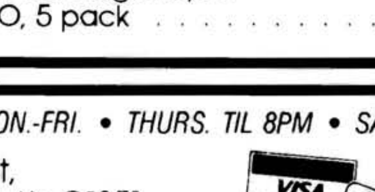
OKI-DOKI

Factory Authorized Service

OKIMATE 20 Color Printer with Serial or Parallel Interface	**$199**
OKI ML-182 120 CPS Dot Matrix IBM or Standard full graphics	**$249**
OKI ML-192 160 CPS Dot Matrix IBM or Standard full graphics	**$399**
FIRST QUALITY CABLES FOR ABOVE	**$22.95**

GUARDIAN DATA SYSTEMS "Protecting Your Information Investment"

25 MEGABYTE BACKUP Cipher 525 Cartridge tape drive uses shugart 824 floppy drive interface - G25X	**$ 495**
G25-Z100 Complete unit w/case power supply and software for Z100	**$1095**
G25KT-1 Kit for Z100 includes drive software and interconnect cable	**$ 695**
G25SFT Complete Software and Programmers Kit. Build your own with addition of drive	**$ 149**



G25PIK Programmers kit with source listing on disk - interface to non Z100 Systems	**$ 49**
DC600A Cartridge Tapes ea $29.00, 5 pack	**$ 139**

SYSTEMS INNOVATIONS INC.

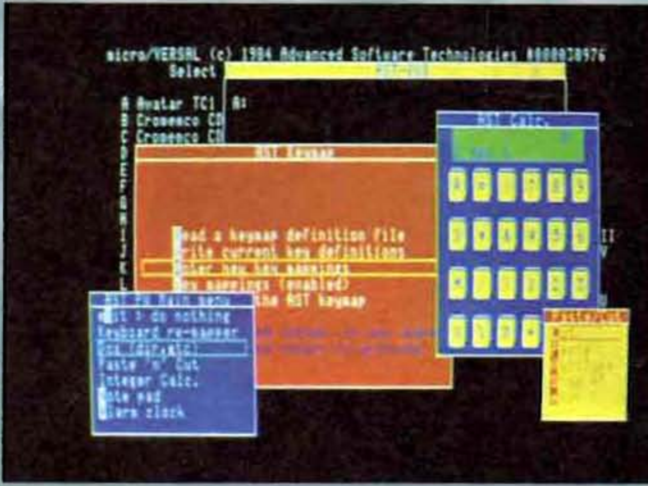
HOURS: 9AM-6PM, MON.-FRI. • THURS. TIL 8PM • SAT. 10AM-3PM

44 Stedman Street,
Lowell, Massachusetts 01851
(617) 459-4449
"Your Complete Computer Resource Center"

VISA MasterCard AMERICAN EXPRESS

Genie™

Software Magic from Advanced Software Technologies



Shown are Genie windows as they appear on the Z-100. From the left: The Genie main menu, the Genie keyboard macro generator the Genie DOS, Calculator, and Cut and Paste menu. In the back is our micro/VERSAL disk to disk file transfer program, running as a regular application.

	Sidekick	Spotlight	Genie
Key macros	69.95	No	**Yes**
DOS services	No	Yes	**Yes**
Cut and Paste	No	No	**Yes**
Notepad	Yes	Yes	**Yes**
Ascii table	Yes	No	**Yes**
Calculator	One	One	**Two**
Calendar	Yes	Yes	**Yes**
Rolodex	Yes	Yes	**Yes**
Alarm	No	No	**Yes**
Load separately	No	No	**Yes**
	$120	$150	**$49.95**

Special Introductory Offer:

$49.95

plus $4.00 shipping and handling
New York residents add 8.25% sales tax
To order send MC/VISA check or M.O. to:

AST's Genie is always there with you. With a simple keystroke it appears, and is at your command to help you, and make your work easier. Genie can move data between your applications, take notes, schedule appointments, and much more. Genie helps with everything from BASIC to Lotus 1-2-3.

Genie is available for the Zenith Z-100 and IBM-PC compatibles. Genie requires MS-DOS version 2 or above.

ADVANCED SOFTWARE TECHNOLOGIES

452 West 47th Street
New York, NY 10036
(212) 247-0150

Sidekick, Spotlight, MS-DOS, Lotus 1-2-3, IBM-PC, and Z-100 are trademarks of Borland International, Software Arts, Microsoft, Lotus Development, IBM, and Zenith.

Heath/Zenith Users' Group

Staff

Manager .... Bob Ellerton
(616) 982-3867

Software Engineer .... Pat Swayne
(616) 982-3463

Bulletin Board and
Software Developer ... Jim Buszkiewicz
(616) 982-3463

Software Coordinator ..... Nancy Strunk
(616) 982-3838

Secretary .... Margaret Bacon
(616) 982-3463

REMark

Editor .... Walt Gillespie
(616) 982-3789

Editorial and Advertising
Assistant .... Lori Lerch
(616) 982-3794

Printers .... Imperial Printing
St. Joseph, MI

	U.S. Domestic	APO/FPO & All Others
Initial	$20.00	$35.00*
Renewal	$17.00	$30.00*

* U.S. Funds

Limited back issues are available at $2.50, plus 10% shipping and handling — minimum $1.00 charge. 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/Zenith 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®

Official magazine for users of HEATH ZENITH computer equipment.

Volume 6, Issue 6 • June 1985

Index of Advertisers

This index is provided as an additional service. The publisher does not assume any liability for errors or omissions.

On The Cover: *Pictured is one of Zenith's newest additions to it's line of Personal Computers. The ZP-150. See story on Page 19.*

Photo courtesy Ed Quinones, Heath Advertising Department.

Transfer "Foreign" Files in Seconds

NOW AVAILABLE FOR Z-100 AND Z-150

No modems and RS-232 links required!

ONLY $39.95!

READ, WRITE and FORMAT more than 70 different CP/M and MS-DOS disks on your computer

with MEDIA MASTER™

Tired of hunting for software in your computer's disk format? Do you need to exchange your Lotus 1-2-3, dBase II, Multiplan, and Wordstar data files among different CP/M and MS-DOS computers?

Then MEDIA MASTER is for you!

MEDIA MASTER supports the following disk formats:

Actrix (SSDD)
Actrix (DSDD)
Avatar TC10 (DSDD)
Casio FP 1000 (DSDD)
Chameleon CP/M-80
Columbia MPC CP/M-80
Cromemco CDOS (SSDD)
Cromemco w/Int'l Term (DSDD)
Cromemco w/Int'l Term (SSDD)
DEC VT180 (SSDD)
Davidge (DSDD)
Digilog (DSDD)
Epson Multifont (DSDD)
Epson QX-10 (DSDD)
Fujitsu Micro 16s (DSDD)
Groupil III CP/M (DSDD)
H/Z Z-100 CP/M (DSDD) - later
H/Z Z-100 CP/M (SSDD)
H/Z Z-100 Z-DOS 1.xx (SSDD)
H/Z Z-100 Z-DOS 1.xx (DSDD)
H/Z Z-100 Z-DOS 2.xx (SSDD)
H/Z Z-100 Z-DOS 2.xx (DSDD)
H/Z Z-90 40 trk, 1 k blk (SSDD)
H/Z Z-90 40 trk, 2 k blk (SSDD)
Heath w/Magnolia CP/M (SSDD)
IBM PC CP/M-86 (DSDD)
IBM PC CP/M-86 (SSDD)
IBM PC-DOS 1.xx (DSDD)
IBM PC-DOS 1.xx (SSDD)
IBM PC-DOS 2.xx (DSDD)
IBM PC-DOS 2.xx (SSDD)
IDEA Bitelex (SSDD)
ISM CP/M (DSDD)
Insight Dev. IQ-120 (SSDD)
Kaypro II/2 (SSDD)
*Kaypro 4, 10 (DSDD)
LNW-80 (SSDD)
Lobo Max-80 (SSDD)
Lobo Max-80 512 (SSDD)
Micral 9050 CP/M-80 (DSDD)
Morrow MD 11 (DSDD)
Morrow MD 2 (SSDD)
Morrow MD 3 (DSDD)
NCR Decision Mate 5 (DSDD)
NEC PC-8001A (SSDD)
NEC PC-8001A (DSDD)
Olympia ETX II (SSDD)
Olympia EX100 (DSDD)
Osborne (SSDD)
Osborne 4 (DSDD)
Osborne Osmosis (SSDD)
Otrona (DSDD)
PMC Micromate (DSDD)
Reynolds & Reynolds (SSDD)
Sanyo (DSDD)
Superbrain (DSDD)
Superbrain Jr. (SSDD)
Systel II (SSDD)
Systel III (DSDD)
TI Professional CP/M-86 (DSDD)
TRS-80 III FEC CP/M (SSDD)
TRS-80 III FEC T805 (SSDD)
TRS-80 III Hurr. Labs (SSDD)
TRS-80 III Mem. Merch (SSDD)
TRS-80 IV CP/M+ (SSDD)
TRS-80 IV Mont. Micro (SSDD)
Teletek 40 trk (SSDD)
Toshiba T100 (DSDD)
TurboDos (DSDD)
Wang Maws CP/M (DSDD)
Xerox 820 II (SSDD)
*Zorba 40 trk (DSDD)

MEDIA MASTER *is also available for DEC Rainbow, Osborne, and Kaypro computers.*

**Requires access to "foreign" computer*

NEW! — FILE MATE™

A Dozen Utilities in One!

ONLY $29.95

Finally — a powerful, easy to use file utility that performs a host of functions. **FILE MATE's** numerous text sifting and paragraph reformatting modes help you exchange "foreign" text files. Other file inspection and repair tools include two binary dumps, an ASCII/HEX file compare, and CRC "fingerprint." FILE MATE is available for popular CP/M and MS-DOS computers.

INTERSECTING CONCEPTS
(formerly MDG & Associates)

4573 Heatherglen Ct., Moorpark, CA 93021

ORDERING INFORMATION:

Include $3 per order for postage/handling. Overseas airmail add $10. To place COD orders:

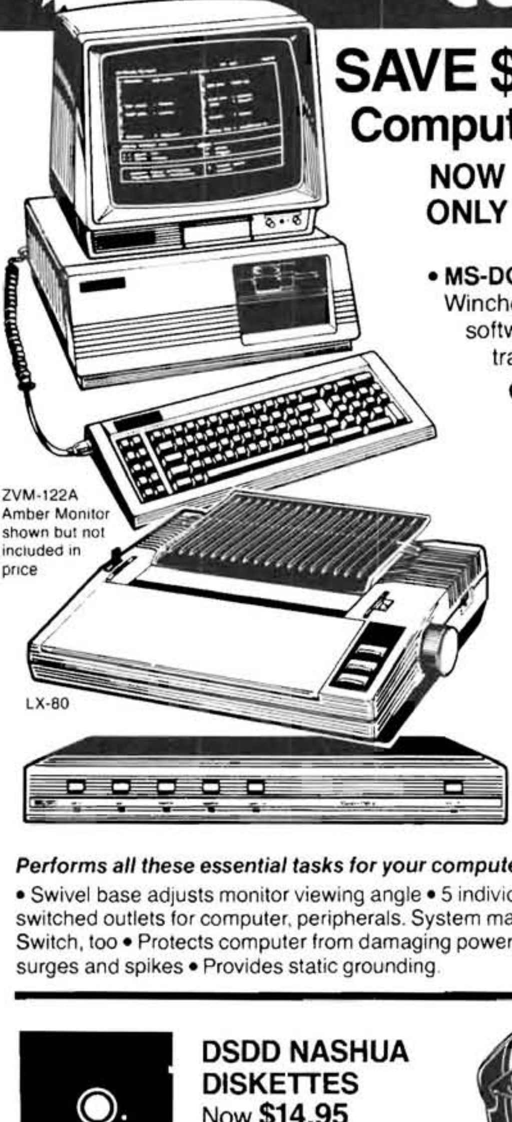
Call TOLL FREE 24 hours:
800-824-7888 (Ask for Operator 251)
Alaska, Hawaii: 800-824-7919 (Ask for Operator 251)
Technical questions: call 805-529-5073

INCLUDES MICROSOFT WORD AND PLAN SOFTWARE

HEATH/ZENITH COMPUTERS & ELECTRONICS

IBM-PC COMPATIBLE COMPUTER

INCLUDES **640K** MEMORY



SAVE $1449 on IBM-PC Compatible Computer – with software & 640K Memory

NOW ONLY **$1499**

Heath HS-151-21 IBM-PC Compatible Computer Kit

• **MS-DOS – full IBM-PC compatibility** • Single 360K Disk Drive (Dual and Winchester disk drives also available) • Microsoft WORD and MULTI-PLAN software included • Famous Heath/Zenith quality • Complete user training available • Full service back-up. Monitors from only **$89.95**

Or, buy the famous Zenith ZF-151-21 IBM-PC Compatible Computer with all the software and options above, factory-assembled and tested... only $1979.

NEW Epson LX-80 PRINTER

Introductory price just **$239⁹⁵** Reg. $369.95

• Correspondence quality FRICTION FEED printer prints directly on your letterhead • Fast! 100 characters per second • Select typestyles at a touch. Over 150 combinations • Mix text and graphics on same line • One-Year warranty.

NEW MasterPiece 4-function POWER CONTROLLER

Performs all these essential tasks for your computer:

• Swivel base adjusts monitor viewing angle • 5 individually switched outlets for computer, peripherals. System master Switch, too • Protects computer from damaging power surges and spikes • Provides static grounding.

Intro•Sale price just **$99⁹⁵** Reg. $149.95

SUPER PRICE!

DSDD NASHUA DISKETTES

Now **$14.95** per box of 10

For IBM, Apple, Zenith, Commodore, Atari and more. Lifetime replacement guarantee. *No limit.*

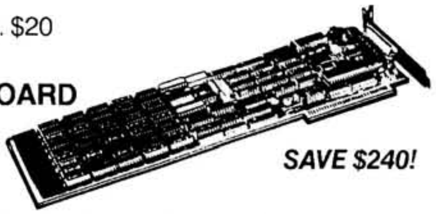
FREE DISK Bring in ad for free diskette!

PROTECTED DISK HOLDER

Lockable high-impact plastic holder keeps dust and fingers away from 70 disks.

Now **$12.88** Reg. $20

320K MULTI-FUNCTION BOARD



Now **$299** Reg. $539

SAVE $240!

Multi-function "Tecmar Captain" expansion board for IBM-PC and compatibles. 320K memory, serial and parallel parts, battery backup. Clock Calendar and 24 programs included.

Prices Good Through July 13, 1985

NATIONAL HUG MEMBERS SAVE 10% ADDITIONAL OFF THESE PRICES

Phone orders accepted.

Inquire about special corporate and government sales and service programs

ZENITH | data systems Authorized Dealer

Your TOTAL SERVICE computer center

• Service • Support • Software • Accessories • User Training • Competitive Prices

HZC-137

Heath ZENITH®

Units of Veritechnology Electronics Corporation

Computers & Electronics

HUGCON '85

Bob Ellerton
HUG Manager

The 1985 International Heath/Zenith Users' Group Conference Is Underway!

Mark your calendar now and send in your registration! The Fourth International Heath/Zenith Users' Group Conference will be held from Friday, August 9 to Sunday, August 11, 1985. Our Conference host for the weekend will be the O'Hare Hyatt Regency in Rosemont, Illinois. The Rosemont Hyatt is very close to O'Hare International Airport providing convenient transportation from most points of the globe. The Hyatt does provide ground transportation to and from the airport.

The plans for the Conference are now well underway with plenty of exhibit space and meeting rooms reserved for the increased activities this year. Rumors indicate that some of the exhibits will introduce new products for all of the Heath and Zenith computers including the H8 and H89. We are contacting a number of vendors with products for the H/Z-151 and H/Z-161 computers. The Exhibit Area will cover the entire bottom floor of the Hyatt.

Walt Gillespie is in charge of the Exhibit Area for this year's Conference. Walt has sent out the Exhibitors Package for this year's HUGCON. If you have not received a package and you are interested in exhibit space for this important event, please get in touch with Walt, as soon as possible. Space for the Exhibit Area is limited.

Walt Gillespie, Exhibit Coordinator
Int'l. Heath/Zenith Users' Group Conference
Hilltop Road
Saint Joseph, Michigan 49085

Jim Buszkiewicz is heading up the various discussion groups and meetings. Jim has indicated that we have 18 confirmed speakers and talks for the Conference and to date, we are not even able to tell you what the representatives from Heath and Zenith will be discussing. To put it another way: "We can't discuss future products (until August)." Nancy Strunk is in charge of the HUG booth located within the Exhibit Area. Margaret Bacon, as always, will handle your registration and provide information about the Conference on request. As you can see, it looks like we are all going to have a busy weekend this August.

If you have looked over the Registration Form from last month, you will see that we are taking a list for those of you attending the Conference, but looking for a little extra from the event. Please take a bit of time to indicate some possible activities around the Chicago area you might be interested in. If there is enough interest in activities for the family or "computer widow", we will have an individual available to help you plan some fun.

Let's take a little space here to review what will be going on at the Conference, as far as activities and meetings this year.

As you probably know, VEC (the Heath/Zenith Computers and Electronics Centers) will be represented again with the promise of bargains for those of you planning to attend this year's Conference. The guys from the stores are already preparing the prices to be handed out when you come in the door.

As mentioned previously, 18 talks have already been scheduled for HUGCON '85 with subjects ranging from software to hardware on many of the existing Heath/Zenith computers. Next month, we will outline all of the talks and speakers so that you can select those discussions that will be of most interest to you. Also, Heath Product Line Managers will be available for both a review of products that you have seen or are about to see from Heath. Get some questions or suggestions ready for these guys.

Jim indicated that two workshops will be conducted during the Conference for both hardware and software. You will be given the opportunity to write down some of your most nagging questions to be answered during the workshop you attend. Cards will be available for your questions, but now is a good time to start thinking about what you would like to ask the experts. These workshops will be a new addition to HUGCON that will give each of us the mechanism to ask those embarrassing questions that "everybody else knows".

Charlie Floto, Publisher of Sextant and Buss, will be covering the historical activities of the Heath/Zenith community for the past eight years. Charlie has been involved with the Heath/Zenith user base even before the first REMark hit the streets and has a good handle on the events, both positive and negative, that have brought a small computer users community of under 2,500 members to a major commercial group with over 26,000 members and growing.

Also scheduled for a look at Heath's activities is Chas Gilmore, Vice President of Product Development for Heath. Chas was one of the original members of the Heath team that created the H8. (You might keep in mind that Chas was also responsible for that great computer product known as the H-10.) Chas will be telling us a little about the past design considerations that brought the H8 to life, as well as what directions he sees Heath going in the future. Chas knows the computer industry and is ready to answer your questions about Heath's activities. Don't miss this one!

Of course, the HUG bunch will have plans for prizes and dinner that are all part of your registration fee. We hope that many of you will be able to join us at the Fourth International Heath/Zenith Users' Group Conference. Next month, we will outline the schedule for the entire Conference. Also, watch next month for complete details and information about the various talks and discussion groups so that you may select from the large number of subjects to be covered this year (maybe we should consider another three days).

Heath/Zenith
Users'
Group

International HUG Conference

*** IV ***

August 9, 10 and 11, 1985
O'Hare Hyatt Regency
Chicago, Illinois

It's time to plan your visit with other members of the Heath/Zenith Users' Group. This year, we are returning to the newly remodeled O'Hare Hyatt Regency, Chicago for HUGCON IV.

*More exhibits, more speakers and **more room** highlight the Conference activities for 1985. The Official Conference Registration form will appear in next month's issue of **REMark**, the official magazine for users of Heath/Zenith computer products.*

Plan to attend now! We sure hope to see you there!

Heath/ZENITH
Users'
Group

INTERNATIONAL
HEATH/ZENITH USERS' GROUP CONFERENCE

Official Conference Registration Form

O'Hare Hyatt Regency
Rosemont, Illinois
August 9, 10, 11

Name(s): ______________________________

____

____

Company: ______________________________

Address: ______________________________

City: ____________________ **State:** __________ **Zip:** __________

Enclosed is $25.00 for each of the individuals listed above to attend the International HUG Conference being held the weekend of August 9, 10 and 11, 1985. Please send tickets along with information regarding hotel reservations and transportation.

Amt. Enclosed: ________ No. Attending: ________

For Our Information:

Which Heath/Zenith computer do you now operate? ____________________

Are you a Non-User-Attendee? Yes No

Are you a computer related manufacturer? Yes No

If yes, would you like exhibit information? Yes No

Are you, or anyone in your party, interested in activities in or around the Chicago area other than the Conference? Yes No

If yes, please indicate any suggestions you may have:

____

Special Notice To Exhibitors:

Exhibitor Information Packages are available on request from the Heath/Zenith Users' Group. Those of you interested in exhibiting your products should contact us as early as possible to ensure a position at this year's event.

For Your Information:

The $25.00 you are paying for your reservation to the International HUG Conference entitles you to all functions of the Conference. Visitor tickets, for those of you simply attending the seminars and exhibits, are available for $10.00. Visitor tickets do not include eligibility for prizes or food while attending the Conference.

Please send your completed registration form or suitable copy to:

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

Registration(s) must be post marked no later than July 31, 1985. Cancellation will not be accepted after this date.

BUGGIN' HUG

All Things Change

Editor:

I purchased my ET-100 early this year and up until a couple of weeks ago was happy to work with ZDOS, a decided improvement over both HDOS and CP/M on my H89.

But, all things change, so I ordered MS-DOS Version 2.0. Have only had it one day, but managed to run up against it when trying to move Pascal and BASIC source files from ZDOS.

Looked back through my issues of REMark to see if anyone else reported the same difficulty, then I called the Heath Software Consultant. Seems that this problem is prevalent only on those systems using 96 tpi drives and is easily overcome with a patch.

As explained to me by Heath:

With a ZDOS disk with DEBUG installed in Drive A, and the disk with the source files you want to move in Drive B:

(Note, I maintain all my Pascal and BASIC source files on a separate disk, so I did a DSKCOPY to create a duplicate source file disk for MS-DOS).

```
A>DEBUG <return>
>L1ØØ 1 1 1       <return>
>E1ØØ             <return>
>xxxx:Ø1ØØ FD.FB <return>     (enter FB after FD.)
>W1ØØ 1 1 1       <return>
>Q                <return>
A>
```

What I have now are two disks, both of which hold the same source files, but one can be read by ZDOS, the other by MS-DOS Version 2.0, until such time as I discontinue using ZDOS, which won't be long, judging by what I have observed of MS-DOS thus far.

After moving the files on to an MS-DOS disk, if you want to return the above disk back to its original configuration, then use DEBUG with MS-DOS in Drive A, and the data disk in Drive B.

```
A>DEBUG <return>
-L1ØØ 1 1 1       <return>
-E1ØØ             <return>
-xxxx:Ø1ØØ FB.FD <return>     (enter FD after FB.)
-W1ØØ 1 1 1       <return>
-Q
A>
```

Now the data disk can once again be read by ZDOS.

Another item of interest for those considering the purchase of Wong's Advanced Technologies RAMDRIVE as described in REMark Volume 5, Issue 4 by Walt Gillespie.

I have a ZBASIC program designed to output the weekly and seasonal statistics normally done by a bowling secretary. For a 12 team league, with 5 bowlers on a team, this program requires close to 100 separate files. I had hoped to utilize RAMDRIVE to speed up the program. However, after setting up a 64K disk in RAM, I found I had only 30 directory entries.

As it turned out, without DSKCOPY, copying the files to and from the ramdisk would have taken almost as much time as that saved using RAMDRIVE, so nothing was really lost.

It is rumored (as forwarded by Omega) that an updated version of RAMDRIVE will come on line early in November which will support MS-DOS Version 2.0. Will certainly be interesting to see if directory entries will be limited only by ramdisk space, as in MS-DOS 2.

Sincerely,

Donald E. Risher
908 Charlotte Place
Charleston, WV 25314

For All The "Hardware Hackers"

Dear HUG,

I have a question for all the "hardware hackers" out there. Is anybody working on a method to implement the entire 768Kb of user RAM on the motherboard of the Z-100 series computers using the new 256K bit chips? It would seem like a logical implementation to me; replace the 64K bit chips on the motherboard with the new 256K bit chips. I understand it would require new PALs be installed too, but that should be possible, huh? That would allow one to fully populate the video RAM board with the 64K bit chips from the motherboard, giving full color capability in interlaced graphics mode. Sounds like a great idea to me!

Sincerely,

Jay K. Joiner
300 Luchana Drive
Litchfield Park, AZ 85340

Magic Wand — (EDIT And PRINT Are Both Release 1.1)

Dear HUG,

I am running Magic Wand under MMS CP/M on a Z-89 with three H-17 drives and a Corvus 20MB hard disc divided into 9 partitions, i.e. looks like nine disk drives to the Z-89.

Although I can EDIT a file on Drive J:, I can't save to J: using the X= command or INCLUDE from J: (or for that matter, any drive past Drive F).

In the February '85 issue of REMark, Neil Rickert described how Magic Wand could be patched to handle more drives, but when I phoned him, it turned out he had Release 1.11 and his patches wouldn't work for me.

I am a business user and don't know any programming languages, but have used DDT by the numbers to install patches. If anyone can help me, I would certainly appreciate it.

Sincerely,

Stoffel Seals Corporation
John F. Stoffel
68 Main Street
P.O. Box 278
Tuckahoe, NY 10707

Check Out WSOK.COM

Dear HUG,

To reply to Craig Tucker's question in the April 1985 issue of REMark, I'd like to suggest that he check out WSOK.COM by Wheatland Design Laboratory of Lawrence, Kansas. The program allows you to select multiple fonts (Italics, Greek and Double High fonts are supplied) along with being able to use other features of your Okidata 92 printer, including microjustification. Also, there is a font compiler program supplied with the package.

Along with the above, you can select "true" Super- and Subscripts along with variable line spacing in increments of either 48ths of an inch or in 144ths of an inch. The .FI Fname command allows you to print a file within another file, which emulates a MailMerge command. These multiple files can be inserted in the inserted file up to six levels deep. Naturally, double-wide is also available.

By using alternate control characters and special dot commands, one can address many of the functions available with your printer.

I'd like to strongly suggest that Mr. Tucker try out this package to allow him to access some of the printer's capabilities.

Additionally, I had some difficulty with my original copy of the program where I did not inform them that I had been using Magnolia Microsystems CP/M. They sent me a new copy of the program which has been configured for Magnolia for no additional charge. They did it quickly and efficiently, which is nice to see at any time.

Thank you for your attention in this matter.

Sincerely,

Paul L Eustace
North Houston HUG
8110 Tattershall Circle
Humble, TX 77338

MS-DOS Version 2 And The GC-1000 Clock

Dear HUG,

This article is for the benefit of the people who have the GC-1000 Most Accurate Clock interfaced with an H/Z-100 in order to have the clock set the system time and date automatically at bootup time.

My husband and I were very happy to discover the Assembly language program written by Jim Schuster as published in the January 1984 issue of REMark to interface the clock to our H/Z-100. It proved to be a great asset.

When we obtained MS-DOS Version 2 in November 1984, we found that the interface program did not work. It seemed to go to "Never Neverland". Neither my husband nor I program in Assembly language, but we undertook the task of trying to make the program work under Version 2.

Our first thought was to reassemble the program under DOS 2. But the program would not assemble as written. First, we had to 'Include' the file — DEFDOSI.ASM — at the beginning of the program along with the 'INCLUDE DEFMS.ASM' file in order to get the DOSI_FUNC and DOSI_TERM variables defined. Second, we had to redefine the IN_BUFFER and OUT_BUFFER. The changes to the buffer variables are as follows:

```
IN_BUFFER       DB    25 DUP (?)
OUT_BUFFER      DB    25 DUP (?)
```

Then the program assembled — but it wouldn't run correctly. It was still going off to "Never Neverland".

After several inserts in the program to display things on the screen to determine which of the loops was getting hung up, my husband came up with what proved to be the solution.

We had read in several places that DOS 2 was considerably slower than ZDOS, as far as sending output to the printer. So my husband decided to change the baud rate of the clock. The baud rate in the original program was set at 9600 baud. He changed it to 4800 — still no luck, however, we had more displays on the screen which indicated to us we were on the right track. He then changed the baud rate to 2400 baud, and low and behold the program worked.

We then changed just the baud rate on the original program, reassembled it under ZDOS, transferred it to the boot-up hard disk partition and it runs under DOS 2.

To make the changes for the baud rate — first, change the settings on the clock (check the manual). Then in the program, the 'REPEAT:' section should read:

```
REPEAT:    IN    AL,EPCIB_CR
           MOV   AL,Ø1ØØ111ØB
           OUT   EPCIB_MR,AL
           MOV   AL,1111Ø11B             ;24ØØ baud
```

The last line is the only one changed from the original program. The other lines are provided only as a means of reference. All remaining lines are the same.

So, if you have the problem of making the program work under DOS 2, there are two options to solve it. The first is to change the baud rate on the clock and in the ZDOS version of the program. Reassemble it under ZDOS, link it and convert it to a COM file with EXE2BIN. Then transfer it to the boot-up drive or disk and put it in an autoexec.bat file.

The second option is to change the baud rate on the clock and change the baud rate in the original version of the program. And, in addition, redefine IN_BUFFER and OUT_BUFFER (noted above), and INCLUDE the additional file DEFDOSI.ASM. Assemble it under MS-DOS, link it and convert it to a COM file with EXE2BIN. Then transfer it to the boot-up drive or disk and put it in an autoexec.bat file.

I hope this will help those of you who have the same problem.

Barbara J. Gabner
Secretary NH-HUG
20207 Cottonglade
Humble, TX 77338

Professional Looking Equipment With A Plotter

Dear HUG:

Here is a tip for HUGgers who build electronic equipment and also have access to a plotter. The results can yield very professional looking equipment.

First write a routine for the plotter to draw a full scale drilling

Vectored to Page 62 ☞

PRINTER PLUS PAK

MPI'S EXCLUSIVE PERFORMANCE ENHANCEMENT PACKAGE FOR YOUR PRINTER . . .

FEATURING . . .

64K
MEMORYMATE™
BUFFER

REPEAT
MODE FOR
99 ORIGINAL
DOCUMENTS
without computer tie up

Only
$99⁰⁰

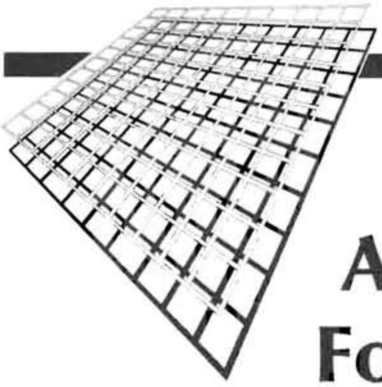
SideWinder
Mode — For
Wide Spreadsheets

ONLY AT HEATH® ZENITH® COMPUTERS & ELECTRONICS CENTERS
AND ZENITH DATA SYSTEMS DEALERS

Units of Veritechnology Electronics Corporation
To find your nearby ZDS dealer call (800) 842-9000, ext. 1.

The Printer People
mpi®

Micro Peripherals, Inc.
4426 S. Century Dr.
Salt Lake City, Utah 84123
801-263-3081 — 800-821-8848



ANSI
A Standard Screen Interface For Heath/Zenith Computers

Glenn F. Roberts, Ph.D.
12048 Greywing Square, #C-3
Reston, VA 22091

The American National Standards Institute (ANSI) has defined a standard mechanism for controlling "two-dimensional character-imaging input-output devices" via sequences of ASCII characters (so-called "escape" sequences). The standard applies to a number of different devices, such as phototypesetters and word processors, but is of most interest to us in its application to Heath/Zenith computers and terminals.

One reason this standard is important to Heath/Zenith computer users is that it provides a mechanism for controlling the screen (moving the cursor, controlling reverse video, auto-wrap, etc.) which can be used on all Heath/Zenith computers, including the H-89, H-19, H/Z-100 and H/Z-100 PC machines, each of which supports some subset of the ANSI standard. The ANSI capabilities of Heath/Zenith computer and terminal equipment also provide some degree of compatibility with the popular Digital Equipment Corporation (DEC) VT-100 and VT-200 series of computer terminals. Furthermore, the ANSI interface to the new H/Z-100 PC series of computers provides a much needed easy-to-use method for screen control.

In this article, I'll discuss the ANSI standard as it applies to the Heath/Zenith computers and describe how to use the ANSI capabilities of these machines. The discussion will focus in particular on the H/Z-100 computer, since the ANSI capabilities of that machine are largely undocumented. Discussion of the H/Z-100 PC in this article is not drawn from first hand experience, but is based on the Zenith documentation and my experience with the IBM-PC and PC compatible computers.

Throughout the remainder of this article, I'll use the term "Z-100" to refer to the Heath and Zenith versions of the model 110, 120 and related computers, and the term "Z-100 PC" to refer to the IBM compatible Heath and Zenith versions of the model 150 and 160 and related computers. This is consistent with Zenith's terminology in its documentation.

The ANSI Standard

The need for a standard for extended control sequences is a relatively recent one in computer history. The standard ASCII character set contains 33 control characters, which were originally designed to provide all of the control functions needed to operate early printers and teletype machines. As more versatile display devices were developed, there were too many functions to be controlled by just these 33 characters, and it became necessary to devise control sequences consisting of more than one character. The convention chosen was that these extended sequences would consist of an ASCII "escape" character followed by one or more of the printable ASCII characters. Such sequences are generally referred to as "escape sequences". The escape character is one of the 33 non-printable ASCII control characters, namely the 27th character, and appears in BASIC programs as CHR$(27).

One of the first sets of escape sequences to become somewhat of an industry standard was that defined by DEC for it's model VT-52 computer terminal. These sequences were adopted by Heath in the H-19 terminal and H-89 computer, and later in the Z-100 computer. Each escape sequence performs a well defined function, for example when the two character sequence <ESC> E is received, the screen is cleared and the cursor is returned to the upper left corner ("home"). Similarly <ESC> C moves the cursor forward, <ESC> M deletes the current line, etc.

The VT-52 and H-19/H-89 terminals also support operation under a second set of escape sequences, referred to as the ANSI control sequences. These machines enter this ANSI mode of operation when they receive <ESC> < (escape followed by a left angle bracket). The computer industry has been moving toward use of the ANSI control sequences, for example, the newer generations of DEC terminals (the VT-100 and VT-200 series) use the ANSI sequences for screen control.

The primary document which defines the ANSI control sequences is _ANSI X3.64-1979,_ entitled "American National Standard Additional Controls for Use with American National Standard Code for Information Interchange," (American National Standards Institute, Inc., 1430 Broadway, New York, NY 10018). The report is 60 pages long (plus appendices) and costs $17.00.

In general, ANSI escape sequences are of the form ESCape, followed by a left square bracket ([), followed by zero, one or more numeric parameters, and terminated by a printable character, usually a letter. Numeric parameters are usually optional and provide information on screen location, repeat counts, or mode selection as we'll see.

Table 1 shows some of the most useful of the ANSI defined escape sequences. The sequences available on Heath/Zenith machines are largely subsets of these, although some use is made of the sequences terminated by the letters p through z, which are reserved for private use. Exactly which sequences are available on which computers is described later in this paper, for now this table is intended simply to introduce you to some of the most useful sequences.

Table 1. Some Useful ANSI Escape Sequences

Mnemonic	Sequence	Default P	Interpretation
CUU	ESC [Pn A	1	Cursor up
CUD	ESC [Pn B	1	Cursor down
CUF	ESC [Pn C	1	Cursor forward
CUB	ESC [Pn D	1	Cursor backward
CUP	ESC [Pl; Pc H	1	Cursor positioning
ED	ESC [Ps J	0	Erase portion of display
EL	ESC [Ps K	0	Erase portion of line
IL	ESC [Pn L	1	Insert line
DL	ESC [Pn M	1	Delete line
DCH	ESC [Pn P	1	Delete character
DA	ESC [Pn c	0	Report device attributes
HVP	ESC [Pl;Pc f	1	Cursor positioning
SM	ESC [Ps h	none	Set mode
RM	ESC [Ps l	none	Reset mode
SGR	ESC [Ps m	0	Select graphics rendition
DSR	ESC [Ps n	0	Device status report

Note: ESC [p through ESC [z are reserved for private use. Some of these are used by Heath/Zenith as described later.

The escape sequences shown in this table can have numeric values which are denoted Pn, Pl, Pc or Ps. Pn denotes a repeat count which causes the associated action to be performed a repeated number of times. Pl and Pc are supplied in conjunction with direct on-screen cursor addressing and represent the line and column numbers, respectively. Ps is used with some sequences to designate a selection parameter, that is, a choice of one of several modes. Selection parameters are used with escape sequences that can perform a number of different actions, for example, the EL sequence (erase in line) can erase all of the current line, erase from the cursor to the end of the line, or erase from the beginning of the line to the cursor. The determination of which function is to be performed is based on the value supplied for Ps.

Since the notation can be a little confusing, let's look at a few examples of the above sequences. In these examples, the designation <ESC> means the single character escape. When each of these sequences is received, the terminal performs the indicated action:

<ESC>[1A Moves the cursor up one line on the screen
<ESC>[12D Moves the cursor back 12 print positions
<ESC>[4;6H Moves the cursor to line 4, column 6 on the screen
<ESC>[P Deletes one character at the cursor position

ANSI And The Heath/Zenith Computers

As I mentioned at the start of this article, each of the Heath/Zenith computers supports a subset of the ANSI escape sequences.

The Heath H-19 terminal and its close cousin, the H-89 computer, normally function in the "Heath" mode in which they respond to Heath's own set of escape sequences. The Heath escape sequences are a superset of the DEC VT-52 sequences. The H-19 and H-89 can be configured, however, to operate in "ANSI" mode by either placing section 5 of switch S402 in the ON position before powering up or by sending the sequence <ESC> < (escape followed by a left angle bracket).

The Z-100 computer recognizes almost all of the H-19 "Heath" mode escape sequences and adds a few more of its own. This emulation is programmed into the Z-100's ROM. It is no accident that such complete emulation of the H-19 and H-89 was provided in the Z-100, since this allows a high percentage of programs written for these older machines to be transferred to the Z-100 with a minimal amount of modification. One notable difference between the H-19 and the Z-100 is that the Z-100 ROM does not support the ANSI escape sequences directly, that is, <ESC> < does not place the machine in ANSI mode as it does on the H-19. The Z-100 can, however, be made to respond to the ANSI escape sequences through the use of a device driver, as we'll see in the next section.

The Z-100 PC is the latest computer introduction from Heath/Zenith. It is designed first and foremost to be "110% compatible" with the IBM Personal Computer. For this reason, it does not provide built-in support for any escape sequence control of the screen functions. Instead, it uses the IBM convention of requiring the user to call interrupt 10H in the ROM BIOS to perform all screen manipulation. Other than BASIC, most languages do not provide direct support of this capability, so unless you're experienced at interfacing higher languages to the BIOS, you may have concluded that there's no easy way to perform screen manipulation on the Z-100 PC. Fortunately, there is an easy way and it makes use of ANSI escape sequences.

The ANSI Device Driver

For the Z-100 and Z-100 PC computers, the way to cause the terminal to recognize the ANSI escape sequences is to install a device driver. Device drivers provide a convenient means of interfacing the operating system to physical devices, such as disk drives, modems, printers, and in this case, the console terminal. I should point out that this discussion of device drivers applies only to machines running under the MS-DOS operating system, version 2 or higher. Unfortunately, this rules out the use of the device driver approach to ANSI emulation under CP/M-85, CP/M-86 and Z-DOS.

Zenith supplies an ANSI console device driver with both the Z-100 and the Z-100 PC versions of MS-DOS 2. These drivers are simply special pieces of software which MS-DOS can be told to automatically install at boot time. Once installed, the drivers allow the console to recognize many of the ANSI escape sequences. These drivers function by intercepting MS-DOS function requests 1H, 6H, and 9H, thus they can be used with any program which uses the approved operating system calls for sending output to the console. Programs which circumvent the operating system by either writing directly to video RAM or outputting text via the ROM routines will not be able to take advantage of the ANSI sequences. Nearly all higher language compilers produce code which calls the operating system routines, thus programs written in higher languages should work fine with the ANSI driver.

Installing The Driver

The ANSI driver for the Z-100 PC should work fine as supplied. The driver supplied with the Z-100 currently has a few problems, which may have been solved in subsequent software releases from Zenith by the time this appears in print. I'll discuss these problems later, for now the following describes how to install the driver.

First locate the driver code on your MS-DOS 2 distribution disk. On the Z-100 distribution disk, I found my copy on disk #2 in the file \DEV\ANSICON.DVD. On the Z-100 PC disk, it should be called ANSI.SYS. Next, copy this file to the disk from which you plan to boot your system. If you are using a hard disk system, I recommend you create a subdirectory called \DEV and install this and any other device drivers there. Call the device driver file ANSI.SYS.

If you don't already have a configuration file, you must create one. This file must be called CONFIG.SYS and must reside in the root directory (not a subdirectory). The instructions in it are read by MS-DOS each time the system is booted. These instructions define how MS-DOS is to be configured, including the number of disk buffers to be allocated, the number of file buffers to be allocated, the names of device drivers to be loaded, and a number of other important system characteristics. Using Edlin or some other text editor install the following line in the CONFIG.SYS file:

```
DEVICE = \DEV\ANSI.SYS
```

or if the ANSI.SYS file is in the root directory, simply:

```
DEVICE = ANSI.SYS
```

That's all there is to installing the driver. After rebooting the system, you will find that the console responds to the ANSI escape sequences. You do pay a small price in memory usage and console output speed when using the ANSI driver. The driver will reduce the amount of memory available for your application programs somewhat. Check the size of the ANSI.SYS file to get an idea of how much. In my experience, console output speed has been only slightly reduced.

Description	Sequence	H-19/89	Z-100	Z-100 PC
Cursor up	ESC [Pn A	X	X	X
Cursor down	ESC [Pn B	X	X	X
Cursor forward	ESC [Pn C	X	X	X
Cursor backward	ESC [Pn D	X	X	X
Position cursor	ESC [Pl;Pc H	X	X	X
Erase in display	ESC [Ps J			
Ps = 0	To end of page	X	X	
Ps = 1	From beginning of page	X	X	
Ps = 2	Entire page	X	X	X
Erase in line	ESC [Ps K			
Ps = 0	To end of line	X	X	X
Ps = 1	From beginning of line	X	X	
Ps = 2	Entire line	X	X	
Insert line	ESC [Pn L	X	X	
Delete line	ESC [Pn M	X	X	
Delete character	ESC [Pn P	X	X	
Device attrib.	ESC [c		X	
Position cursor	ESC [Pl;Pc f	X	X	X
Set mode	ESC [Ps h			
Reset mode	ESC [Ps l			
Ps = 2	Enable/Disable keyboard	X		
Ps = 4	Insert mode on/off	X		
Ps = 20	Auto LF on CR	X		
Graphics mode	ESC [Ps m			
Ps = 0	Reverse video off	X	X	
Ps = 0	All attributes off			X
Ps = 1	High intensity mode			X
Ps = 4	Underscore characters			X
Ps = 5	Blink characters			X
Ps = 7	Enable reverse video	X	X	X
Ps = 8	Invisible characters			X
Ps = 10	Enter graphics mode	X	X	
Ps = 11	Exit graphics mode	X	X	
Ps = 30	Black foreground			X
Ps = 31	Red foreground			X
Ps = 32	Green foreground			X
Ps = 33	Yellow foreground			X
Ps = 34	Blue foreground			X
Ps = 35	Magenta foreground			X
Ps = 36	Cyan foreground			X
Ps = 37	White foreground			X
Ps = 40	Black background			X
Ps = 41	Red background			X
Ps = 42	Green background			X
Ps = 43	Yellow background			X
Ps = 44	Blue background			X
Ps = 45	Magenta background			X
Ps = 46	Cyan background			X
Ps = 47	White background			X
Device status	ESC [Ps n			
Ps = 6	Cursor position report	X		X
Private Use Sequences				
Set special	ESC [? Ps h			
Clear special	ESC [? Ps l			
Ps = 0	40 × 25 monochrome			X
Ps = 1	40 × 25 color			X
Ps = 2	80 × 25 monochrome			X
Ps = 2	Enter Heath mode	X		
Ps = 3	80 × 25 color			X
Ps = 4	320 × 200 color			X
Ps = 5	320 × 200 monochrome			X
Ps = 5	Reverse video		X	
Ps = 6	640 × 200 monochrome			X
Ps = 7	End of line wrap	X	X	X
Ps = 8	Auto key repeat		X	
Set special	ESC [> Ps h			
Clear special	ESC [> Ps l			
Ps = 1	25th line	X		
Ps = 2	Key click	X		
Ps = 3	Hold screen	X		
Ps = 4	Block cursor	X		
Ps = 5	Cursor existence	X		
Ps = 6	Keypad shifted	X		
Ps = 7	Alternate keypad	X		
Ps = 8	Auto LF on CR	X		
Ps = 9	Auto CR on LF	X		
Reassign keys	ESC [Ps p			
Ps = (see text)				X
Transmit screen	ESC [p	X		
Transmit 25th ln	ESC [q	X		
Modify Baud rate	ESC [Ps r			
Ps = 1	110 baud	X		
Ps = 2	150 baud	X		
Ps = 3	300 baud	X		
Ps = 4	600 baud	X		
Ps = 5	1200 baud	X		
Ps = 6	1800 baud	X		
Ps = 7	2000 baud	X		
Ps = 8	2400 baud	X		
Ps = 9	3600 baud	X		
Ps = 10	4800 baud	X		
Ps = 11	7200 baud	X		
Ps = 12	9600 baud	X		
Save cursor pos.	ESC [s	X	X	X
Restore cursor	ESC [u	X	X	X
Reset machine	ESC [z	X		
Alt. keypad on	ESC =	X	X	
Alt. keypad off	ESC >	X	X	

Table 2 — ANSI sequences recognized by Heath/Zenith machines. An X in a given column indicates that that machine recognizes the associated sequence.

Table 2 summarizes the ANSI sequences which are recognized by the Z100 and Z100 PC ANSI drivers, as well as the sequences recognized by the H-19/H-89 when it is in ANSI mode. The second part of the table shows which of the "private use" sequences have been implemented on each machine. Where selection parameters (Ps) are required, the different options are described. As you may notice, there are unfortunately a few differences in how some escape sequences are interpreted on the different machines.

A Few Problems With The Z-100 Version

As I mentioned, there are a few problems with the ANSI device driver supplied with the MS-DOS 2 software for the Z-100. First of all, the driver software contains a few bugs, and second, it is not configured for use as CON: (the console output device). I'll tackle these problems one at a time, but first let me point out that I'm referring to the version of ANSICON.DVD dated 4-04-84, 8:08a, 957 bytes in length. If your version is different, it may be a later release and the following patches may not apply.

The ANSI driver operates in one of four states at any one time. I'll number these states 0, 1, 2 and 3. In state 0, it is processing characters normally, waiting for an ESCape. In state 1, it has received an ESCape and is waiting for the next character. In state 2, it has received part of an unimplemented ANSI escape sequence, and is waiting for the next character (which it will throw away). In state 3, it has received the two character sequence, <ESC> [, and is waiting for the parameters and/or terminating character associated with the sequence. When processing of an escape sequence is completed, the driver should always return to state 0.

The first bug is in the code which handles the <ESC> > (Alt. keypad on) and <ESC> = (Alt. keypad off) sequences. Upon completion of these sequences, the driver returns to state 1 instead of state 0. This means that subsequent characters may be treated as part of an escape sequence, instead of as printable characters. The second bug is in the portion of code which handles sequences with multiple parameters (e.g. the Position Cursor sequence). The code uses the carry bit to indicate success or failure of various routines, but does not always set the bit properly. The third bug is in the initialization portion of the driver's interrupt routine. The DS and DX registers are modified, but are never saved on the stack.

You can patch the file ANSICON.DVD using the Debug utility supplied with MS-DOS. The following is the appropriate Debug input:

```
NANSICON.DVD
L Ø
A 3BD
CALL Ø1D9
JMP Ø167
MOV BYTE PTR [DI],Ø
RET
PUSH AX
PUSH BX
PUSH CX
PUSH DX
PUSH DS
JMP 42
POP DS
POP DX
POP CX
POP BX
POP AX
JMP 77
(hit the Return key)
A 3F
JMP Ø3C7
(hit the Return key)
A 74
JMP Ø3CF
(hit the Return key)
A 186
CLC
JMP Ø3C3
(hit the Return key)
A 2E7
JMP Ø3BD
(hit the Return key)
A 2DD
JMP Ø3BD
(hit the Return key)
A 7D
MOV DX,Ø3D7
(hit the Return key)
R CX
Ø3D7
W Ø
Q
```

After you make these patches and install the driver as described earlier in this paper, you will have a device called "ANSI", which you may treat as you would any other device (e.g. CON, PRN, AUX). In general, devices may be opened, read from, written to and closed just like files. When you write to ANSI, the output will appear on the console screen. However, ANSI escape sequences will be interpreted and processed properly.

You may use the redirected input and output features of MS-DOS 2 to allow any program to read from and write to the ANSI driver. If you had a program called MYPROG, which produces output containing ANSI sequences, you could invoke the program by typing

```
MYPROG >ANSI
```

and the output would be sent to the ANSI console driver. The advantage of using redirected I/O is that you write your program as if it is sending output to the standard output device (i.e. just use normal write or print statements), and then redirect the output at the time you run the program.

You may want to set up the Z-100 ANSI driver as a replacement for the console driver (CON:). This is the way the Z-100 PC version is set up. If you do this, you will no longer be able to use the Heath escape sequences to control the screen, since the ANSI driver does not interpret them properly. This can be a problem, since many programs for the Z-100, such as full-screen editors and spreadsheet programs make use of the Heath escape sequences. Nevertheless, the following Debug input will patch the ANSI driver to be a replacement for the console driver:

```
NANSICON.DVD
L Ø
E 4 Ø3
E A 43 4F 4E 20
W Ø
Q
```

Now, after you install the driver, you can write to it by simply writing to the console or to the device CON:.

Using The ANSI Escape Sequences

From a programming point of view, you make use of the ANSI escape sequences in essentially the same way you use the Heath escape sequences on the H-19/89 and Z-100. A number articles describing the use of escape sequences for screen control have appeared in the past in REMark and elsewhere, so I won't say much about the subject here. Simply stated, you need only send the appropriate sequences of characters to the console using your programming language's standard output statements (PRINT, WRITE, Writeln, printf, etc.).

The following are a few examples showing how to clear the console screen using the <ESC>[2J sequence:

```
BASIC:          PRINT CHR$(27);"[2J";

Pascal:         write(chr(27),'[2J');

'C':            printf("\033[2J");

Assembler:      ESC       EQU       27
                CLRSCN    DB        ESC,'[2J','$'
                          .
                          .
                          LEA       DX,CLRSCN   ; point to string
                          MOV       AH,09       ; request print
                          INT       21H         ; Call MS-DOS
                          .
                          .

Fortran:              IESC = 27
                      WRITE(*,1000) IESC
                1000  FORMAT(A1,'[2J')
```

If you're going to be making much use of the ANSI escape sequences in your programming, it is best to set up a series of subroutines to perform commonly needed functions, such as clear screen, move cursor, reverse video, etc.

Other Useful Things To Know

One of the useful features of the ANSI device driver for the Z-100 PC is that it has a built-in key remapping capability. This is not a part of the ANSI standard. The remapping capability lets you program any key (including the function keys) to produce any sequence of one or more characters. For example, you can program one of the function keys to produce the string "DIR<CR>" (where <CR> represents a carriage return). Whenever that function key is pressed in response to the MS-DOS prompt, a directory listing will be produced. You can also use this capability to effectively rearrange the keys on the keyboard by reprogramming which keys produce which characters.

The sequence used to reprogram keys is:

```
  <ESC>[n;n;...np
or
  <ESC>["string";p
or
  <ESC>[n;"string";n;n;"string";np
etc.
```

Here are a few examples taken from the MS-DOS Programmer's Utility Pack documentation:

* Reassign the Q and q key to the A and a key (and vice versa):

```
<ESC>[65;81P                A becomes Q
<ESC>[97;113P               a becomes q
<ESC>[81;65p                Q becomes A
<ESC>[113;97p               q becomes a
```

* Reassign the F10 key to the DIR command followed by carriage return.

```
<ESC>[0;68;"dir";13p
```

The 0;68 is the extended ASCII code for the F10 key. The BASIC manual contains a list of all of the extended codes for the various function and utility keys on the Z-100 PC's keyboard.

Another fun and useful application of the ANSI capabilities under MS-DOS 2 is to change the MS-DOS prompt using the PROMPT command. The PROMPT command lets you change the prompt to any string of printable characters. In addition, it allows you to put certain special characters and strings in the prompt. When setting the prompt, you specify these special characters and strings using letters prefixed by a '$'. For example, $e represents the escape character, $t represents the current time of day, $p represents the current subdirectory as set by the CD command, and $g represents the '>' character.

If you've installed the ANSI driver as the console device (CON:), you might try setting the MS-DOS prompt as follows:

```
PROMPT $e[s$e[1;65H$t$e[u$p$g
```

This will change the MS-DOS prompt so that it prints the time of day in the upper right corner of the screen, then returns the cursor to where it was and prints the current subdirectory followed by a '>'.

With a little thought, you'll find there are lots of ways to make use of the ANSI escape sequences to make programming easier and to make using your computer more enjoyable.

Introducing The ZP–150 PC

Jack Frank, Director
Application Software Engineering
Zenith Data Systems Corporation

Greetings from Zenith Data Systems. Let me introduce myself. I am Jack Frank, Director of Applications Software Engineering for Zenith Data Systems Corporation. My current assignments include the directing of the managers, programmers, and writers who acquire, license, and development application software for use on Zenith and Heath computers. I have been with ZDS since the early days (if August 1981 is early enough). I helped get products like CP/M–80 versions of Microstat, Multiplan, and even Pearl 3 out for your H–89s. I must admit that I wrote most of the Z–100 Demo program (also known as Z–Chart). Rumor has it that I know more about Microsoft Word than Microsoft, but that's another story. Enough about my work — I'm writing this article about my office.

Like so many of you, at heart I am first and foremost a programmer. The requirements of this job, however, have made me into a manager. As a manager, I deal with the real key component of any computer system — DATA. Tons of it. If I leave my Chicago office for one day, I return to more than 64k of new or revised data that I receive via Z–Mail. I am a true end–user of microcomputers. I have so much data that I have computerized about 80%.

My office has always contained at least two computers. (I know how Jerry Pournelle works — a lot of people need two, if not three, computers when so much of their business is contained 'in' them.) My first computer is always in the middle of doing something. My second computer contains documents I want handy for a phone call I know I might get at anytime today, or for a spreadsheet that I want to think about during the day and modify the instant I get 'a great thought'.

It was a sad day when my 'second' computer (Z–89 w/Z–39 drives...) was donated to a worthy cause — though replacing it with a Z–150 w/11 megs, 8087, 640k ram, Z–319 video card, and battery date/clock card 'system' really made the change tolerable.

As some of you readers know by now, Heath and ZDS have announced some new products. What you may not know is that my old Z–100 (ZS–100 speed mod, 448k, 11 megs...) has been retired. I have replaced my Z–100 with a ZP–150 portable computer. But that's not all that's in my office at the moment; so I would like to, with your assistance, clear out some of the things that led me to replace my Z–100 with a ZP–150 portable computer (I'll use PC for portable computer).

First, note that I am taking the Z–100 home so I can work around the clock (good justification to management). As I walk you around my office, we first come across an HP–110 PC, one that I have carried around with me for seven long (and back–breaking) months. It weighs nine pounds. We are getting rid of this one. Besides reducing my PC (personal carrying) by 25%, the ZP–150 gives me a word processor that has full screen editing, full document/paragraph/sentence/word editing, as well as compatibility with my desktop Word (things HP never dreamed of). Oh, I might miss having 1–2–3 in ROM for a while, but considering HP's price, there is no doubt in my mind that I spent $495 on another copy of 1–2–3 that I really did not need. The ZP–150's PLAN has every feature that I need for road or second computer work.

As I remove the HP from my office, I should also like to point out that a 9–pin plug is hard for me to consider a 'standard RS–232' port. And that HP–IL plug in back — too bad it wasn't a 'delete for credit' option. That TERMINAL program must have been written before 8080 CPS — I could have written a better communication program if the HP has the ZP–150's BASIC. Good, one down.

Over here is my boss' old Radio Shack 100. He loved it. Can you believe that his new computer was purchased out of MY BUDGET? (I guess when your boss can start thinking in 80 column mode, as well as do business with 16–line spreadsheets and calendars, it's a good ZDS investment.) There are a lot of these machines out there. His old one will have lots of company — you know how 8–bit computers hang out together. Lacking pop–up applications, as well as a data base program and a spreadsheet didn't help the Radio Shack 100, either. And running out of memory when you buy these additional programs just didn't cut it.

We did consider replacing the Radio Shack 100 with a Radio Shack 200. I have been told that sometimes people really get used to 40–column screens — look at all the Apple II users without the best selling 80 Column cards (Where?). Also, the price — almost $1000.00 and still getting an 8–bit version of BASIC that's weaker than MBASIC 4.7? And only 24K, at that price? This article alone would have used up 60% of it's memory. I think you can imagine how many phone numbers I could afford to have in my computer phone book on that machine.

Back here is my DG/1. Actually, it hasn't been mine for a while. I gave it to one of my staff members as a sort of present/gag gift. I carried it on the road, once. With the DG/1, 10 disks (I believe in backups), the AC power supply, the battery charger, and a great carrying case (bigger than two bread boxes), I had to check my luggage. That was the first time in seven years that I have checked my luggage on a business trip.

As Director of Application Software, I hailed Lotus Development Corp's announcement that your copy of 1-2-3 could be used on any of your computers — as long as it was not done concurrently. That is great progress in software licensing. I have bought my Z-150 PC compatible version of 1-2-3 and could now use it on any of my Zenith PC compatible machines, except my DG/1. Could I buy my fourth version of 1-2-3 in just 1-1/2 years (Z-100, Z-150, HP-110 and now DG/1)? Would you buy $1,980's worth of Lotus Software?

I have absolutely nothing against 3-1/2" drives. (I also have a Macintosh at home for my six-year-old son). But, what is the 3-1/2" disk format? What is the CP/M-80 standard format? Someday, someone will set THE 3-1/2" disk format (probably a company with initials not used in this article...) and software will be available in MS-DOS 5-1/2" standard and MS-DOS 3-1/2" standard, but today this situation is simply expensive.

Have you seen this Olivetti? Or this NEC 8201? Well if you haven't, just reading this far will have given you a list of shared problems. Since these machines are almost identical to the Radio Shack, you have a good overview of their 'family' specifications. I even have a WorkSlate over here, too. It was deep in the old machine room and I think at one time we had the power pack, but we saw no need for it. As I take all three of these out of the office in one big stack, I can finally see my chair — that's where I generally find my lap.

Well, that clears the office. I know I haven't owned, tested, or pounded on every portable computer out there. I couldn't bring myself to buying a Sord computer. I have tried them at trade shows and if 'kicking the tires' of this machine is an indication of anything, then typing on a Sord (or running any of the software I have seen on them for five minutes) may have been enough to justify my fiscal prudence.

I am not known for having a clean office, but currently it is cleaner than any of my programmer's offices or cubes. This is my two-computer office. My Z-150 is my work horse and my new ZP-150 is my constant companion. Before I draw you into the details of these machines, I have one more (if you can believe that) point to make. Don't knock 'em till you've tried 'em.

If you do not now own a portable, don't write off the usefulness or the market for these machines. I have talked to many lap owners. I even read the 'lap' magazines. These machines are not fads or the tools of writers. These machines are going to grab a large percent of the desktop microcomputer market — as second computers. Lap computers already account for a growing number of first computers. And I believe that Heath and ZDS have a machine that can show thousands of computer users how lap computers can be justified in either role.

What does the ZP-150 offer that makes me say all of this? Thought you'd never ask. The hardware has all the features you would expect. RS-232 port (25 pins, the RIGHT sex), parallel printer port, reset switch, direct-connect internal 300 baud modem, speaker, volume control (my favorite), numeric keypad (old keypunch style), real feeling keyboard, and an 80 × 16 line LCD. Sorry, batteries not included.

But, the key to this machine is the software (remember my job title). The ZP-150 contains a new Microsoft operating system, several new Microsoft applications, as well as some old reliables, and software design concepts that exceed what is available under MS-DOS from any of the well known application companies.

The ZP's software is known as Microsoft's WORKS. Heath is first with this one. Bill Gates, Microsoft's President, considers the ZP-150 the launching machine for this piece of software. Microsoft's WORKS consists of major subsets of Microsoft WORD, PLAN and PC-BASIC. It also contains Telcom, Calendar and File — three new Microsoft products. Add in HH-DOS (the operating system — or environment in this case), the pop-up calculator, dialer and alarm clock (complete with snooze button!), and you see that the ZP-150 contains TEN new Microsoft products. This machine contains about 224k worth of new Microsoft software! Software that has had over 15 person years of testing by Heath, ZDS, and Microsoft.

First, let me cover the software design concepts involved in the ZP-150. All programs use the same commands, function keys, and menu structure. If you learn to use the editing keys, the menu and the method for entering/exiting WORD, then you know how to edit data in a data base, access the Calendar menu, and load/exit all the other applications. For example, the F6 function key always extends — extends a spreadsheet range, extends the marking of a text block, extends the number of records selected in a data base.

Besides applications that have a common user interface (to coin a phrase), all the applications use a common scrap for data interchange. (For the users of the Macintosh, this is your clipboard.) Put a paragraph received during a communications session into the scrap and it can be moved into a WORD document as easy as 1-2-3.

All the applications can be temporarily interrupted so you can use the pop-ups. Quick — how much is .75% of 945? Have a phone connected to your ZP-150? Bring up the dialer window, enter a name that is stored in a phone book data base created by FILE, and the number will be dialed in an instant. Pick up the phone and while it rings, pop up the calculator and tell accounting that the 7.0875% variance isn't so bad.

I stated that we have some subsets of existing Microsoft products (but I don't consider that an issue to avoid). WORKS' PC-BASIC is both a subset and superset. Does your version of desktop BASIC have commands for accessing data base files created by a data base program? Oh, you only have random and sequence files? You're still using LSET, MKI and CHR$(34) to play with files? Can't your BASIC simply call a data base program and ask it to fill an array with all the records containing zip codes <60601? WORKS can.

Of course you have PAINT, OPEN "COM:", CIRCLE, LSET, MKI, PLAY, RND, and my favorite, REM. Just remember that the ZP-150 doesn't have joy sticks or color and that peeking and poking to ROM isn't too useful. Don't forget that your new BASIC has ON RESTART (remember, WORKS software can be interrupted, but resumed logically!). And shouldn't all versions of BASIC have all math functions and operations in double precision?

To be fair, I must state that WORD doesn't have style sheets and glossaries, but when I finalize my document on desktop Word (which is how this article was written), I will then use ZDS' official style sheet and my personal glossary. Plan isn't going to consolidate external sheets on a lap computer, but it can be used to prepare the external sheets at branch offices all over the world and WORKS can then modem those files to desktop Multiplan for final processing.

A few comments on the new applications. Perhaps the best new Microsoft product included is the Telcom program. This program should be available for desktops. As a telecommunication program, it offers a lot:

* VT-52 emulation 13
* HOST mode for remote control of the ZP-150 via the phone lines
* Scripts for those frequent callers
* AFTER the fact saving to memory (oops, forgot to save this message)
* Multiple phone books

So you can see, this product requires it's own article.

FILE? File is fairly simple. It is similar to creating data bases under a spreadsheet program. Rows and Columns. Any bonuses? Plenty, if you consider boiler plate formatting (i.e., ###-###-#### or (###)-###-####), relational linking to another data base, and complete control from BASIC as pluses.

CALENDAR. It won't let you forget. This program is not only a full feature scheduler and alarm clock, it is a TO DO LIST builder. If you enter every task you are asked to perform, any task you do not complete will haunt you every single day until it's done (like your boss). Coupled with the alarm, prioritizing of tasks, long descriptions, and the WORKS ability to move your appointments or "to do" list to WORD for status reports, you have a personal organizer that is better organized than you have ever been.

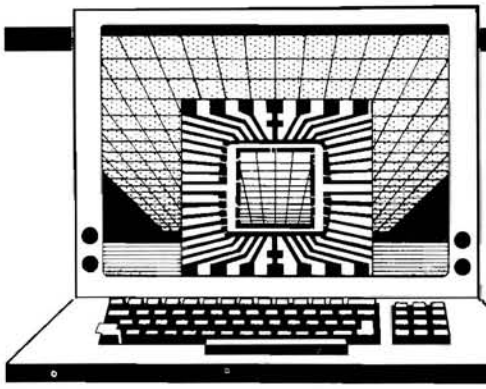
The final word on the software is operating system. There is no A> prompt. The system manager, as it is called, shows you an application on the left-hand side of the screen and all of that application's files on the right. Move the cursor to a file name, press RETURN, and the proper application is automatically brought up with that data file. Just like selecting a Macintosh icon.

Want your computer to wake up at 1:00 am to call a bulletin board when the phone rates are low and the system may not be busy? Set the wake up command to rise and shine at 1:00 am and automatically enter Telcom, run a script, grab the latest news and logoff. Meanwhile, you will blissfully sleep through it and read all about it in the morning.

One last word on the operating system — All the WORKS software runs in ROM. When you 'load' BASIC, you really aren't loading it. It is running in ROM. The 32k in the ZP-150 is yours. The operating system doesn't need your memory. Have you used up all your memory? You can still load Telcom and xmodem any old files to your desktop. You can still run your pop-ups. And of course, you can still run the delete command.

Well, thanks for stopping by. This technology transfer has been enjoyable. I think we both have gained a lot out of this visit. My staff has been given enough incompatible, over-priced, strangely designed and old lap computers to clutter all of their offices. I, on the other hand, have a clean office with two of the best microcomputers in the world — my full blown Z-150 and my very compatible and logical ZP-150. And you? Well, I hope you have gained the knowledge to do one of two things.

First, if you don't have a lap computer, you now have about 100 reasons to run to your Heath/Zenith Computers and Electronics Center and buy a ZP-150 (and you'll be able to run home with the box in hand!). Or, if you made one mistake, a reason to give (sell) your old lap computer to a former friend, relative (distant), or staff member so that you personally can have nothing but the best.



Inside MBASIC Program Files

John C. Harper
NASA - ASCENSION
Patrick AFB, FL 32925

Reference: Inside Microsoft BASIC (MBASIC), REMark, August 1984.

This information is applicable to many Disk Operating Systems. The programs have been run under CP/M, ZDOS, MSDOS and TURBODOS. To make them work, you must have a disk version of Microsoft BASIC that supports random access files. These include most MBASIC versions, ZBASIC, GW-BASIC, BASICA, PC-BASIC and others. The programs may require slight modification if your system differs from my CP/M 2.2, MBASIC 5.2 system.

In the referenced article, D.D. Dodgen presented a useful cross-reference utility for MBASIC programmers. For those of you having trouble making the programs run on your systems, I submit the following information. For you HDOS/MBASIC programmers, I expand on the information presented in the Dodgen article.

Where Is That Program?

The Dodgen article gives location 28761 as the starting address of all MBASIC programs under HDOS. MBASIC 5.2 under CP/M 2.2 running on an H8 with (only) hard sectored 5.25 inch disk drives, loads programs into memory beginning at address 25095. (NOTE: All my numbers are decimal, the MBASIC default radix.) If you have a system identical to mine, this number is all you need to make the published programs work. If your hardware differs from mine, CAL.BAS (Listing 1) will determine your system's magic number.

CAL.BAS

This program will look at the MBASIC disk file of itself, determine where your version of MBASIC will load it for execution, and verify that location by PEEKing itself in memory. This is possible because the program file on disk contains an exact duplicate of the MBASIC program in memory.

The program file in memory (and on disk) is a Linked Sequential File. Every line of program starts with the two byte memory address of the next line. The first two bytes of the first line of a program will contain this LINK to the second line. Given the address of the second line, subtracting the length of the first line will give you the address of the start of program area on your system.

When you SAVE a program without the 'A' (ASCII) option, MBASIC writes a copy of program memory to the disk. MBASIC adds two bytes to this program file, one at the beginning (File-Type token) and another at the end (End-of-File character, ASCII 26). The file-type token is not moved into memory with the program when it is LOADed from disk.

CAL.BAS

```
10 REM
20 'Insure that above line is FIRST line of this program!
    Insure that you hit RETURN immediately after the M in
    REM - NO SPACES!
30 'You may omit these REMarks (lines 20, 30 & 40).
    After typing program, type SAVE"CAL and RETURN. No
    A or P option.
40 'Make any syntax changes required by your implementation
    but don't change line 10
50 OPEN"R",1,"CAL.BAS",3
60 FIELD#1,1 AS TYPE$,2 AS ADD$
70 GET#1
80 ADD=ASC(LEFT$(ADD$,1))+ASC(RIGHT$(ADD$,1))*256-6
90 PRINT:PRINT" STARTING ADDRESS OF AN MBASIC PROGRAM
     ON YOUR MACHINE IS";ADD
100 PRINT"      MEMORY VERIFICATION";
110 PRINT PEEK(ADD)+PEEK(ADD+1)*256-6
```

Listing 1. CAL.BAS — A program that finds the start of program space.

Caution: This program will only work on the system on which MBASIC first SAVEs it. You cannot use PIP or COPY to move it from CP/M to MSDOS or H89 to SuperBrain. When MBASIC SAVEs it to your system's disk, it creates a program file that is unique to your particular hardware/software system. The LINKS will be different. ASCII transfers are permissible, but CAL.BAS expects to find itself on the default drive in tokenized format.

There are two File-Type tokens. ASCII 255 indicates a normal (SAVE "filename") program and ASCII 254 indicates a protected (SAVE "filename",P) program. I will only be concerned with file-type 255.

MBASIC Tokens

The tables in the Dodgen article are for the MBASIC version run-

CALYPSO.BAS

```
10000 '                        CALYPSO

10010 '                  Program File Explorer
10020 '
                             J.C. HARPER
10030 '             South Atlantic Computer Club (SACC)
10040 '
      INITIALIZE

10050 DEFINT A-Z
10060 PRINT CHR$(27)+"E":'H19 clear screen
10070 MYAD!=25095:'Start of program space on MY machine
10080 MAD!=MYAD!
10090 CHCK$=CHR$(0)+CHR$(0)+CHR$(26):'End of program
        sequence
10100 INPUT"WHAT FILE? ", FILE$
10110 IF INSTR(FILE$,".")=0 THEN FILE$=FILE$+".BAS"
10120 OPEN "R",1,FILE$,10
10130 FIELD#1,10 AS RCRD$
10140 GET#1:'Move 10 characters from disk to input buffer
        (RCRD$)
10150 WRK.BFR$=RCRD$:'move them to safety in work buffer
        (WRK.BFR$)
10160 FL.TP$=LEFT$(WRK.BFR$,1):'First character is file-type
10170 PRINT:PRINT TAB(25)"FILE TYPE TOKEN IS";ASC(FL.TP$):
        PRINT
10180 WRK.BFR$=MID$(WRK.BFR$,2):'Throw away type token
10190 '
      SCREEN LOOP

10200 FOR LOOP1=1 TO 20:'20 lines at a time
10210   IF EMPTY THEN 10230:'EMPTY = end of program file
10220   GET#1:WRK.BFR$=WRK.BFR$+RCRD$:'Fill work buffer
10230   SIZE=INSTR(WRK.BFR$,CHCK$):'Look for end of program
10240   IF SIZE <> 0
         THEN EMPTY=-1:SIZE=SIZE+2:
              WRK.BFR$=LEFT$(WRK.BFR$,SIZE)
         ELSE SIZE=19
10250   IF SIZE > 10 THEN CNTR=10 ELSE CNTR=SIZE
10260 '
      LINE LOOP

10270   PRINT USING"  #####      ";MAD!;
10280   FOR LOOP2=1 TO CNTR:'Max 10 characters/line
10290      PNTR$=MID$(WRK.BFR$,LOOP2,1)
10300      PRINT USING"  ###";ASC(PNTR$)
10310      IF ASC(PNTR$) < 32 OR ASC(PNTR$) > 126
            THEN HOLD$=HOLD$+"."
            ELSE HOLD$=HOLD$+PNTR$
10320   NEXT
10330   PRINT TAB(68);HOLD$:'Print the ASCII equivalent
          symbols
10340   HOLD$=""
10350   IF LEN(WRK.BFR$) > 10
         THEN WRK.BFR$=MID$(WRK.BFR$,11):MAD!=MAD!+10
         ELSE 10400
10360 NEXT:'Get another line
10370 MORE$=INPUT$(1)
10380 GOTO 10200:'Get another screen full
10390 '
      CLOSE FILE - END

10400 CLOSE
10410 MAD!=(MAD!-MYAD!)+LEN(WRK.BFR$)
10420 PRINT:PRINT TAB(20)"Program occupies";MAD!;
        "bytes of memory."
```

Listing 2. CALYPSO.BAS — Reads MBASIC Tokenized program files and displays them on CRT.

ning under HDOS. Different versions of MBASIC may have different tokens for the same BASIC commands. MBASIC 5.2 differs slightly from the HDOS version discussed in the referenced article. Rather than present tables for my system, the program CALYPSO.BAS (Listing 2) will assist you in making a table of MBASIC tokens for your particular system.

CALYPSO.BAS

This program reads a BASIC program file (or any file) from disk and displays it on your terminal. By writing short program segments and using CALYPSO to examine their disk files, you will be able to build a table of tokens for your system. An example program segment for this purpose might look like this:

```
10 REM REMARK TOKEN
20 'ALTERNATE REMARK TOKEN
30 PRINT"PRINT TOKEN"
40 ?"ALTERNATE PRINT TOKEN"
```

SAVE this segment under any filename (e.g. SAVE"TEST"). The CALYPSO output of this segment is shown in Figure 1.

```
                   FILE TYPE TOKEN IS 255

 25095  26  98  10   0 143  32  82  69  77  65  .b... REMA
 25105  82  75  32  84  79  75  69  78   0  56  RK TOKEN.8
 25115  98  20   0  58 143 219  65  76  84  69  b.:...ALTE
 25125  82  78  65  84  69  32  82  69  77  65  RNATE REMA
 25135  82  75  32  84  79  75  69  78   0  75  RK TOKEN.K
 25145  98  30   0 145  34  80  82  73  78  84  b..."PRINT
 25155  32  84  79  75  69  78  34   0 104  98   TOKEN".hb
 25165  40   0 145  34  65  76  84  69  82  78  (.."ALTERN
 25175  65  84  69  32  80  82  73  78  84  32  ATE PRINT
 25185  84  79  75  69  78  34   0   0   0  26  TOKEN"....

            Program occupies 100 bytes of memory.
```

Figure 1. Sample output of CALYPSO.BAS with TEST.BAS (see text) as input.

Using CALYPSO.BAS

RUN CAL.BAS and find the beginning of program space on your system. Change line 10070 of CALYPSO to reflect this value. Run CALYPSO.

CALYPSO will ask 'WHAT FILE?' Any legal filename may be entered. If you don't specify a file-extension, CALYPSO will assume you want '.BAS'. If you don't specify a disk drive, it will assume the default (usually A:) drive.

NOTE: MBASIC treats UPPER and LOWER case characters as DIFFERENT! It can create the files TEST.BAS and Test.BAS as two separate files. Unless you are aware of the problems this can cause your DOS, depress the 'CAPS LOCK' key. If you specify a non-existent (or misspelled) filename, CALYPSO will oblige by creating an empty file of that name and displaying it for you. You must then KILL the empty file to eliminate it from your disk.

CALYPSO will open the specified file and display the first character as the file-type token. The body of the display lists ten bytes (ASCII decimal value) of the source file. The number in the left column is the memory address of the first of these ten bytes. To the right, these same ten bytes are displayed as their equivalent ASCII character or '.', if unprintable.

The display will show up to 200 bytes of the source program. Hitting any key will scroll another 200 bytes into the display. CALYPSO will terminate when it reaches the end of the source program. Control C will terminate anywhere. The information in the Dodgen article should be sufficient to understand the CALYPSO output.

Change the program segment to reveal other MBASIC commands, reserved words or functions. Program segments don't have to DO anything. Just go through your MBASIC's list of Reserved Words, substituting them for the REM and PRINT commands in the above segment.

Watch out for single digit numeric constants! My version of MBASIC tokenizes them to one byte values. This saves 2 bytes of memory every time a number is less than 10. I leave it to you to discover how (or if) your version of MBASIC does this.

Cross-Reference-Generator

CRG.BAS (Listing 3) is a different version of the Cross-Reference utility presented in the referenced article. It uses the information presented here and in the Dodgen article.

This version does not require that it be MERGEd with the source program. It is a stand-alone program that can generate a cross-reference listing of an MBASIC Tokenized program file of any size, from any disk drive.

Again, if you don't understand how your DOS handles upper/lower case filenames, depress the 'CAPS LOCK' key.

This program uses H/Z-19 screen formatting characters (lines 10050–10060). Change them to suit your system. A printer is suggested, but not required for efficient use of the troubleshooting capabilities of a cross-reference program. If you don't have a printer, change the output routines for CRT output (lines 10870–11120) or answer 'N' or 'n' to the '-HARD COPY?' request.

Line 10100 contains the number 25095. This is the starting address of MBASIC program area on MY system. You must determine this value for YOUR system. You must change this number to have CRG.BAS run correctly on your system. CAL.BAS will tell

```
TARGET    REFERENCE        PROGRAM NAME - B:CALYPSO.BAS
 LINE       LINES
------    ---------
10200       10380
10230       10210
10400       10350

3 LINE REFERENCES AND 3 TARGET LINES IN 43 PROGRAM LINES.


VARIABLE   REFERENCE        PROGRAM NAME - B:CALYPSO.BAS
  NAME       LINES
--------   ---------
A          10050
CHCK$      10090 10230
CNTR       10250 10280
EMPTY      10210 10240
FILE$      10100 10110 10120
FL.TP$     10160 10170
HOLD$      10310 10330 10340
LOOP1      10200
LOOP2      10280 10290
MAD!       10080 10270 10350 10410 10420
MORE$      10370
MYAD!      10070 10080 10410
PNTR$      10290 10300 10310
RCRD$      10130 10150 10220
SIZE       10230 10240 10250
WRK.BFR$   10150 10160 10180 10220 10230 10240 10290
           10350 10410
Z          10050

17 VARIABLES USED 66 TIMES IN 1690 BYTES OF PROGRAM SPACE.
```

Figure 2. Cross-reference lists produced by CRG.BAS for CALYPSO.BAS.

CRG.BAS

```
10000 '            CROSS-REFERENCE GENERATOR
10010 '                  Version 2.0
10020 '                   J.C.Harper
10030 '       South Atlantic Computer Club (SACC)
10040 DEFINT A-Z:LA=0:PB=0:'     initialize
10050 EB$=CHR$(27):CK$=EB$+"E":EA$=EB$+"J":CG$=EB$+"x5":
        CI$=EB$+"y5"
10060 DEF FNAB$(RA,CH)=EB$+"Y"+CHR$(31+RA)+CHR$(31+CH)
10070 DEF FNUA!(PA$)=ASC(LEFT$(PA$,1))+ASC(RIGHT$(PA$,1))
        *256
10080 DIM LF$(200),VA$(200):
        '    line number & variable lists
10090 DEF FNSE$(X$)=RIGHT$(X$+LEFT$(X$,1),2)
10100 SA!=25095:SD!=SA!
10110 'make test strings
10120 CA$=CHR$(143)+CHR$(132)+CHR$(0)+CHR$(14)
10130 CB$=CHR$(15)+CHR$(11)+CHR$(28)+CHR$(29)+CHR$(31)
        +CHR$(34)
10140 CC$="ABCDEFGHIJKLMNOPQRSTUVWXYZ":CD$=".0123456789":
        CE$="%!#$"
10150 BA=1:CF$=" ALL AS BASE ":PRINT CK$
10160 'screen format
10170 PRINT FNAB$(3,25);"CROSS REFERENCE GENERATOR [2.0]":
        PRINT:PRINT:PRINT
10180 PRINT"Input file must be SAVE file format without
        'A' OR 'P' option."
10190 PRINT"If device is not specified, default is assumed."
10200 PRINT"If file type is not specified, .BAS is assumed.":
        PRINT:PRINT
10210 INPUT"Input FileName: ",RB$:IF RB$=""THEN END
10220 'open input file
10230 IF INSTR(RB$,".")>1
      THEN FA$=RB$:RB$=LEFT$(RB$,INSTR(RB$,".")-1)
      ELSE FA$=RB$+".BAS"
10240 OPEN"R",1,FA$:FIELD#1,128 AS IA$:GET#1
10250 IF LEFT$(IA$,1)<>CHR$(255)THEN PRINT
        "WRONG FILE FORMAT":END
10260 WB$=MID$(IA$,2):GET#1:PRINT FNAB$(6,1);EA$;CG$
10270 PRINT FNAB$(6,28);"PROGRAM NAME - ";FA$
10280 WA=LEN(WB$):WC$="":PB=1:IF WA<128 THEN WB$=WB$+IA$:
        GET#1
10290 AA$=LEFT$(WB$,2):
        IF AA$=CHR$(0)+CHR$(0)THEN 10830ELSE LC=LC+1
10300 LD-FNUA!(AA$)-SA!:BA=BA+LD:SA!=SA!+LD:
        LE$=MID$(WB$,3,2)
10310 PRINT FNAB$(9,30);"PROCESSING LINE";FNUA!(LE$)
10320 'get a line of BASIC
10330 WB$=MID$(WB$,5):WA=LEN(WB$):LD=LD-4
10340 IF WA<LD THEN WC$=WC$+WB$:WB$=IA$:GET#1:LD=LD-WA:
        WA=LEN(WB$):GOTO 10340
10350 WC$=WC$+LEFT$(WB$,LD):WB$=MID$(WB$,LD+1)
10360 'examine line one character at a time
10370 ON INSTR(CA$,MID$(WC$,PB,1))
        GOTO 10280,10280,10280,10490
10380 ON INSTR(CB$,MID$(WC$,PB,1))
        GOTO 10450,10440,10440,10440,10430,10420,10470
10390 IF INSTR(CC$,MID$(WC$,PB,1))>0 THEN 10620
10400 PB=PB+1:GOTO 10370:
        '       ignore this character - get another
10410 'skip numeric constants
10420 PB=PB+4:'           skip 8 byte number
10430 PB=PB+2:'           skip 4 byte number
10440 PB=PB+1:'           skip 2 byte number
10450 PB=PB+2:GOTO 10370:'       skip 1 byte number
10460 'skip string constants
10470 PB=INSTR(PB+1,WC$,CHR$(34)):
        IF PB=0 THEN 10280ELSE PB=PB+1:GOTO 10370
10480 'line reference
10490 LI$=MID$(WC$,PB+1,2):LH=LH+1:LA=0
10500 LA=LA+1:IF LI$=LEFT$(LF$(LA),2)THEN 10580
10510 IF FNSE$(LI$)<FNSE$(LEFT$(LF$(LA),2))THEN 10540
10520 IF LA>=HB THEN 10540 ELSE 10500
10530 'start another line reference list
10540 HB=HB+1:LA=HB:LF$(HB)-LI$+LE$
```

```
10550 LA=LA-1:IF FNSE$(LEFT$(LF$(LA),2))
        <FNSE$(LEFT$(LF$(LA+1),2))THEN 10600
10560 SWAP LF$(LA),LF$(LA+1):GOTO 10550
10570 'add to line reference list
10580 IF LE$=RIGHT$(LF$(LA),2)THEN 10600
10590 LF$(LA)=LF$(LA)+LE$
10600 PB=PB+3:GOTO 10370
10610 'variable names
10620 V=V+1:V$="":LA=Ø
10630 V$=V$+MID$(WC$,PB,1):PB=PB+1:
        IF INSTR(CC$+CD$,MID$(WC$,PB,1))>Ø THEN 10630
10640 IF INSTR(CE$,MID$(WC$,PB,1))>Ø
        THEN V$=V$+,MID$(WC$,PB,1):PB=PB+1
10650 IF INSTR(CF$," "+V$+" ")>Ø THEN V=V-1:GOTO 10370
10660 IF MID$(WC$,PB,1)="("THEN V$=V$+"(":PB=PB+1
10670 'is this variable on list
10680 IF HC=Ø THEN 10740
10690 LG=1:HA=HC+1
10700 SC=INT((LG+HA)/2):' binary search of variable list
10710 IF V$=LEFTS(VA$(SC),INSTR(VA$(SC)," ")-1)THEN LA=SC:
        GOTO 10790
10720 IF LEFT$(VA$(SC),INSTR(VA$(SC)," ")-1)<V$
      THEN IF LG=SC
           THEN 10740 ELSE LG=SC:GOTO 10700
      ELSE IF HA=SC THEN 10740 ELSE HA=SC:GOTO 10700
10730 'new variable name
10740 HC=HC+1:LA=HC:VA$(HC)=V$+" "+LE$
10750 LA=LA-1:IF HC=1 THEN 10810
10760 IF LEFT$(VA$(LA),INSTR(VA$(LA)," "))<V$THEN 10810
10770 SWAP VA$(LA),VA$(LA+1):GOTO 10750
10780 'old variable name
10790 IF RIGHT$(VA$(LA),2)=LE$THEN 10810
10800 VA$(LA)=VA$(LA)+LE$
10810 GOTO 10370
10820 'print CRT summary
10830 PRINT FNAB$(11,20);'LINE REFERENCES -";LH;TAB(30);
        "TARGET LINES -";HB
10840 PRINT FNAB$(13,25);HC;"VARIABLE USED";V;"TIMES"
10850 PRINT FNAB$(15,23);"PROGRAM OCCUPIES";BA;
        "BYTES OF MEMORY"
10860 'printer output
10870 PRINT FNAB$(17,23);EA$;
        "DO YOU WANT HARD COPY? (Y/N)";CI$;
10880 RB$=INPUT$(1):PRINT CG$:
        IF RB$="N"OR RB$="n"THEN 11170
10890 IF RB$<>"Y"AND RB$<>"y"THEN 10870
10900 PRINT FNAB$(17,23);EA$;"PRINTER IS NOT CONFIGURED":
        LPRINT
10910 PRINT FNAB$(17,23);EA$;"PRINTING TARGET LINE LIST"
10920 LPRINT
      "TARGET          REFERENCE          PROGRAM NAME - ";
      FA$
10930 LPRINT" LINE             LINES"
10940 LPRINT"------           ---------"
10950 FOR LA=1 TO HB:LPRINT FNUA!(LEFT$(LF$(LA),2));TAB(15)
10960 FOR LB=3 TO LEN(LF$(LA))STEP 2
10970 FB=FB+1:IF FB=11 THEN LPRINT:LPRINT TAB(15);:FB=1
10980 LPRINT USING"######";FNUA!(MID$(LF$(LA),LB,2));:
        NEXT:LPRINT
10990 LPRINT:FB=Ø:NEXT:LPRINT
11000 LPRINT LH;"LINE REFERENCES AND";HB;
        "TARGET LINES IN";LC;"PROGRAM LINES."
11010 LPRINT CHR$(12);
11020 PRINT FNAB$(17,23);EA$;"PRINTING VARIABLE LIST"
11030 LPRINT
      "VARIABLE        REFERENCE        PROGRAM NAME - ";
      FA$
11040 LPRINT"  NAME                LINES"
11050 LPRINT"--------           ---------"
11060 FOR LA=1 TO HC:
        LPRINT LEFT$(VA$(LA),INSTR(VA$(LA)," ")-1);
        TAB(19);
11070 FOR LB=INSTR(VA$(LA)," ")+1 TO LEN(VA$(LA))STEP 2
11080 FB=FB+1:IF FB=11 THEN LPRINT:LPRINT TAB(19);:FB=1
11090 LPRINT USING"######";FNUA!(MID$(VA$(LA),LB,2));
11100 NEXT:LPRINT:LPRINT:FB=Ø:NEXT:LPRINT
11110 LPRINT HC;"VARIABLE USED";V;"TIMES IN";BA;
        "BYTES OF PROGRAM SPACE."
11120 LPRINT CHR$(12);
11130 'make disk file
11140 PRINT FNAB$(17,23);EA$;"SAVE .CRF FILE? (Y/N)";CI$;
11150 RB$=INPUT$(1):PRINT CG$:
        IF RB$="N"OR RB$="n"THEN 11210
11160 IF RB$<>"Y"AND RB$<>"y"THEN 11150
11170 CLOSE:PRINT FNAB$(17,23);EA$;
        "WRITING .CRF FILE TO DISK"
11180 CJ$=LEFT$(FA$,INSTR(FA$,"."))+"CRF":OPEN"O",1,CJ$:
        PRINT#1,HB,HC,SD!
11190 FOR LA=1 TO HB:PRINT#1,FNUA!(LEFT$(LF$(LA),2)):
        NEXT:FOR LA=1 TO HC
11200 PRINT#1,LEFT$(VA$(LA),INSTR(VA$(LA)," ")-1):NEXT
11210 PRINT CI$:CLOSE:END
```

Listing 3. CRG.BAS — A cross-reference utility. Also a pre-processor for PC.BAS.

you this address.

Figure 2 is the output of CRG.BAS with CALYPSO.BAS as the input file. Compare this output with the input program listing. It will tell you where branching instructions are (Reference Lines) and where they jump to (Target Lines). It will also show every variable name used and the line number(s) in which it appears.

If you choose, CRG.BAS will write a disk file of target lines and variable names. This file will be used by a program presented later. It will have the same filename as the source file with the extension CRF (e.g. CALYPSO.CRF). This file will be written to the same disk drive that contains the source program.

On Smaller Programs

Kenneth Mortimer (BUGGIN' HUG, October 1984) says that in these days of cheap memory and disk storage there is no reason to attempt to keep programs short. I would agree, if long programs executed as fast as short programs. Another problem arises with a large system of programs, such as a business package. Long programs cause frequent and unnecessary disk changes. You should have as many programs and as much data as possible 'on-line'.

He won't like the way I have compressed CRG.BAS and PC.BAS, but the 'fat' versions of these two programs would have required a dedicated issue of REMark. I doubt the editor would consider page space as cheap. I consider these listings a reasonable compromise.

You can't please everyone, but consider the following.

A major problem of programming in BASIC is that one file is used for everything. The Source File (for the human) is also the Object File (the file the computer runs).

The best program for development/debugging/learning is not the most efficient for running. The requirements of programmer and computer are different.

For ease of programming, a program should:

1. Have short, single statement lines. Easier to change or replace.
2. Have lots of REMarks. To remember what's happening where,

when.

3. Use long, meaningful variable names. Easier to add new names — find & change old ones.
4. Use indented FOR, WHILE and IF-THEN statements. To help trace program flow.

For fast execution and efficient storage a program should:

1. Have long, multiple statement lines. Makes GOTO and GOSUB work faster.
2. Have NO remarks. REMarks take up a lot of space and REMark lines slow down a program.
3. Use short variable names. Variable names take up a lot of space.
4. Use NO unnecessary spaces, TABs or other formatting Characters. More wasted space.

The MBASIC Compiler is a good way to reconcile these different requirements, if you can afford it. An alternative method is to write/debug your programs in the big/slow (human) format and then rewrite/debug them in the smaller/faster (computer) format. PC.BAS can help with this alternative.

PC.BAS (Listing 4) requires two input files: First is the 'fat' program you have written and debugged; Second is the CRF file of that program created by CRG.BAS.

PC.BAS is formatted for the H19 terminal.

I will use CALYPSO.BAS (a human oriented program) in a trial run of PC.BAS.

Run CRG.BAS and answer 'CALYPSO' to the 'What file?' request. Include drive name if necessary (e.g. B:CALYPSO). Suit yourself at the '-HARD COPY?' request. If you choose a printed cross-reference listing, you will be asked 'SAVE CRF FILE?'. Answer 'Y' or 'y'. If you do not want a printed listing, CRG.BAS will automatically write the CRF file. The CRF file contains a list of target lines and variable names used in CALYPSO and will be written to the same disk as the CALYPSO source file. This file must be from your latest version of CALYPSO.

Run PC.BAS and answer 'CALYPSO' to the 'What file?' request. PC.BAS will expect to find CALYPSO.BAS and CALYPSO.CRF on the same drive. It will write a third file (CALYPSO.A) to that same disk. CALYPSO.A is LOGICALLY identical to CALYPSO.BAS, but occupies less space and runs faster.

What PC.BAS does:

1. Eliminates all end-of-line REMarks.
2. Eliminates REMark lines that are not targets of GOTO or GOSUB.
3. Eliminates TABs and CARRIAGE RETURN/LINE FEEDs not in strings.
4. Eliminates spaces not in strings.
5. Combines lines where possible.
6. Shortens variable names to two characters.

After concentrating a program, store the original (Source) file on a low cost disk in your archives.Use this file if you want to change the program. Then run CRG.BAS and PC.BAS on the new version to create a small/fast Object file for your working disk.

Some Surprises

Since PC.BAS operates on the Token file, it bypasses some of the limitations imposed by MBASIC.

PC.BAS can eliminate more spaces than you can when entering a

PC.BAS

```
10000 '          MBASIC PROGRAM CONCENTRATOR
10010 '                  Version 2.2
10020 '                  J.C.Harper
10030 '      South Atlantic Computer Club (SACC)
10040 'initialize
10050 DEFINT A-Z:OPTION BASE 1:PA=Ø:BC$="":BF$="":LD=Ø:
        EB$=CHR$(27)
10060 CI$=EB$+"E":EA$=EB$+"J":CJ$=EB$+"x5":CL$=EB$+"y5"
10070 DEF FNAC$(RB,CK)=EB$+"Y"+CHR$(31+RB)+CHR$(31+CK)
10080 DEF FNPD$(NE!)=CHR$(NE!-INT(NE!/256)
        +CHR$(INT(NE!/256))
10090 DEF FNUA!(PC$)=ASC(LEFT$(PC$,1))
        +ASC(RIGHT$(PC$,1))*256
10100 DEF FNGA(CA$,WA$,PG)=INSTR(CA$,MID$(WA$,PG,1))
10110 'make test strings
10120 CB$=CHR$(9)+CHR$(1Ø)+CHR$(32)+CHR$(Ø)
10130 CC$=CHR$(15)+CHR$(11)+CHR$(12)+CHR$(14)
        +CHR$(28)+CHR$(29)+CHR$(31)+CHR$(34)
10140 CD$="ABCDEFGHIJKLMNOPQRSTUVWXYZ":CE$=:.123456789Ø"
10150 CF$="%!#$":CG$=CHR$(143):CH$=CHR$(137)+CHR$(139):
        PRINT CI$
10160 'screen format
10170 PRINT FNAC$(3,25);
        "MBASIC PROGRAM CONCENTRATOR [2.2]":PRINT:
        PRINT:PRINT
10180 PRINT"Input file must be SAVE file format without
        'A' or 'P' option."
10190 PRINT"If device is not specified default is assumed."
10200 PRINT"If file type is not specified, .BAS is assumed."
10210 PRINT"FileName.CRF must have been created by
        CRG.BAS and be on same disk"
10220 PRINT"as input file.":PRINT:PRINT:
        INPUT"Input FileName: ",RA$
10230 IF RA$=""THEN END
10240 PRINT CJ$
10250 'make file names
10260 IF INSTR(RA$,".")>1
      THEN PF$=RA$:RA$=LEFT$(RA$,INSTR(RA$,".")-1)
      ELSE PF$=RA$+".BAS"
10270 CM$=RA$+".CRF":NC$=RA$+".A"
10280 'get program stats
10290 OPEN"I",1,CM$:INPUT#1,HA,HB,SA!:NA!=SA!:DIM LF(HA)
10300 'load line reference array
10310 FOR LD=1 TO HA:INPUT#1,LF(LD):NEXT
10320 'load variable list - make new variable names
10330 DIM VB$(HB,2):LA$="A":FOR LD=1 TO HB:
        INPUT#1,VB$(LD,1)
10340 TB$=VB$(LD,1):FA$=LEFT$(TB$,1):
        IF FA$<>LA$THEN RESTORE:LA$=FA$
10350 IF RIGHT$(TB$,1)=
        "("THEN PE=1:TB$=LEFT$(TB$,LEN(TB$)-1)ELSE PE=Ø
10360 IF INSTR(CF$,RIGHT$(TB$,1))>1
      THEN TD$=RIGHT$(TB$,1):TB$=LEFT$(TB$,LEN(TB$)-1)
      ELSE TD$=""
10370 IF LEN(TB$)<2 THEN 1Ø39Ø
10380 READ SB$:TB$=FA$+SB$
10390 IF TD$<>""THEN TB$=TB$+TD$
10400 IF PE=1 THEN TB$=TB$+"("
10410 VB$(LD,2)=TB$:NEXT:CLOSE
10420 'open input file
10430 OPEN"R",1,PF$:FIELD#1,128 AS BB$:GET#1
10440 IF LEFT$(BB$,1)<>CHR$(255)THEN PRINT
        "WRONG FILE FORMAT":END
10450 BE$=MID$(BB$,2):GET#1:PRINT FNAC$(6,1);EA$;
10460 PRINT FNAC$(6,28);"PROGRAM NAME - ";PF$:
        OPEN"O",2,NC$
10470 PRINT#2,CHR$(255);:'       start output file
10480 BD=LEN(BE$):BC$="":PA=1:
        IF BD<128 THEN BE$=BE$+BB$:GET#1
10490 AA$=LEFT$(BE$,2):IF AA$<>CHR$(Ø)+CHR$(Ø)THEN 1Ø52Ø
10500 IF LEN(BG$)>Ø THEN GOSUB 11Ø6Ø
10510 GOTO 111ØØ
10520 LB=FNUA!(AA$)-SA!:SA!=SA!+LB:LC$=MID$(BE$,3,2)
10530 LE=FNUA!(LC$):TA=Ø
10540 'target line?
```

```
10550 FOR LD=1 TO HA:IF LE=LF(LD)THEN TA=1:LD=HA
10560 NEXT
10570 PRINT FNAC$(9,30);"PROCESSING LINE";FJUA!(LC$)
10580 'get a line of BASIC
10590 BE$=MID$(BE$,5):BD=LEN(BE$):LB=LB-4
10600 IF BD<LB THEN BC$=BC$+BE$:BE$=BB$:GET#1:LB=LB-BD:
        BD=LEN(BE$):GOTO 10600
10610 BC$=BC$+LEFT$(BE$,LB):BE$=MID$(BE#,LB+1)
10620 'time saving pre-processor
10630 IF TA>0 THEN BA=1
10640 IF FNGA(CHR$(9)+CHR$(10)+CHR$(32),BC$,1)>0
        THEN BC$=MID$(BC$,2):GOTO 10640
10650 IF LEFT$(BC$,1)=":"OR LEFT$(BC$,1)=CHR$(143)
      THEN IF BA=1 THEN AB=1:BF$=LEFT$(BC$,LEN(BC$)-1):
      GOTO 10990 ELSE 10480
10660 IF LEFT$(BC$,1)=CHR$(132)
        THEN BF$=LEFT$(BC$,LEN(BC$)-1):BA=1:AB=1:
        GOTO 10990
10670 'examine line one character at a time
10680 ON FNGA(CB$,BC$,PA)GOTO 10780,10780,10780,10990
10690 ON FNGA(CC$,BC$,PA)GOTO 10830,10820,10820,10820,
        10820,10810,10800,10850
10700 IF FNGA(CD$,BC$,PA)>0 THEN 10890
10710 IF FNGA(CG$,BC$,PA)=0 THEN 10740
10720 IF PA=1 THEN 10970
10730 IF FNGA(CHR$(255),BC$,PA-1)=0 THEN 10970
10740 IF FNGA(CH$,BC$,PA)=0 THEN 10770
10750 IF PA=1 THEN AB=1:GOTO 10770
10760 IF FNGA(CHR$(255),BC$,PA-1)=0 THEN AB=1
10770 BF$=BF$+MID$(BC$,PA,1)
10780 PA=PA+1:GOTO 10680:
        '       skip this character - get another
10790 'numeric constants
10800 BF$=BF$+MID$(BC$,PA,4):PA=PA+4:
        '  pass 8 byte number
10810 BF$=BF$+MID$(BC$,PA,2):PA=PA+2:
        '  pass 4 byte number
10820 BF$=BF$+MID$(BC$,PA,1):PA=PA+1:
        '  pass 2 byte number
10830 BF$=BF$+MID$(BC$,PA,2):PA=PA+2:GOTO 10680:
        '1 byte number
10840 'string constants
10850 PB=INSTR(PA+1,BC$,CHR$(34))
10860 IF PB=0 THEN PB=LEN(BC$):
        BC$=LEFT$(BC$,PB-1)+CHR$(34)+CHR$(0)
10870 BF$=BF$+MID$(BC$,PA,PB-PA+1):PA=PB+1:GOTO 10680
10880 'variables
10890 VA$="":LD=0
10900 VA$=VA$+MID$(BC$,PA,1):PA=PA+1
10910 IF FNGA(CD$+CE$,BC$,PA)>0 THEN 10900
10920 IF FNGA(CF$,BC$,PA)>0 THEN VA$=VA$+MID$(BC$,PA,1):
        PA=PA+1
10930 IF MID$(BC$,PA,1)="("THEN VA$=VA$+"(":PA=PA+1
10940 'if this is a variable name, shorten it
10950 LD=LD+1:IF VA$=VB$(LD,1)THEN BF$=BF$+VB$(LD,2):
        GOTO 10680
10960 IF LD=HB THEN BF$=BF$+VA$:GOTO 10680 ELSE 10950
10970 IF MID$(BC$,PA-1,1)=":
        "THEN BF$=LEFT$(BF$,LEN(BF$)-1):PA=PA-1:GOTO 10970
10980 'file building decisions
10990 IF BG$=""THEN GOSUB 11050:BA=0:GOTO 11020
11000 IF BA=1 THEN BA=0:GOSUB 11060:GOSUB 11050:
        GOTO 11020
11010 IF LEN(BG$)+LEN(BF$)<250
        THEN GOSUB 11080ELSE GOSUB 11060:GOSUB 11050
11020 IF AB=1 THEN AB=0:GOSUB 11060
11030 GOT0 10480
11040 'build new file
11050 BG$=BF$:BF$="":ND$=LC$:RETURN
11060 NA!=NA!+LEN(BG$)+5:NB$=FNPD$(NA!)
11070 PRINT#2,NB$;ND$;BG$;CHR$(0);:BG$="":RETURN
11080 BG$=BG$+":"+BF$:BF$"":RETURN
11090 'close new file
11100 PRINT#2,CHR$(0);CHR$(0);CHR$(26);
11110 CLOSE
11120 PRINT CL$:END
11130 'second character of new variable name
11140 DATA A,B,C,D,E,"2",G,H,"3",J,K,L,M,"4","5",P,Q,"6",
        "7",T,U,V,W,X,Y,Z
```

Listing 4. PC.BAS — A program concentrator for MBASIC programs.

program from the keyboard. Try leaving out the space after IF or THEN. The documentation on my version of MBASIC says they must be there. PC.BAS has no problem eliminating ALL spaces. CALYPSO.BAS will show you that they are gone even though a LISTing of the program will show some spaces.

PC.BAS can combine statements on a single line better than you can from the keyboard. A Tokenized line can be 255 characters long. When the Tokens are expanded to their ASCII equivalent in a LISTing, the resulting line can be much longer than 255 characters. The LISTed line will be truncated at 255 ASCII characters, but the entire line is present in memory and it works as well as the original lines.

Limitations

Because of the above, a concentrated program often cannot be LISTed, EDITed, MERGEd or SAVEd in ASCII format. If you want to perform these operations, experiment with the number in the IF statement in line 11010. It determines the maximum length to which lines will be concatenated. Reducing it from it's present value of 250 to 100 should result in a moderately concentrated program that will work in these cases. (CRG.BAS and PC.BAS were reduced from Source files with this number set to 100. The REMarks and some formatting characters were replaced after concentration.)

There are limits on the number of variable and target line references CRG.BAS and PC.BAS can handle. Each target line can be referenced (called) from more than 120 reference lines. Each variable can appear in more than 100 lines (Shorter names can appear in more lines). These limits are quite high. A 'STRING TOO LONG' Error will result if you exceed these limits.

When shortening variable names to two characters, it was important to keep the first letter of the new name the same as that of the old name. This is necessary because the DEFINT, DEFDBL, etc. commands affect variables with names beginning with specific letters. The last line of PC.BAS contains a list of possible second characters for the new variable names. If a program has more than 26 variable names beginning with the same letter, you will get an 'OUT OF DATA' error.

CRG.BAS and PC.BAS can be made to run faster. They can, also, be combined into a single program and share some common routines. I don't plan to do this. ✱

About The Author

John Harper is a retired Army NCO, presently working for Bendix Field Engineering Corporation at the NASA Tracking Station on Ascension Island, South Atlantic Ocean. He programs for fun and to keep his mind off the fact that, for the last five years, he's been living and working on top of a 'dormant' volcano.

The Z-100 PC (Z150/Z160) Technical Column (#5)

Displaying Text On The Z-100 PCs Part II

Mark J. Foster
Senior Systems Engineer
Systems Software Engineering
Zenith Data Systems Corporation

Last month, we covered the basics of placing characters on the screen by using the Z-100 PC's ROM I/O calls. This month, we'll pick up on that discussion by providing a few more details on more sophisticated display calls, using attributes and multiple pages.

Text Mode Display Attributes

The table presented last month showed how the ROM uses attributes to describe special properties of displayed information. The attribute byte includes such information as the foreground and background colors, intensified video, flash, etc. The primary method of communicating this information to the ROM is to use the 'display-character-with-attribute' function in the ROM. The description of this function (repeated from last month) is as follows:

Display character with attribute: AH = 09H
AL = Character to be displayed
BH = Page number (normally 0)
CX = Repeat count (normally 1)
BL = Attribute, as follows:
--- For video modes 0-3 ---
Bit 7: Flash this character
Bit 6: Background Red bit
Bit 5: Background Green bit
Bit 4: Background Blue bit
Bit 3: Intensity bit (1 = bright)
Bit 2: Foreground Red bit
Bit 1: Foreground Green bit
Bit 0: Foreground Blue bit
--- For video modes 4-5 ---
Value of 0: Black
Value of 1: Cyan
Value of 2: Magenta
Value of 3: White
--- For video mode 6 ---
Value of 1: White

Shown in the function description are both text mode and graphics mode attributes. To begin with, take a look at the text mode attributes. Notice that there are two sets of Red-Green-Blue bits, which are bits 0-2 (for the foreground color), and bits 4-6 (for the background color). The foreground color is the color which the character itself will be displayed, and the background color is the color which will surround the character in the character's 'cell' (this is an 8 x 8 dot imaginary box which surrounds all characters on the screen). If you want to use a specific color, the following table shows how to encode the R-G-B bits to get a particular tint:

Red	Green	Blue	Color
0	0	0	Black
0	0	1	Blue
0	1	0	Green
0	1	1	Cyan
1	0	0	Red
1	0	1	Magenta
1	1	0	Brown/Dk. Yellow
1	1	1	White

Notice that both of the above R-G-B fields use the same encoding, so you can use the same values for both foreground and background color. In addition to the colors shown here, the Intensity bit adds even more FOREGROUND colors. This bit will make the character 'brighter' than other characters which do not have the intensity bit set. On a color monitor, this brightness change has a very dramatic affect by changing not only the brightness of each character, but the color you perceive. The color is actually the same, but the saturation level is different enough to change its appearance substantially. The intensity bit only effects the foreground color, however. The background color remains the same. This actually means that you have at your disposal a total of sixteen foreground colors and eight background colors.

The last attribute bit is quite straightforward — flash or blink. Simply turn this bit on if you want to display characters on the screen which flash. As with the other attributes shown above, all combinations of attributes are acceptable electronically — though humans usually have a hard time with combinations like flashing bright-magenta on cyan characters!

With all of this information, it's time to finally sneak in a little code. The following routine will display a user-defined string, pointed to by DS:SI, at the screen position in DX (DH:row, and DL:column), on the page number in BH, using the attribute in BL! The routine assumes that the string is terminated by a NULL byte of 00H. The one instruction you might not be familiar with is 'LODSB'. This single instruction tells the machine to fetch a byte from the memory pointed to by DS:SI into AL, and then to increment SI to point to the next character.

```
DISPLAY_STRING PROC NEAR
DS1: MOV    AH,2  ;Get the command to set cursor position
     INT    1ØH   ;Set cursor position to that requested
     LODSB        ;Get char from string, point to next
     TEST   AL,AL ;Is this the end of the string?
     JZ     DS2   ;Yes - finished; return to the user
     MOV    AH,9  ;Get command to display char w/attrib
     MOV    CX,1  ;Set repeat-count to 1 (just 1 char!)
     INT    1ØH   ;Display a character of the string
     INC    DL    ;Advance to the next column
     JMP    DS1   ;Continue displaying characters
DS2: RET          ;Finished displaying the string
DISPLAY_STRING ENDP
```

Note that the routine does not support wrap-around at the end of a line or at the end of the screen. This is actually desirable when you want to display a frame around the edge of the screen, for instance. If the routine checked for wrap-around, when you displayed a character at the last position of the bottom of the screen, the screen would scroll upwards, thereby messing up your nice screen layout.

The following program fragment demonstrates how to use the display string; it will display "Z-100 PC!" in the middle of the screen, using bright-white on blue characters. The example assumes that the DS register is already set (this is true when you are working from scratch in DEBUG, or when your program is a .COM file).

```
        MOV     SI,OFFSET ZMSG ;Point to the message string
        MOV     DH,12    ;Display on line 12
        MOV     DL,35    ;Display at column 35
        MOV     BH,Ø     ;...on page Ø
        MOV     BL,1FH   ;Set attribute; brt. blue/white
        CALL    DISPLAY_STRING ;Print the message on screen

ZMSG:DB         'Z-1ØØ PC!',Ø
```

Multiple Pages And Speed!

In the Z-150 and Z-160's text modes, there are multiple pages to work with. In 40-column mode (video modes 0 and 1), there are eight display pages; 80 column mode supports four pages. Few current programs are written to take advantage of this feature — in fact, you might be wondering what good multiple pages are, anyway! It all boils down to speed.

Whenever your program writes to the screen, the person who is running the program must wait. The PERCEIVED delay may give users the impression that your program is very slow. One very interesting point is that you can actually slow down your program slightly, but give the user the impression that the program is running faster! This may seem odd, but multiple pages are the key.

There are many ways to increase the perception of speed, but the one we will focus on here is to let your program update an alternate page, and then 'flip' suddenly to that page. Given a reasonably fast program, virtually all users will report that the program seems to be running much faster than if the information is drawn on the currently displayed page, despite the fact that the program has actually incurred additional overhead, and is running slightly slower.

The procedure for displaying information on alternate pages is very simple. As the above example shows, the page number is just another parameter to the ROM's video code. You could change the 'MOV BH,0' instruction in the second example to 'MOV BH,1' to display the information on the second page in video memory. Unfortunately, this just puts the text into the memory on the video card. Since only one page is being displayed at a time on your monitor, you won't be able to see the second page! Therefore, you must tell the ROM to switch video pages after you've finished drawing the new information on the alternate page. To do this, just place the new page number in AL, and a function code of 5 (Select New Display Page) in AH, and then perform the usual Interrupt 10H...

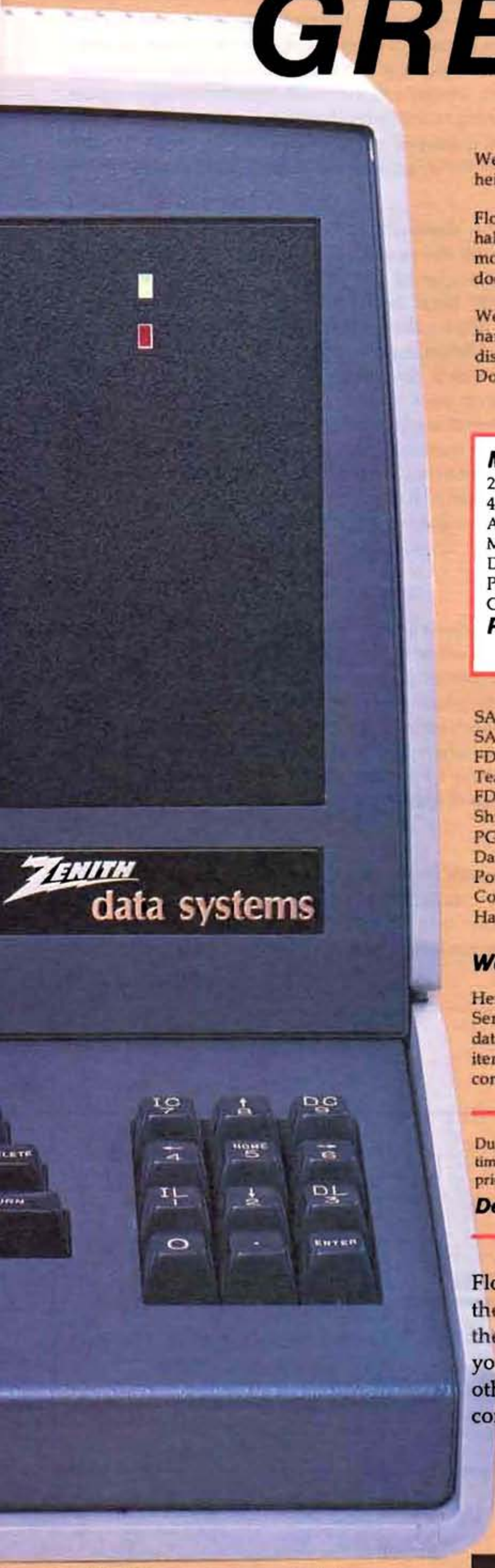
```
     MOV   AL,1         ;We want to display page 1
     MOV   AH,5         ;Set function to Set-Page
     INT   1ØH          ;Display the second page!
```

This simple technique can be very dramatic. Multiple pages are also very effective when your program has multiple 'modes'. To support these types of programs, you can write the different video pages with information corresponding to the screen display associated with each mode once, when your program first starts up. Then, when the user switches between program modes, all you have to do is set the new display page — it isn't necessary to keep redrawing screens over and over again.

Next Time...

To continue on with the current series, next time I'll talk about graphics displays, including details on how to create your own graphics character set! Till then, keep playing with your machine, and continue learning about it — that's how you become an expert! ✲

GREAT DRIVES!



The H89 TWOET systems Just Got Hard to beat!

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 density, half height drives in either 48 or 96 tpi format, all 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. And now by popular demand, the new 10 megabyte hard disk system! We are now able to offer you the storage you want and need for a fair price! The hard disk system comes with all software needed to run with the 17 or 37 controller and/or the Magnolia Double Density controller.

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....... $355.00

Model TWOET 55F

Two Teac 55F half height
96 tpi double sided
All hardware
Metal, shielded mounting plates
Data cable with chassis connector
Power"Y" connector
Complete documentation
Price....... $385.00

NEW!

HARD DISK
for
H-89
$995.00
Internal

SA-860 DS/DD half hgt 8"	505.00 2@ 495.00 ea.
SA-455 DS 48tpi ½ hgt 5.25"	140.00
FDD-100-8 SS/DD 8"	125.00
Teac 55F 96tpi DS/DD half height	155.0
FDD-200-8 DS/DD 8"	***Sale 185.0***
Shugart 10 MB half height hard disk	725.00
PGS, HX-12 RGB color monitor	525.00
Data connectors of all types	***CALL***
Power connectors for all drives	***CALL***
Controllers	***CALL***
Hard disk Z150/160	Internal 750.00

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 $60.00 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 $80.00 item. And the cable that comes with your TWOET system includes the external chassis disk I/O connector.

Due to production deadlines, this ad is 2 months old at time of printing. We encourage you to call for our latest pricing and catalog requests.

Dealer inquiries invited.

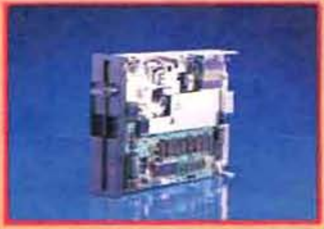
We accept cash, MasterCharge, Visa, personal checks (allow 10 days to clear the bank), wire transfers, money orders, purchase orders from government agencies, and approved businesses. Call for info.

Floppy Disk Services inc. has been supplying custom enclosures and disk drives to the hobby and professional market for 6 years now! We have specifically supported the Heath/Zenith community with a high degree of success. We care about you and your system integrity. That's why we have been a success in a time when many others are going out of business. We are proud to be a part of the Heath/Zenith community and will continue our support with new products and ideas.

H-88/89/90 and Z-150/160 are registered trademarks of Heath Corp.

GREAT PRICES!

SA-455



Shugart SA-455 half height 5¼ is the standard for excellence in the disk drive community. Backed by a one year warranty and manufactured by the leader of drive technology, Shugart Corp. Ready to ship at $140.00 ea. Compatible with ALL of the Heath/Zenith computers.

Maynard and DTC

We can supply PC compatible upgrade boards for your Z-150, Z160 system. Hard disk controllers, memory upgrade, and space saving Maynard modular boards. As an example, you can install a Maynard Modular 5 & 8 controller that will run 5¼, 8 inch and optional boards such as a clock/calendar module, game port, serial port or parallel port. Call for details on the slick way to upgrade and save a slot at the same time...

Irwin-110

Data loss! Its a terrible thought. Minimize your chance of lost programs and data files with the new 110 tape drive from Floppy Disk Services inc. It works off the standard 5¼ controller which means you save money over standard tape backups that you must buy the controller for. The data cartridge holds 10mb of data and can retrieve in streaming or individual file form. Software included. Available in many configurations. For the IBM-PC, Z-150 or Z-160 only. (CP/M software is being worked on). Tape unit $395 ea., tape cartridge $29.95 each...

Z-150/160 Upgrade

We can upgrade your Z-150/160 with half height floppies, hard disks, 8 inch drive systems, clock modules, memory and much more! Glance over our ad and call toll free outside NJ for our latest catalog. Or call and ask our sales/technical experts for their upgrade suggestions...

SA-712 & MicroScience

Half height 10 megabyte (20mb available soon) hard disk drives. The MicroScience has a plated media and is available with a full size bezel option. The Shugart has an on board low current option and is backed by a one year warranty. It also has a full size bezel option. both drives are available for the H/Z-89, H/Z-150, H/Z-160. $550.00 each.

Connectors

Floppy Disk Services inc. can supply custom data and power cables for your needs. We use only the best! UL rated or listed components, and all power connectors are crimped by our AMP electric certified crimper. No hand crimping to vary the degree of accuracy. Only the best! Floppy Disk Services inc. can make 1 or 1000 connectors to your specs, fast! Call and ask for details...

Special Sale!

Because of our enormous buying power in the industry we have made and excellent purchase of disk drives for your Heath/Zenith computer. Two Teac 55F 80 track double sided drives, in case with power supply and shielded data cable, your price $395.00. Single drive system, $245.00.

Enclosures!

We manufacture and stock enclosures for every disk drive on this page and for almost any other drive you are likely to find. Available assembled and tested with state-of-the-art power supplies, these enclosures house a wide combination of half- and full-height drives to complement most available computers.

Toll Free Order Line:
(800) 223-0306

Tech Help or Info:
(609) 799-4440

39 Everett Drive Bldg. D, Lawrenceville, New Jersey 08648

FLOPPY
DISK
SERVICES™
INC.

RamPal

For Z-150/160

RamPal-100

For Z-100 dual processor

Full System Memory Without Extra Memory Boards!

- 640K on the standard memory board of the Z-150/160
- 768K on the motherboard° of the Z-100 Dual Processor
- Save money and save an expansion slot!

RamPal for the Z-150/160 List Price: $39.95
RamPal-100 for the Z-110/120 List Price: $79.95

° For late model motherboards. Contact Software Wizardry or your dealer for details on older models

Software Wizardry
THE MAGIC TOUCH

1106 First Capitol Drive
St. Charles, MO 63301
(314) 946-1968

UMENU

for the Z-100

List price $24.95

MS-DOS Menu interface program

Sets your system up so that all your programs and files can be selected quickly with easy to use menus & submenus.
Lightning Quick with the Software Wizardry Lightpen!

- Menu-based access to all features of MS-DOS
- Create on-line help files
- Item selection from keyboard or optional lightpen
- Fully user-configurable menus with graphics
- Direct DOS command line entry within UMENU
- Runs memory-resident under MS-DOS 2
- Optionally displays current path name
- Unlimited fully linked menus

Available from Heath/Zenith Computers & Electronics,
and authorized Software Wizardry Dealers.

Software Wizardry
THE MAGIC TOUCH

1106 First Capitol Drive
St. Charles, MO 63301
(314) 946-1968

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.

HDOS HARDCOPY SOFTWARE

Part Number	Description of Product	Selling Price	Vol. Issue
885-1008	Volume I Documentation	9.00	
885-1013	Volume II Documentation	12.00	
885-1015	Volume III Documentation	9.00	
885-1037	Volume IV Documentation	12.00	8
885-1058	Volume V Documentation	12.00	

MISCELLANEOUS HDOS COLLECTIONS

Part Number	Description of Product	Selling Price	Vol. Issue
885-1032	Disk V H8/H9	18.00	8
885-1044-[37]	Disk VI H8/H89	18.00	
885-1064-[37]	Disk IX H8/H89 Disk	18.00	
885-1066-[37]	Disk X H8/H89	18.00	10
885-1069	Disk XIII Misc H8/H89	18.00	

GAMES

HDOS

Part Number	Description of Product	Selling Price	Vol. Issue
885-1010	Adventure Disk H8/H89	10.00	4
885-1029-[37]	Disk II Games 1 H8/H89	18.00	8
885-1030-[37]	Disk III Games 2 H8/H89	18.00	8
885-1031	Disk IV MUSIC H8 Only	20.00	25
885-1067-[37]	Disk XI H8/H19/H89 Games	18.00	12
885-1068	Disk XII MBASIC Graphic Games	18.00	10
885-1088-[37]	Disk XVII MBASIC Graph. Games	20.00	14
885-1093-[37]	D&D H8/H89 Disk	20.00	16
885-1096-[37]	MBASIC Action Games H8/H89	20.00	18
885-1103	Sea Battle HDOS H19/H8/H89	20.00	20
885-1111-[37]	HDOS MBASIC Games H8/H89	20.00	23
885-1112-[37]	HDOS Graphic Games H8/H89	20.00	23
885-1113-[37]	HDOS Action Games H8/H89	20.00	23
885-1114	H8 Color Raiders & Goop	20.00	23
885-1124	HUGMAN & Movie Animation Pkg	20.00	41
885-1125	MAZEMADNESS	20.00	41
885-1130	Star Battle	20.00	47
885-1133-[37]	HDOS Games Collection I	20.00	59
835-8009-[37]	HDOS & CP/M Galactic Warrior	20.00	32
885-8022	HDOS SHAPES	16.00	45
885-8026	HDOS Space Drop	16.00	49
885-8032-[37]	HDOS Castle	20.00	59

CP/M

Part Number	Description of Product	Selling Price	Vol. Issue
885-1206-[37]	CP/M Games Disk	20.00	11
885-1209-[37]	CP/M MBASIC D&D	20.00	19
885-1211-[37]	CP/M Sea Battle	20.00	20
885-1220-[37]	CP/M Action Games	20.00	32
885-1222-[37]	CP/M Adventure	10.00	35
885-1227-[37]	CP/M Casino Games	20.00	36
885-1228-[37]	CP/M Fast Action Games	20.00	39
885-1236-[37]	CP/M Fun Disk I	20.00	55

ZDOS

Part Number	Description of Product	Selling Price	Vol. Issue
885-3004-37	ZDOS ZBASIC Graphic Games	20.00	37
885-3009-37	ZDOS ZBASIC D&D	20.00	50
885-3011-37	ZDOS ZBASIC Games Disk	20.00	52
885-3017-37	ZDOS Contest Games Disk	25.00	58

UTILITIES

HDOS

Part Number	Description of Product	Selling Price	Vol. Issue
885-1022-[37]	HUG Editor (ED) Disk H8/H89	20.00	20
885-1025	Runoff Disk H8/H89	35.00	
885-1060-[37]	Disk VII H8/H89	18.00	
885-1061	TMI Load H8 ONLY Disk	18.00	
885-1062-[37]	Disk VIII H8/H89 (2 Disks)	25.00	
885-1063	Floating Point Disk H8/H89	18.00	
885-1065	Fix Point Package H8/H89 Disk	18.00	10
885-1075	HDOS Support Package H8/H89	60.00	
885-1077	TXTCON/BASCON H8/H89	18.00	
885-1079-[37]	HDOS Page Editor	25.00	15
885-1080	EDITX H8/H19/H89 Disk	20.00	
885-1082	Programs for Printers H8/H89	20.00	
885-1083-[37]	Disk XVI Misc H8/H89	20.00	11
885-1089-[37]	Disk XVIII Misc H8/H89	20.00	20
885-1090-[37]	Disk XIX Utilities H8/H89	20.00	22
885-1092-[37]	Relocating Debug Tool H8/H89	30.00	14
885-1098	H8 Color Graphics ASM	20.00	19
885-1099	H8 Color Graphics Tiny PASCAL	20.00	19
885-1105	HDOS Device Drivers H8/H89	20.00	24
885-1116	HDOS Z80 Debugging Tool	20.00	27
885-1119-[37]	BHBASIC Support	20.00	29
885-1120-[37]	HDOS 'WHEW' Utilities	20.00	33
885-1121	HDOS Hard Sec Sup Pkg 2 Disks	30.00	37
885-1123	XMET Robot Cross Assembler	20.00	40
885-1126	HDOS Utilities by PS:	20.00	42
885-1127-[37]	HDOS Soft Sector Support Pkg	30.00	45
885-1128-[37]	HDOS DISKVIEW	16.00	46
885-1129-[37]	HDOS CVT Color Video Terminal	20.00	46
885-8001	SE (Screen Editor)	25.00	28
885-8003	BHTOMB	25.00	28
885-8004	UDUMP	35.00	28
885-8006	HDOS SUBMIT	20.00	31
885-8007	EZITRANS.	30.00	30
885-8015	HDOS TEXTSET Formatter	30.00	42
885-8017	HDOS Programmers Helper	16.00	42
885-8024	HDOS BHBASIC Utilities Disk	16.00	46

CP/M

Part Number	Description of Product	Selling Price	Vol. Issue
885-1210-[37]	CP/M ED (same as 885-1022)	20.00	20
885-1212-[37]	CP/M Utilities H8/H89	20.00	21
885-1213-[37]	CP/M Disk Utilities H8/H89	20.00	22
885-1217-[37]	HUG Disk Duplication Utilities	20.00	26
885-1223-[37]	HRUN HDOS Emulator 3 Disks	40.00	37
885-1225-[37]	CP/M Disk Dump & Edit Utility	30.00	40
885-1226-[37]	CP/M Utilities by PS:	20.00	40
885-1229-[37]	XMET Robot Cross Assembler	20.00	40
885-1230-[37]	CP/M Function Key Mapper	20.00	42
885-1231-[37]	Cross Ref Utilities for MBASIC	20.00	43
885-1232-[37]	CP/M Color Video Terminal	20.00	46
885-1235-37	CP/M COPYDOS	20.00	54
885-1237-[37]	CP/M Utilities	20.00	55
885-1245-37	CP/M 85 KEYMAP	20.00	63
885-1246[-37]	CP/M HUG File Manager & Utilities	20.00	64
885-5001-37	CP/M 86 KEYMAP	20.00	51
885-5002-37	CP/M 86 HUG Editor	20.00	52
885-5003-37	CP/M 86 Utilities by PS:	20.00	54
885-5008-37	CP/M 8080 To 8088 Trans. & HFM	20.00	64
885-8018-[37]	CP/M FAST EDDY & BIG EDDY	20.00	43
885-8019-[37]	DOCUMAT and DOCULIST	20.00	43
885-8025-37	CP/M 85/86 FAST EDDY	20.00	49

ZDOS

Part Number	Description of Product	Selling Price	Vol. Issue
885-3005-37	ZDOS ETCHDUMP	20.00	39
885-3007-37	ZDOS CP/EMulator	20.00	47
885-3008-37	ZDOS Utilities	20.00	47
885-3010-37	ZDOS KEYMAP	20.00	51
885-3022-37	ZDOS/MSDOS Useful Programs I	30.00	63
885-3023-37	ZDOS/MSDOS EZPLOT	20.00	63
885-8029-37	ZDOS FAST EDDY	20.00	53

H/Z100 ZDOS - H/Z150 MSDOS

Part Number	Description of Product	Selling Price	Vol. Issue
885-3012-37§§	ZDOS HUG Editor	20.00	52
885-3014-37§§	ZDOS/MSDOS Utilities II	20.00	54
885-3016-37§	ZDOS/MSDOS Adventure	10.00	57
885-3020-37§	MSDOS HUG Menu System	20.00	62
885-3021-37§§	ZDOS/MSDOS Cardcat	20.00	63
885-3024-37§	ZDOS/MSDOS 8080 To 8088 Trans	20.00	64
885-3025-37§§	ZDOS/MSDOS Misc. Utilities	20.00	64

§ All program files run on both
§§ Program files run partially on both

PC/IBM COMPATIBLE

Part Number	Description of Product	Selling Price	Vol. Issue
885-6001-37	MSDOS Keymapper	20.00	59
885-6002-37	CP/EMulator II & ZEMulator	20.00	59
885-8033-37	MSDOS Fast Edit	20.00	62

PROGRAMMING LANGUAGES

HDOS

Part Number	Description of Product	Selling Price	Vol. Issue
885-1038-[37]	Wise on Disk H8/H89	18.00	
885-1042-[37]	PILOT on Disk H8/H89	19.00	
885-1059	FOCAL-8 H8/H89 Disk	25.00	13
885-1078-[37]	HDOS Z80 Assembler	25.00	21
885-1085	PILOT Documentation	9.00	
885-1086-[37]	Tiny HDOS PASCAL H8/H89	20.00	13
885-1094	HDOS Fig-Forth H8/H89	40.00	18
885-1132-[37]	HDOS Tiny BASIC Compiler	20.00	59
885-1134	HDOS SMALL-C Compiler	30.00	63

CP/M

Part Number	Description of Product	Selling Price	Vol. Issue
885-1208-[37]	CP/M Fig-Forth H8/H89 2 Disks	40.00	18
885-1215-[37]	CP/M BASIC-E	20.00	26

BUSINESS, FINANCE AND EDUCATION

HDOS

Part Number	Description of Product	Selling Price	Vol. Issue
885-1047	Stocks H8/H89 Disk	18.00	
885-1048	Personal Account H8/H89 Disk	18.00	
885-1049	Income Tax Records H8/H89 Disk	18.00	
885-1055-[37]	MBASIC Inventory Disk H8/H89	30.00	
885-1056	MBASIC Mail List	30.00	
885-1070	Disk XIV Home Fin H8/H89	18.00	
885-1071-[37]	MBASIC SmBusPk H8/H19/H89	75.00	17
885-1091-[37]	Grade/Score Keeping H8/H89	30.00	14
885-1097-[37]	MBASIC Quiz Disk H8/H89	20.00	18
885-1118-[37]	MBASIC Payroll	60.00	30
885-1131-[37]	HDOS CHEAPCALC	20.00	47
885-8010	HDOS CHECKOFF	25.00	32
885-8021	HDOS Student's Statistics Pkg	20.00	44
885-8027	HDOS SCICALC	20.00	50

CP/M

Part Number	Description of Product	Selling Price	Vol. Issue
885-1218-[37]	CP/M MBASIC Payroll	60.00	31
885-1233-[37]	CP/M CHEAPCALC	20.00	47
885-1239-[37]	Spread Sht. Contest Disk I	20.00	
885-1240-[37]	Spread Sht. Contest Disk II	20.00	
885-1241-[37]	Spread Sht. Contest Disk III	20.00	
885-1242-[37]	Spread Sht. Contest Disk IV	20.00	
885-1243-[37]	Spread Sht. Contest Disk V	20.00	
885-1244-[37]	Spread Sht. Contest Disk VI	20.00	
885-8011-[37]	CP/M CHECKOFF	25.00	32

ZDOS

Part Number	Description of Product	Selling Price	Vol. Issue
885-3006-37	ZDOS CHEAPCALC	20.00	47
885-3013-37	ZDOS Checkbook Manager	20.00	54
885-3018-37	ZDOS Contest Spreadsheet Disk	25.00	58
885-8028-37	ZDOS SCICALC	20.00	50
885-8030-37	ZDOS MATHFLASH	20.00	55

DATA BASE MANAGEMENT SYSTEMS

HDOS

Part Number	Description of Product	Selling Price	Vol. Issue
885-1107-[37]	HDOS Data Base System H8/H89	30.00	23
885-1108-[37]	HDOS MBASIC Data Base Sys.	30.00	23
885-1109-[37]	HDOS Retriever ASM (3 Disks)	40.00	23
885-1110	HDOS Autofile (2 Disks)	30.00	23
885-1115-[37]	HDOS Navigational Program	20.00	25
885-8008	Farm Accounting System	45.00	30

Vectored to Page 67 ☞

HUG NEW PRODUCTS

HUG P/N 885-3026-37 MS-DOS SMALL-C Compiler $25.00

Introduction: This set of diskettes forms a complete compiler for the SMALL-C language. The compiler converts a file of statements in SMALL-C into a file of statements in MS-DOS assembly language code. Assembling the generated file, along with the runtime support library code (included), creates an executable file. The SMALL-C language is a true subset of the UNIX C language of sufficient power and complexity to write useful programs, including the SMALL-C compiler in which itself was written. Many example programs and massive documentation is provided with the compiler package. The full source code for this compiler is also included so the compiler can be modified and reassembled.

Requirements: The SMALL-C compiler requires the MS-DOS operating system (Version 2.0 or greater) on an H/Z-100 computer system. The computer should have at least 128k of memory. It should also have at least two 5" disk drives.

The following program and files are included on the HUG P/N 885-3026-37 MS-DOS SMALL-C Compiler disks:

Disk A

C88	.COM	STDIO	.H
CRUN	.OBJ	HELLO	.C
BUG2	.C	WDCNT	.C
SEE	.C	EXAMPLES	.DOC
SMALLC	.DOC	README	.DOC
SMALLC88	.LIB	CMAKE	.BAT
PARROT	.C	BUG1	.C
SSORT	.C	CHARCNT	.C
C88USE	.DOC	CRUN	.DOC

Disk B

COMPILER	.LNK
C88MAINT	.DOC
CMAINT	.BAT

Disk B \COMPILER

C88A3	.C	C88B2	.C
C88F1	.C	C88E1	.C
C88G1	.C	C88H1	.C
C88DEFS	.H	C88	.H
C88A3	.OBJ	C88B1	.OBJ
C88D1	.OBJ	C88E1	.OBJ
C88H1	.OBJ	C88H2	.OBJ
C88D1	.C	C88A1	.C
C88B1	.C	C88C1	.C
C88H2	.C	C88DEFS	.H
C88I1	.C	C88A2	.OBJ
C88A1	.OBJ	C88C1	.OBJ
C88B2	.OBJ	C88G1	.OBJ
C88F1	.OBJ	C88A2	.C
C88I1	.OBJ		

Disk B \RUNTIME

PARSE	.ASM	IODEFS	.H
CRUN	.C	SMALLC88	.LIB
CRUN	.OBJ		

Program Author: David A. Wallace

Program Content: The author of this package derived this compiler from Ron Cain's Version 1.1 SMALL-C Compiler, which was published in two issues of Doctor Dobb's Journal in 1980. He transcribed the original code and all published fixes to it, added several enhancements of his own, and retargeted the compiler to produce assembly code compatible with MS-DOS's assembler. The runable compiler is on Disk A, along with the standard I/O definitions file, STDIO.H. There's also an explanatory or "getting-started" file, README.DOC, on that disk.

C88USE.DOC explains how to run the compiler, SMALLC.DOC is a description of the language itself, CRUN.DOC describes the runtime library, and EXAMPLES.DOC explains the included sample programs.

Disk B contains the files necessary to modify or reconstruct the compiler and runtime support library files. Two subdirectories are included on this disk: \COMPILER\ (contains the compiler source files) and \RUNTIME\ (contains the runtime source files). This disk is included with this product for those programmers and experimenters who are quite knowledgeable in the 'C' language, as well as assembly language and are not required for the proper operation of the compiler itself.

According to the author, the compiler is entirely compatible at the source code level with the UNIX C compiler; it supports a true subset of UNIX C.

Comments: None

TABLE C Rating: (10)

HUG P/N 885-5006-37 and 885-5007-37 HUGPBBS Update

Due to an oversight (bug) in Version 1.00 of the HUG Personal Bulletin Board System, a new version, 1.01, is being made available to original owners at no extra charge. Because of minor hardware differences in the Hayes 300 baud and 1200 baud modems, the original version, 1.00, of HUGPBBS will hang up on an incoming call. This will only occur on the Hayes 300 baud Smartmodem. Version 1.01 fixes this problem. To receive your update, return your original disk to Nancy Strunk here at the Heath Users' Group, and it will be updated and returned to you free of charge. This update affects both, the executable file and source code.

HUG P/N 885-6003-37 MS-DOS EZPLOT $20.00

Introduction: EZPLOT is a user friendly, high-resolution graphics function plotting routine.

Requirements: EZPLOT works on the H/Z-150/160 computer running MS-DOS (Version 2.0 or higher). One disk drive is needed and the program itself occupies 60k of RAM. To use the internal screendump capability of EZPLOT, one of the following printers is needed: C. Itoh Prowriter, Epson MX or FX series with Graftrax, NEC 8023A, IDS Prism, or an Okidata 92.

The following files are included on the HUG P/N 885-6003-37 EZPLOT disk:

EZPLOT	.COM	AAM	.DAT
EZPLOT	.ND	FNC	.DAT
README	.DOC		

Program Content: Some of EZPLOT's features are as follows:

1. EZPLOT is menu driven. The user can quickly and easily format, view, and modify the plot in a logical step-by-step manner.
2. EZPLOT can plot as many as three functions [ie. f1(x), f2(x), f3(x)] on the same set of axes. It can also plot X-Y plots, giving the user the capability of plotting contours or path functions. Finally, it can plot two independent functions [ie. f1(x1), f2(x2)] overlaid on one another.
3. The plot can easily be viewed at any stage of development.
4. EZPLOT makes it simple to define the title, axis labels, and if plotting multiple function, the function names to be used in the legend.
5. The user can override the default axis ranges and set more esthetic ranges, if needed.
6. The user has control of the number of hashmarks on each axis by setting the number of axis intervals.
7. The user can plot segments of the curve to enlarge areas of interest.
8. EZPLOT gives the user several methods for generating a graphics screendump to printer.
9. The user does not have to worry about presorting the data. EZPLOT automatically checks to see if the data needs sorting and if so, does it.
10. Data files for EZPLOT can easily be produced using any programming language including: BASIC, FORTRAN, PASCAL, and 'C' or COBOL.

An extensive users' manual is included with EZPLOT.

Comments: This program is quite user friendly and according to the author, "virtually GORILLA proof".

TABLE C Rating: (10)

Heath/Zenith Users' Group

TABLE C
Product Rating

10 - Very Good
9 - Good
8 - Average

Rating values 8-10 are based on the ease of use, the programming technique used, and the efficiency of the product.

7 - Has hardware limitations (memory, disk storage, etc.)
6 - Requires special programming technique
5 - Requires additional or special hardware
4 - Requires a printer
3 - Uses the Special Function Keys (f1,f2,f3,etc.)
2 - Program runs in *Real Time**
1 - Single-keystroke input
0 - Uses the H19 (H/Z89) escape codes (graphics, reverse video)

Real Time — a program that does not require interactivity with the user. This term usually refers to games that continue to execute with or without the input of the player, e.g. p/n 885-1103 or 885-1211[-37] SEA BATTLE.

ORDERING INFORMATION

For Visa and MasterCard phone orders; telephone Heath Company Parts Department at (616) 982-3571. Have the part number(s), descriptions, and quantity ready for quick processing. By mail; send order, plus 10% postage and handling ($1.00 minimum charge, up to a maximum of $5.00. UPS is $1.75 minimum -- no maximum on UPS. UPS Blue Label is $4.00 minimum.), 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.

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.

Sequential Files — Part 4

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

Last month, we built an MBASIC program called SEQUTIL.BAS. With it we were able to read the contents of a sequential file into arrays for each field of the record. This month, we'll look at ways to use those arrays to look up, sort, change, delete, and add to the data we wish to save, and to write the data back from the arrays to our sequential file. To do this, we'll read our telephone number file, SEQFILE.DAT, into arrays using SEQUTIL.BAS through line 1180. That is, we'll delete lines 1200 – 1270 which printed out the contents of the array. Their only purpose was to prove that the array did contain our file data. Now we know that the program works, so we won't need that proof and can delete the lines which printed the array contents.

Before going farther, we should be sure to have a file which is large enough to show us what is going on. If your file SEQFILE.DAT contains fewer than 50 names and phone numbers, go back to MODSEQ.BAS and add names and numbers for a total of 50 to 100 names. You can make up names and numbers, or use a phone book to enter real data. When you've got enough names in the file, run SEQUTIL.BAS with the LPRINT statements so that you have a copy of what's in your file. Keep this copy handy as we'll use it to show the reason for some of the things we'll do as we progress through this series. Remember, data handling is one of the areas where computers really earn their keep. The principles we'll develop here apply to many data-handling applications and will enable you to write better, more convenient, and faster programs.

Let's begin by looking up a name from our computerized phone book. We'll start by reading our file into an array. Then we'll ask for a name and try to find it in the array. Because the entries are in no special order, we'll have to check each entry from the beginning to the end until we have a match. This is the slow way of doing things, but it has two very important strong points. First, it's the easiest to program, and second, it's a smaller program. For limited or occasional use, it's the way to go. We can write complex programs to do fancy things (we did with random files, and will do it again here), but remember, if the effort required to write a program is greater than the effort required to do the job manually, then do it the manual way. What I'm trying to say is that there's a place for cheap and dirty programming. Not sloppy programming, mind you. There's no place for that, but you don't always need all the frills.

As we make our comparisons, we'll check both the last name and the first name arrays. That will give you more flexibility in what you can find. Once found, the name and phone number will be displayed on the screen. The same procedure could be applied if there were fields in the record for addresses, or if the file were set up to inventory parts or anything else. Try to envision your own applications as we go through these exercises. Let's write our program.

```
2 ' **** FINDNAME.BAS
4 ' **** DAVID E. WARNICK
6 ' **** COPYRIGHT 1984
1000 DIM LN$(100),NM$(100),MI$(100),PN$(100)
       'DIMENSION ARRAYS
1010 OPEN "I",#1,"SEQFILE.DAT" 'OPEN THE SEQUENTIAL FILE
1020 CT=1        'SET THE COUNTER
1030 FOR X=1 TO 100     'SET UP THE LOOP
1040 IF EOF(1) THEN X=100:GOTO 1150 'STOP AT END OF FILE
1050 INPUT #1,R$          'READ A RECORD
1060 A=INSTR(1,R$,"\")  'FIND THE FIRST \
1070 B=INSTR(A+1,R$,"\")          'FIND THE SECOND \
1080 C=INSTR(B+1,R$,"\")          'FIND THE THIRD \
1090 LN$(CT)=LEFT$(R$,A-1)
       'ARRAY VARIABLE = LAST NAME
1100 NM$(CT)=MID$(R$,A+1,B-A-1)
       'ARRAY VARIABLE = FIRST NAME
1110 MI$(CT)=MID$(R$,B+1,C-B-1)
       'ARRAY VARIABLE = MIDDLE INITIAL
1120 D=LEN(R$)'LENGTH OF RECORD
1130 PN$(CT)=RIGHT$(R$,D-C)
       'ARRAY VARIABLE = PHONE NUMBER
1140 CT=CT+1      'INCREMENT THE COUNTER
1150 NEXT X       'DO IT FOR ALL RECORDS
1160 CLOSE #1     'CLOSE THE FILE WHEN DONE
```

The data file has now been read into an array. The counter, CT, is set to a number which is one greater than the number of elements in our array because it was incremented after the last record was added. You may have noted that we could have used the loop control variable, X, to specify the array element to be loaded. The quantity of records would have stayed in X if line 1040 read GOTO 1160. However, this would not have allowed us to exit the loop properly. The computer would become confused later because it would think it's still in that loop.

NEVER JUMP OUT OF A LOOP WHEN YOU'RE FINISHED WITH IT. ALWAYS MAKE THE LOOP TERMINATE ITSELF.

This applies to a premature loop exit, such as we would experience if our file had fewer than 100 records. When line 1040 sets X to 100, the terminal value for this loop's operation, then directs program execution to the NEXT X statement, it forces the loop to terminate normally. There's no danger of our program getting confused because it thinks it's still in a loop.

Our next step is to enter the name we want to find. Then we'll set up a loop to search each array element from the first (number 1) to the last (number CT-1). If a match for our input is found, the information will be printed and the loop will terminate. Program execution will be directed to ask for another name. If nothing is found, you'll receive a message and be asked for another name.

```
1200 INPUT "NAME TO LOOK UP ";LU$    'ENTER NAME TO FIND
1210 FOR X=1 TO CT-1     'SET UP THE LOOP
1220  IF LU$=NM$(X) OR LU$=LN$(X) THEN PRINT NM$(X);
      "  ";MI$(X);"  ";LN$(X);" ";PN$(X):X=CT-1:
      GOTO 1240
1230  IF  X=CT-1 THEN PRINT LU$;" IS NOT IN THE FILE"
      'ERROR MESSAGE
1240 NEXT X      'KEEP LOOKING
1250 PRINT       'PRINT A BLANK LINE
1260 GOTO 1200  'GET ANOTHER NAME TO LOOK UP
```

This program will run forever unless we provide a means to stop it. One line will permit this. First, we'll check for the lookup string LU$ to be empty (just a carriage return was struck). If it is, we'll tell the program to end.

```
1205 IF LU$="" THEN END
```

Let's take a more detailed look at line 1220 before we proceed. We've used the test operator "IF" before, but this time we threw in a new twist. We added the logical operator "OR". This let us check both the first and the last names in the file. When you run the program, try entering either first or last names from your printed list. You'll see that it finds them.

As written, the program will exit the loop as soon as the first match is found. If we wanted to find every name which matched our input, we could have eliminated

```
:X=CT-1:GOTO 1240
```

from line 1220, and removed line 1230 from the program. Try it and see what happens. Notice that the computer now checks every entry in the array. If more than one match for your entry is found, each one will be printed.

Enter FINDNAME.BAS and save it. Then make a run of this program. This is where the printout of the file's contents comes into play. When asked for a name to look up, be sure to enter the first name on the list. See how quickly it's found? Now enter the last name on the list. Notice how much longer it takes to find the information. This is a very graphic illustration of what happens when we use a sequential file and the data is near the end of the file. We have to look at everything to get what we want. It isn't too bad here because we used a small file and had it in RAM. If it had been a large file on a disk, and many records had to be read, seconds would have become minutes.

OK, we've got a problem. So how do we solve it? If you followed my last series of articles, you know that a binary search of a sorted file is the fastest way to find data. So, the answer is to sort the arrays, then write a binary search routine. We discussed sorting in the April and May 1984 issues of REMark on random files, so the theory won't be covered in detail here. We'll use a bubble sort. It's not the fastest sort, but with a hundred records or less, and the fact that it's the easiest sort to write, it will be fine. We'll add the sort to our FINDNAME.BAS program. Load that program and add the following lines to it, then save the resulting program as SORTFIND.BAS. We'll write over the search routine.

```
2 ' **** SORTFIND.BAS
1200 Y=0          'SET FLAG TO ZERO
1210 FOR X=1 TO CT-2
      'LOOP THROUGH THE ARRAY MINUS 1 ELEMENT
1220 IF (LN$(X)+NM$(X)) > (LN$(X+1)+NM$(X+1))
      THEN SWAP LN$(X),LN$(X+1):SWAP NM$(X),NM$(X+1):
      SWAP MI$(X),MI$(X+1):SWAP PN$(X),PN$(X+1):Y=1
      'IF OUT OF ORDER, SWA
1230 NEXT X       'CONTINUE THROUGH ARRAY
1240 IF Y=1 GOTO 1200
      'IF ANY SWAPS WERE MADE, DO IT AGAIN
1250 PRINT "SORT DO:PRINT     'MESSAGE AND BLANK LINE
```

Add the lines above and type 1205 <CR> and 1260 <CR> where <CR> is a carriage return. This will remove lines 1205 and 1260. Then save as SORTFIND.BAS. All that is left is to add the binary search routine. We covered the binary search in great detail in the random file series, but again, the correspondence I received indicates a need for a full discussion.

If we have a deck of cards, each with a name on it, and each sorted by the last name, the quickest way to find the card we want would be to look at card 50. If it's not a match, it is either too small or too large. In either case, we can throw half the deck away, leaving us with a deck of 50 cards. This time take the 25th card. Again, throw half the deck away and keep only 25 cards. Continue this process of looking at the middle card and throwing half the deck away until there's only one card left. That's the card we wanted, or a match never existed at all.

We could have gotten lucky and picked the card we wanted right away, but suppose we didn't. It can be mathematically proven that the most cards we'd have to look at is equal to the lowest power of two which is equal to or greater than the number of cards we have. Add one to that power, and that's it. In other words, if we have ten cards, 2 to the 3rd power is 8 and two to the 4th power is 16. Four is the lowest power of 2 that is equal to or greater than 10. Add one, and the most cards we would have to look at in a deck of ten cards is 5. That's not very impressive, but as files grow in size, it gets better and better. Each time a file doubles in size, we only have to check one more record. The price we pay for this is that the file must be sorted, and we need random access to it. This means that the sequential file must be read into an array and sorted. That's what the first part of SORTFIND.BAS does.

The following table will show graphically the advantage of the binary search routine we're about to write.

Number Of Records In File	Maximum Number Of Comparisons
1	1
2 – 4	3
5 – 8	4
9 – 16	5
17 – 32	6
33 – 64	7

65 - 128	8
129 - 256	9
257 - 512	10
513 - 1024	11
1025 - 2048	12
2049 - 4096	13

As you can see, it gets better as the file gets larger. Because our file is sorted by last name, that's the only field we'll search. You'll find that your data is returned to you very quickly, no matter what name you request. Again, only the first match is given. The binary search routine looks like this.

```
1300 INPUT "LAST NAME TO SEARCH FOR ";SR$
       'ENTER SEARCH REQUEST
1310 IF SR$="" GOTO 1500          'ENTER <RETURN> TO EXIT
1320 BT=1          'SET BOTTOM OF SEARCH RANGE
1330 TP=CT-1       'SET TOP OF SEARCH RANGE
1340 IF BT > TP GOTO 1430         'DATA NOT IN FILE
1350 RN=INT((BT+TP)/2)            'SET MIDDLE OF RANGE
1360 'IF RECORD IS TOO SMALL, MOVE BOTTOM UP AND TRY AGAIN
1370 IF LN$(RN) < SR$ THEN BT=RN+1:GOTO 1340
1380 'IF RECORD IS TOO BIG, MOVE TOP DOWN AND DO IT AGAIN
1390 IF LN$(RN) > SR$ THEN TP=RN-1:GOTO 1340
1400 PRINT NM$(RN);" ";MI$(RN);" ";LN$(RN);" ";PN$(RN)
       'PRINT THE DATA
1410 PRINT        'PRINT A BLANK LINE
1420 GOTO 1300   'BACK FOR ANOTHER SEARCH REQUEST
1430 PRINT "NO ENTRY IN FILE FOR ";SR$'ERROR MESSAGE
1440 PRINT        'PRINT A BLANK LINE
1450 GOTO 1300   'BACK FOR ANOTHER SEARCH REQUEST
1500 END
```

Add lines 1300 thru 1500 to the SORTFIND program and run it. As it asks for a name to look up, enter the last name from your file. Try it several times with different names, and you'll see that all answers come back very quickly. Also try to enter names for which there is no data in the file. This will check out the error routine and prove that the binary search can eliminate all possibilities in a big hurry.

Thus far in the series, you've learned to create, add to, sort, and search sequential files. Now that the file is sorted, it would be a real shame to throw it away without saving the data in its correct order. To do this, all we've got to do is open a temporary file and write the contents of our array to it. We'll use a separate file name so we can list both our original input and the sorted file on our printers using the CP/M LIST command. We'll add our routine to the SORTFIND.BAS program beginning at line 1500.

```
1500 OPEN "O",#1,"SORTED.DAT"    'OPEN A FILE FOR OUTPUT
1510 FOR X=1 TO CT-1     'SET UP A LOOP
1520 R$=LN$(X)+"\"+NM$(X)+"\"+MI$(X)+"\"+PN$(X)
       'SET UP RECORD
1530 PRINT #1,R$           'WRITE RECORD TO FILE
1540 NEXT X       'CONTINUE TO END OF ARRAY
1550 CLOSE #1    'CLOSE THE FILE WHEN FINISHED
1560 END
```

Add these lines to the SORTFIND program and save them. Run the program. When you enter a RETURN only as a request to look up data, a new file SORTED.DAT will be written. Exit to your operating system and use the LIST command to print copies of SORTED.DAT and SEQFILE.DAT. Then compare the two files.

Next month, we'll wrap up this 5-part series with conversion to and from random files. We'll also discuss changing and deleting files. See you then.

USS Fast Attack

Simulation Becomes A Fine Art

Ralph F. Rumpf
6036 Legion Rd.
Stevensville, MI 49127

Simulate — To assume or have the appearance or form of, without the reality.

The above is a definition taken from our home dictionary. To me it is an accurate description of the newest game from Interdiscipline Corporation: USS Fast Attack.

If you are a lover of fast action, shoot 'em up arcade type games, then USS Fast Attack is not the game for you. If, on the other hand, you are more inclined towards a thinking game where you must always be at least one step ahead of your opponent, then read on.

Simulations are common on larger, more expensive computer systems, where they are used to evaluate solutions for a great number of problems, from spacecraft orbits to economic trends to thermonuclear war. The complexity and depth of most simulations require the speed and processing power of large, powerful and expensive computer systems. Interdiscipline has managed to bring this type of simulation to the world of microcomputers. The USS Fast Attack program is not as fast or powerful as a simulation on a larger computer, but it is a full scale simulation, none the less.

The Program

The USS Fast Attack simulates the operation of a modern attack class submarine. The primary modules for the program involve the transit (navigation), battle stations and weapons control (fire control) systems. In order to accomplish these tasks, the program uses several sophisticated screens that take full advantage of the graphics capabilities of the Z-100. The screens are complex and display a great deal of information in a small amount of space. Because a number of colors are used in these displays, I recommend that a color display be used with this program, since some of the data is difficult, if not impossible, to view on a monochrome display.

On a large scale computer system all stations of the submarine would be manned by individual crew members. Positions such as the navigator, helm, weapons officer, etc., would be operated by individuals whose actions are coordinated by the computer system that controls the simulation. Since this cannot be accomplished on the Z-100, you will man all positions necessary for the operation of your submarine.

I would like to make it clear from the start that even though this is not a simulation created on a larger computer, as I said, it is still a simulation. Like all simulations, USS Fast Attack is complex and involved. It is not the kind of thing that you can jump into without reading the manual at least once before you start. In fact, you will probably refer to the manual many times as you play the game and constantly find new ideas to try as you play.

The navigation portion of the program requires you to utilize the on-board systems to maneuver the submarine to a Mission Reference Point (MRP), where possible enemy vessels are presumed to be located. This is generally accomplished by use of the control systems to navigate, trim the submarine, maintain your depth and use the tracking systems to reach your destination. Among the devices available to locate your targets are radar and sonar, as well as the normal periscope.

One of the first things to be accomplished on any mission is trimming the vessel. A normal submarine requires a very complex system of weights and counter-weights to maintain its buoyancy in the water. Experienced personnel keep the system in optimum balance thus maintaining efficient vessel operation. In the case of the USS Fast Attack simulation, you maintain the trim system by pumping sea water into the various tanks to provide the vessel with the proper pitch. The purpose of the trim adjustments is to efficiently counter-balance the vessel for the weight of weapons and personnel taken on in the mission. These selections are made from the Trim System display. Here you can select which tanks of the four provided you want to work with and how much water you want to move into each tank. Proper trim is very important to the operation of the vessel. You will find it difficult to maintain your operational depth unless the vessel is in trim. Once you have the vessel in proper trim, then you are ready to proceed to the Mission Reference Point (MRP). If adjustments to trim are necessary, you can make them while you are under way.

As you approach the MRP, you will need to prepare the submarine for combat. You accomplish this task with the Battle Stations module, but you can only do this within a certain distance from the MRP. The battle stations module allows you to select weapons, or electronic counter measures if they are on board, load and prepare the weapons for use. You can also operate the sonar system from this module to track suspected targets. Once you have maneuvered to within striking range of your target, hopefully undetected, you are ready to launch your weapons using the Fire Control Module.

The Fire Control Module allows you to select, arm and launch your weapons. Essentially, this module is similar to the battle stations module, except that you can actually launch your weapons and you have some additional capabilities. The Position Keeper becomes active in this module as does an enhanced plot feature. The position keeper allows you to input data for the torpedo control system, this then allows the system to track your targets for accurate weapons deployment. The plot feature allows you to track up to six independent targets, a very handy feature for some of the mission assignments.

On the surface that sounds like a pretty simple game, but looks can be deceiving. There are a number of other features that come into play as you pit yourself against the features of this simulation. On one hand there is the enemy. Obviously, if you employ poor tactics or make a mistake, then you are going to be detected. If you are, you are in trouble. The enemy will go all out to eliminate you and your vessel. When this happens, you are very likely to incur damage, another problem, to your ship. Among the likely candidates for problems are breaches of the hull causing flooding, electrical failures and weapons failures. Depending on what you are doing, none of these are limited to situations where you yourself are under attack. For example, the maximum test depth of your vessel is 1000 feet, go deeper and anything can happen. However, if you are seeking a thermal layer to avoid sonar detection, then you often have to take chances. Another feature that may cause you some headaches is the fact that surface vessels are not the only types of enemy vessels you may have to deal with. There are also a number of enemy attack submarines, similar to your own, that are perfectly capable of destroying you before you ever knew there was anything to worry about. As the situation warrants, you may be called upon to abort weapons launches, repair emergency damage, charge the air tanks, reload weapons under attack, blow ballast and emergency surface. In some situations you will have very little time to make the correct response, so you have another challenge to meet. Using a thermal layer for cover to sneak beneath a convoy is a real challenge!

The Manual

The manual for USS Fast Attack is a moderately well done manual containing sufficient information to allow you to succeed in the simulation. I found a few typos in mine, but they are insignificant and do not affect play. I also found the manual to be fairly well organized and presented in a logical manner. Among the types of information contained in the manual are descriptions of the controls and instruments, descriptions of the various tasks (i.e. transit, battle stations, etc.), weapons and vessel performance specifications and a mini-course in submarine dynamics. Some of the information is a bit deep and may require more that one reading to grasp the concepts that are being described. I found this to be true regarding the submarine dynamics and the description of the position keeper and its use. Like the simulation itself, the manual provides a lot of information in a short space, approximately 36 pages. I recommend that you read it at least once and keep it handy when you play. The folks at Interdiscipline must have thought it worth your while to put that much information together in one spot, so I imagine it would be wise to use it, I know mine is a bit ear-marked.

Comments

Basically, I am a shoot 'em up person, but I found that USS Fast Attack offered a challenge of another sort. Hand and eye coordination are important, but the mind has to be fed too. The game is not likely to become boring for a while, as long as you view it as a simulation. That means that things are not designed to keep you glued to the edge of your seat with excitement. As any good chess player can tell you, there is a lot of excitement in waiting out your opponent. There is a lot of "cat and mouse" in USS Fast Attack, and that adds to the realism of the simulation. Modern naval warfare is not a shoot 'em up proposition, especially in submarine warfare. I would imagine that it can become very tense hanging 500 feet below the ocean's surface waiting to see if you will ever come up again, as a destroyer or cruiser passes overhead.

Some things are slow to happen and tend to drag on, but this is a function of the simulation, you can only do so much at one time. The program is obviously a compiled BASIC program as the speed of the display and some operations attest to, none the less, it presents a solid challenge to almost any armchair admiral.

I feel that Interdiscipline did a very good job in utilizing the graphics capabilities of the Z-100 for USS Fast Attack. Of course, when you have a machine like the Z-100, you always want a little more. I would have liked more dramatic displays when a torpedo struck the target, I guess I'm just overly picky. Also, I noticed that whenever I did launch a torpedo at a target, it only took one to sink the enemy ship. (I know that I am good, but even I don't believe that I am that good!) I don't know if that is a function of today's weaponry or what, but it would be interesting to have a damaged ship being protected by a number of other ships. I was also somewhat disappointed by the use of color for the game. I like color displays very much and the color displays for this simulation are great. But I don't own a color monitor as yet and some parts of the game can become a bit tricky to discern on a monochrome screen, especially when you are trying to determine who is who on a six target sonar plot!

I guess that I can't close without a small comparison to GATO, a submarine type game for the Z-100 PC, I reviewed some months ago. I think I miss the little touches that GATO added to the game. GATO included a number of islands that were visible on sonar and through the periscope as you passed them. Also, if you ran out of torpedoes or fuel or were damaged, you had to locate the sub tender to resupply and have repairs made. I realize that the concepts of the two games are markedly different (GATO was based in the Second World War), I still think that small touches such as that would add more realism to the simulation as a whole. Still, all in all, USS Fast Attack is an excellent simulation that I will certainly be playing again and again.

USS Fast Attack is available from:

Interdiscipline Corporation
403 S. Brandon
Seattle, WA 98108
(206) 763-2099

USS Fast Attack is provided on two soft-sectored diskettes for the Z-100 that are not copy protected. Since they are nice enough to provide them in unprotected format, the least that we can do is not make illegal copies. Hardware requirements for the system are:

32K Color Ram Monitor
Rom 1.2 or higher
Z-DOS or MS-DOS
128K System Ram Two disk Drives

The cost of USS Fast Attack is $39.95, not a bad price for this much simulation.

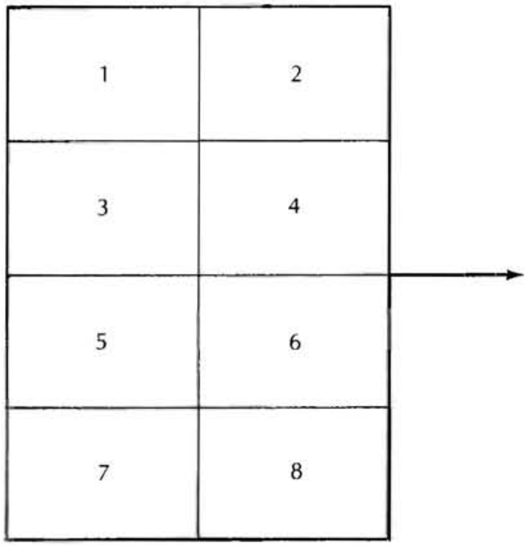
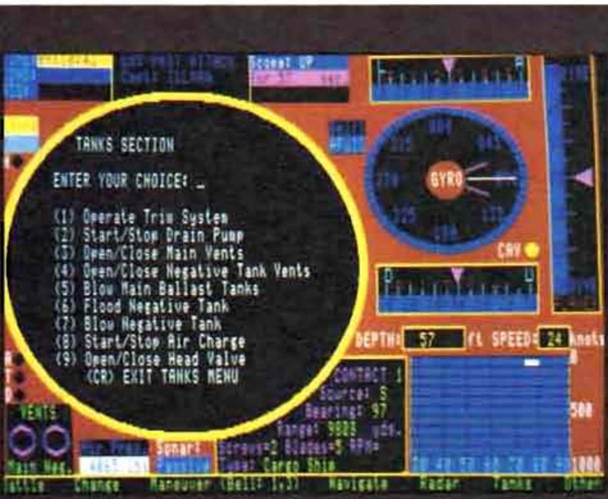


Photo #	Description
1	Tanks Selection Menu. From here you can operate the trim system.
2	Trim System Display. Used to trim the vessel.
3	Position Keeper Display. Used in final target tracking prior to weapons launch.
4	Navigation Plot. Indicating your vessel, suspected enemy contacts and approximate ranges.
5	Radar Plot. Showing the target (blue) and your vessel (white).
6	Weapons Display. From here you arm and load the various weapons available for the mission.
7	Visual contact with the target vessel.
8	Torpedo launch and a hit on the target vessel.

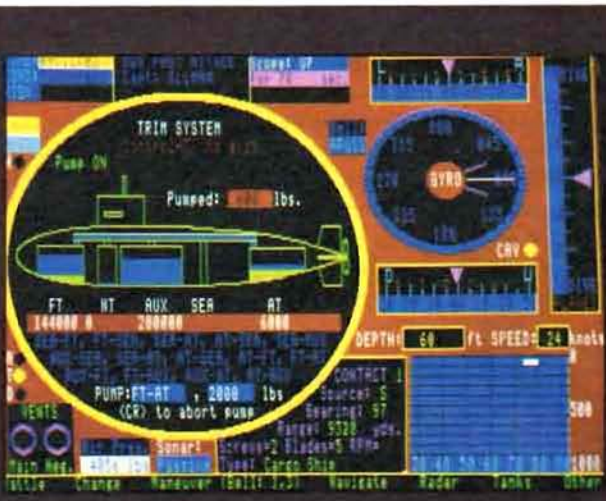


```
TANKS SECTION

ENTER YOUR CHOICE: _

(1) Operate Trim System
(2) Start/Stop Drain Pump
(3) Open/Close Main Vents
(4) Open/Close Negative Tank Vents
(5) Blow Main Ballast Tanks
(6) Flood Negative Tank
(7) Blow Negative Tank
(8) Start/Stop Air Charge
(9) Open/Close Head Valve
   (CR) EXIT TANKS MENU
```

DEPTH: 57 ft SPEED: 24 knots

CONTACT 1
Source: S
Bearing: 97
Range: 9809 yds.
Screws=2 Blades=5 RPM=
Type: Cargo Ship

Scope: UP
GYRO
CAV
VENTS
Main Neg.
Sonar:
500
1000
Battle Change Maneuver (Bell: 1,3) Navigate Radar Tanks Other

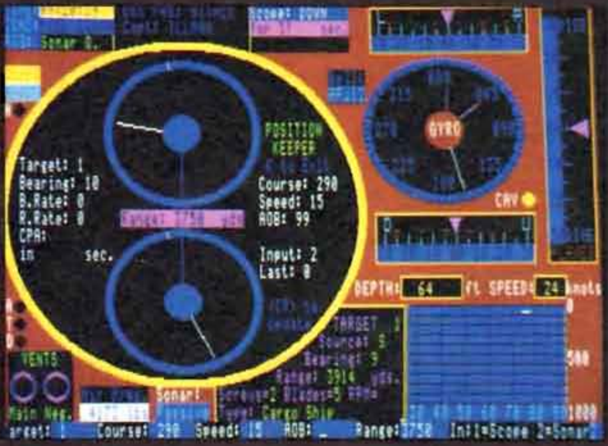


```
TRIM SYSTEM
Pump ON
Pumped:      lbs.
FT     NT    AUX   SEA      AT
144000 0     200000         6000
PUMP:FT-AT  , 2000  lbs
   (CR) to abort pump
```

DEPTH: 60 ft SPEED: 24 knots

CONTACT 1
Source: S
Bearing: 97
Range: 9320 yds.
Screws=2 Blades=5 RPM=
Type: Cargo Ship

Scope: UP
GYRO
CAV
VENTS
Main Neg.
Sonar:
500
1000
Battle Change Maneuver (Bell: 1,3) Navigate Radar Tanks Other

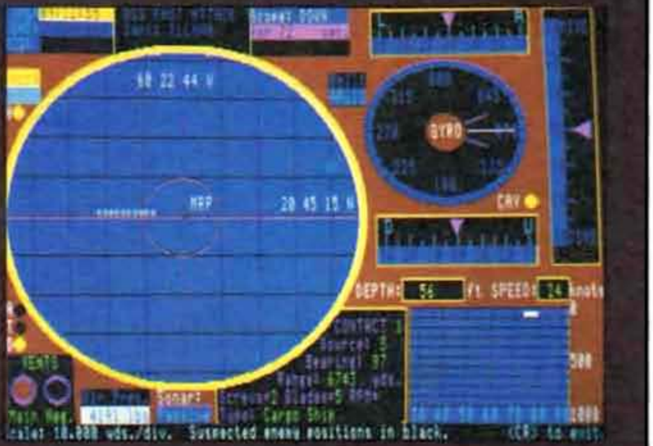


```
POSITION
KEEPER
Target: 1
Bearing: 10
B.Rate: 0
R.Rate: 0
CPA:
in        sec.
Course: 290
Speed: 15
AOB: 99
Input: 2
Last: 0
```

DEPTH: 64 ft SPEED: 24 knots

TARGET 1
Source: S
Bearing: 9
Range: 3914 yds.
Screws=2 Blades=5 RPM=
Type: Cargo Ship

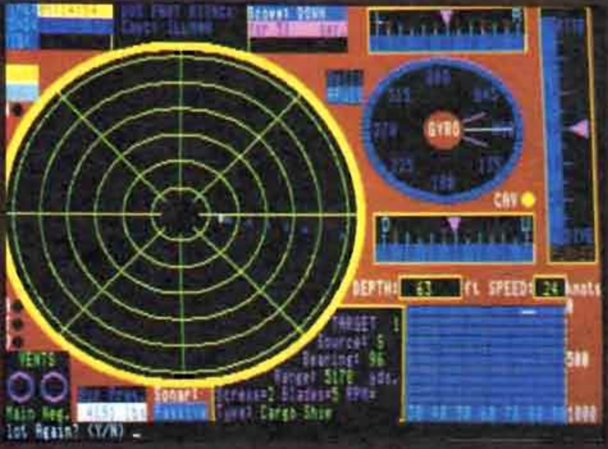
```
arget: 1   Course: 290  Speed: 15  AOB: _   Range:3750  In:1=Scope 2=Sonar
```



DEPTH: 56 ft SPEED: 24 knots

CONTACT 1
Source: S
Bearing: 97
Range: 6743 yds.
Screws=2 Blades=5 RPM=
Type: Cargo Ship

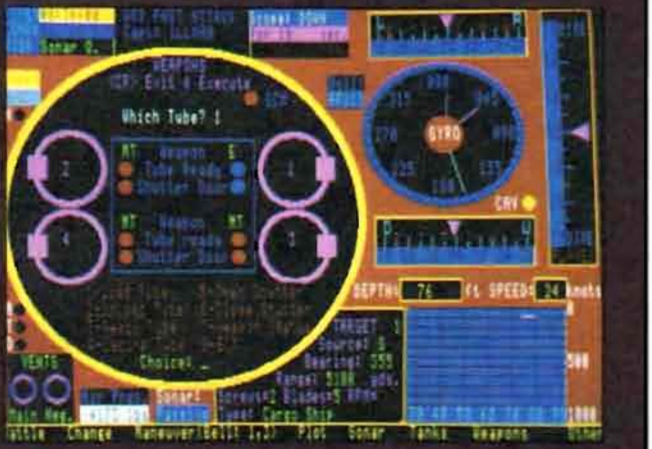
```
cale: 10,000 yds./div.  Suspected enemy positions in black.      (CR) to exit
```



DEPTH: 63 ft SPEED: 24 knots

TARGET 1
Source: S
Bearing: 96
Range: 5170 yds.
Screws=2 Blades=5 RPM=
Type: Cargo Ship

```
lot Again? (Y/N) _
```

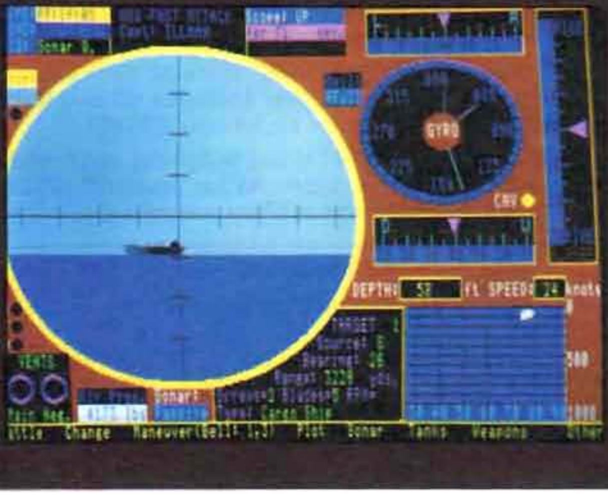


```
WEAPONS
(CR) Exit & Execute
Which Tube? 1
MT   Weapon   E
     Tube Ready
     Shutter Door
MT   Weapon   MT
     Tube Ready
     Shutter Door
Choice: _
```

DEPTH: 76 ft SPEED: 24 knots

TARGET 1
Source: S
Bearing: 359
Range: 5100 yds.
Screws=2 Blades=5 RPM=
Type: Cargo Ship

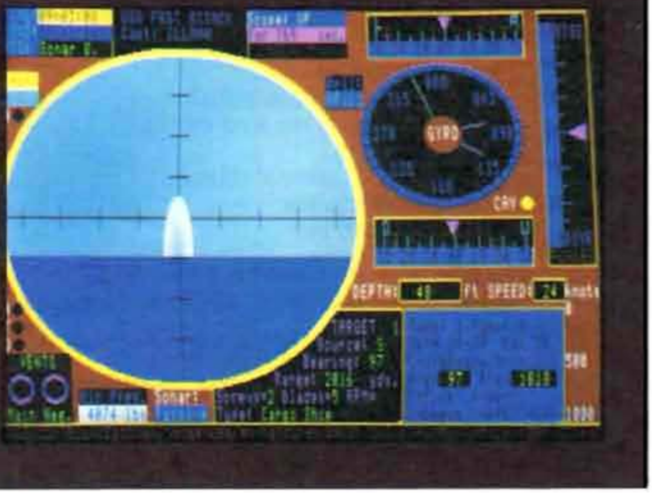
Scope: DOWN
Sonar 0.
2
1
4
3
GYRO
CAV
VENTS
Main Neg.
Sonar:
500
1000
Battle Change Maneuver(Bell: 1,3) Plot Sonar Tanks Weapons Other



DEPTH: 52 ft SPEED: 24 knots

TARGET 1
Source: S
Bearing: 26
Range: 3228 yds.
Screws=2 Blades=5 RPM=
Type: Cargo Ship

Scope: UP
Sonar 0.
GYRO
CAV
VENTS
Main Neg.
Sonar:
500
1000
Battle Change Maneuver(Bell: 1,3) Plot Sonar Tanks Weapons Other



DEPTH: 48 ft SPEED: 24 knots

TARGET 1
Source: S
Bearing: 97
Range: 2816 yds.
Screws=2 Blades=5 RPM=
Type: Cargo Ship

Controlled Data Recording Systems Inc.

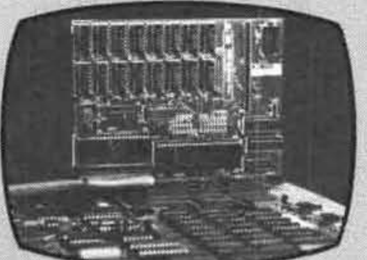
Quality Products and Support for the Heath/Zenith Community

New for the H/Z89-90 Computer Users:
SUPER RAM 89

Heath/Zenith 89 and 90 computer users can now get the speed and power of a high capacity Ram Drive System at a reasonable price.

Using the Super Ram 89 package, standard software shows an immense improvement in speed. Depending on the software being run, programs may execute 10 times faster than when run through standard floppies.

Super Ram 89 comes as a two card set that plugs into the left hand (16k expansion) side of the computer. No computer modifications required.



Board 1 has two banks of 256k chips possible for a total of 512k. Either or both banks are able to use 64k chips instead. Board 1 can be used by itself, with board 2 added at a later date.

Board 2 has an additional 512k, plus it has a real time clock capability, and a SASI interface hardware capability. Board 2 piggybacks onto board 1.

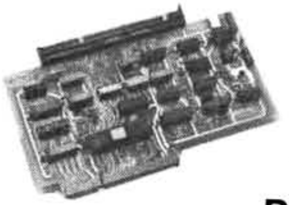
The Ram Drive Software (SRAM) allows one or two logical ram drives. The ram drive(s) can be located starting anywhere from logical A: to O: (standard drives get relocated). SRAM can be set to start at logical A: and warm boot with ram (no floppy disk accesses needed). Ram drive attatches to any of the versions of CP/M 2.2 bios used in the H/Z89-90.

Super Ram 89 Pricing:

Board 1 includes hardware manual and ram drive software with no ram: $190.00. Each 256k bank, add $90.00. Board 2 (must have board 1). With no ram, no clock, no SASI $90.00. Each 256k bank, add $90.00. Ask about clock, SASI pricing.

FOR THE H/Z89-90 COMPUTERS

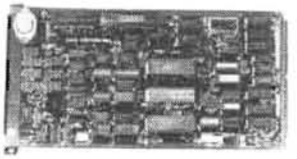
THE ONE CONTROLLER FOR 8" & 5.25" DRIVES THE FDC-880H PRICE:$395



Includes controller board CP/M boot prom, I/O decoder prom, hardware/software manuals BIOS source listing. HDOS driver now available for $50.00.

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. The FDC-880H operates with or without the Heath hard sectored controller.

THE FDC-H8 DOUBLE DENSITY 8" AND 5.25" CONTROLLER FOR THE H8 COMPUTER PRICE $495



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

- 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.

MODIFY89

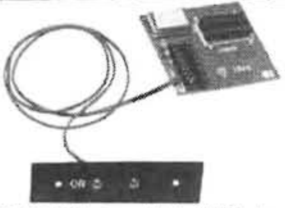
Allows owners of the FDC-880H or the FDC-H8 to read/write to the following disktypes:

Access S.S.D.D., Cromemco S.S.D.D., DEC VT180 S.S.D.D., IBM PC/Zenith 100, (CP/M) D.S.D.D.*, Kaypro II S.S.D.D., Morrow Micro Decisions S.S.D.D., NEC PC-8001 A S.S.D.D., Osborne S.S.D.D., Osborne S.S.D.D., Otrona D.S.D.D.*, Zorba D.S.D.D., TI Professional S.S.D.D., TRS-80 Model I (Omnikron CP/M), TRS-80 Model III (MM CP/M), Xerox 820 S.S.D.D., Zerox 820-II S.S.D.D. Plus more, Standard (Tests for H/Z37 and C.D.R. Disk types)

PRICE $49.95

FOR THE Z100 COMPUTERS

THE ORIGINAL Z100 SPEED MODULE. RUN YOUR Z100 PROGRAMS FASTER



The ZS100 runs the Z100 CPU 50% faster, (7.5 MHZ) in 8088 mode. The ZS100 installs easily with no soldering. The ZS100 is externally switchable between speed mode and normal. The ZS100 improves the time performance of applications packages with no software modifications needed. The ZS100 is for all Heath/Zenith 100, 110 and 120 series computers.

New: a powerful floppy drive package for any computer that can use 8" floppy drives.

Two extended technology 5.25" drives in a package with interface cables case and power supply with the capability of looking exactly like two standard 8" drives to the computer.

Now have **2.4 megabytes of 5.25" floppy file storage capacity** in one package. A switch enables the drives to look like 8" drives or 5.25" 96 TPI drives.

ASK FOR PRICE.

For information about these products contact:
Controlled Data Recording Systems Inc.
7210 Clairemont Mesa Blvd.
San Diego, Ca. 92111 Phone (619) 560-1272 or contact a Heath/Zenith dealer that carries the C.D.R. Systems Inc. line of products.

A Parallel Printer Buffer For The Z–100

Dr. Michael Scott
3554 W 22nd Ave.
Vancouver, B.C.
V6S 1J3

A printer buffer is a marvelous thing to have, and now that I have one, I don't know how I ever lived without it. The original idea and most of the credit must, of course, go to John Bono, who designed it and wrote it up in BYTE (June 1984, p. 142), but I'll describe my implementation for the Z–100, which makes a few changes, offers a few tips and corrects some typos. My buffer was made on an S–100 size piece of PCB and uses the +8V, GND, and utility clock signals already present on the bus of the H–100. I had neither the knowledge nor the courage to attempt to use the bus pins for access to the data lines, and I think it's probably safer not to. If you disagree, you obviously don't need to read this article!

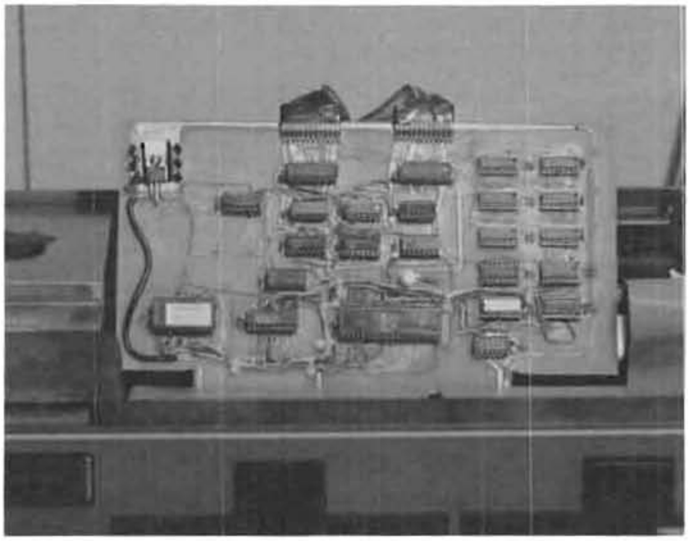
My printer is a Roland R1010, which means Epson MX 80, but any Centronics compatible interface should work.

A printer buffer is basically a bank (here 64K) of RAM, which is isolated from the rest of the computer and simply stores text files sent by the machine, then sends them to the printer. The advantage is that since moving around chunks of data from one memory cache (the computer) to another (the buffer) is very fast, you can send a large text file in a matter of seconds, and then be using the computer again while the buffer chugs along handling the much slower operations of the printer.

Some experience and hardware skill is necessary, but not a vast amount — after all, I am strictly an amateur myself. An oscilloscope is handy, and some kind of simple pulse-catcher would be helpful. If you are careful, and lucky, your version may run off–the–board first time, and since the design works, you could troubleshoot it by simply removing all the ICs, including the 7805, and doing some continuity testing. Beware of the kind of sockets you use. Some cheapies have terrible contact designs, and can lead to lots of puzzling headaches. A connection that is good on the solder pad may not be at the pin when the IC is plugged in!

There is no need to repeat all the information in the BYTE article where you will find the schematics, parts list and the listing for the software, but you might want some preliminary information before deciding to hunt for that issue of BYTE, only to discover that it looks too complex, or too expensive.

The design uses a Z80 microprocessor, which is cheaper than a memory chip these days, a 2716, eight 6665As, and about 14 74LS



Component side. Right connector — Input. Left connector — Output. (Pins 1, 20, 49, 50 of S100 used)

series TTL chips. It cost me about $160 CAN to build, but I know it could be done for about $80–$100 US if you shop around, especially for the 6665s, which represent half the total cost. Another item of some possible cost is programming the EPROM, if you don't have the equipment to do it yourself. This cost me $15 CAN.

If you are still interested, here's how to do it:

Get the article and photocopy it, or cut it out so you can refer to it often. Then make the following changes, kindly supplied by John Bono, when I phoned him for advice:

Parts List: Do not use just any old kind of 4164, as implied. Only use those types that have 128 cycle refresh times. The TI 4164s and some other makes have 256 cycling, and just won't work in this design without modifying the software considerably. If you happen to have some on hand and want to use them, then you'll have to research those modifications. The Motorola 6665A is fine and other makes may be too, but check the data sheet before you buy and open the sealed package.

You also do not need the MC4024 clock and its associated discrete components, since you will be using the 2 MHz unsynchronized utility clock signal on pin 49 of the S–100 bus. You will need an extra 74LS00 if you want to try running at the full 2 MHz.

The power supply is taken from the bus, so all you'll need is a 5V regulator and an extra 1uF tantalum capacitor. (Figure 2) I also used one .01 uF bypass capacitor per chip, (i.e. 24 capacitors rather than 18), but that's maybe being obsessional! You will also need nine extra 330 ohm resistors, for pull-ups.

The connectors depend on how you want to do it. You need to take twelve lines from connector J3 on the main board, route them somehow to your buffer board which sits in the card cage, and then take eleven lines from the buffer board and connect them to your printer. I unsoldered the whole plug at J3 and soldered in a ribbon cable at holes 1 to 12. A female card-edge connector at the end of this ribbon plugs into a male right-angle PCB connector on the buffer. (The Molex KK .100 series components would be ideal, but I couldn't obtain them at the time). A similarly attached ribbon cable goes from the board to a female panel-mount DB25 connector at J13, where my standard printer cable now plugs.

As far as the board is concerned, I used double-sided PCB with a direct etching technique, but if I had to do it again I'd use single-sided board, put down traces for the 5V and GND lines, and use point-to-point wiring. Troubleshooting traces under firmly soldered sockets on the component side is a real pain, to say nothing of soldering inaccessible connections. Double-sided PCBs are great if you have the facilities for making plated-through connections. Other possibilities might be some of the ready-made (and expensive) S-100 breadboards, like the VECTOR 88 series, wire-wrap or solder — make sure you don't confuse the two types.

Typos In The Schematics

Page 450 (top) IC14 should, of course, be the other part of the 74LS367 on page 452. The data lines D0-D7 on the left are numbered incorrectly, though that doesn't really matter. And while you are about it, draw a box around IC1, the MC4024 clock, and its associated components.

Changes in the schematic include a 330 ohm resistor connected between pins 6 and 11 of the Z80 (to pull-up the clock), and eight similar resistors between 5V and each data output line. Also, you will be using on-board voltage regulation, (as per Figure 2) rather than the separate power supply suggested.

There are too many typos in the hex codes of the program listing to be worth correcting, but the assembly code is fine. I'll get to that part later.

The first thing to do is get the programming of the EPROM done, if you are getting it done for you. Then it might be ready when your board is. If you do get it commercially programmed, then insist on a printout of its contents and check it byte-for-byte against the Hex codes supplied by your assembler. If the 2716 is correctly programmed, then that's one thing you don't need to consider if you have trouble later. Since I don't have a Z80 assembler, I rewrote it in 8080 assembly language and handed it to the programmer in the form of a .PRN file. I had to do this on a friend's H89 because although I have CP/M, I do not have a text editor for it. (I am a bit annoyed that my old PIE editor will not work on the Z-100, and see no reason why I should pay Software Toolworks a third royalty on PIE, having already purchased it for the H89, and again for Z-DOS). Presumably you could do the program in 8085 ASM code with CP/M, though I don't know much about it, and another advantage would be that your CP/M .HEX file will be Intel standard, and the store will be able to take it directly off the disk. (Usually only 8" though.)

If you want to wire in your cable as I did to avoid messy cables snaking in and out of various slots in the back, then choose some convenient time (when you are not likely to need to use it for an hour or so) to take it apart and unsolder the plug at J3. This is quite a job, and requ.res lots of solder-wick! Then solder in your 12-strand ribbon at pins 1 through 12, mount your DB25 connector at J8 or J13, and temporarily jumper the free (buffer-board) ends of the cables. Your printer should still work perfectly, and you will be reassured that even if there is some delay in getting your buffer to work, then you can still use everything.

Making The Board

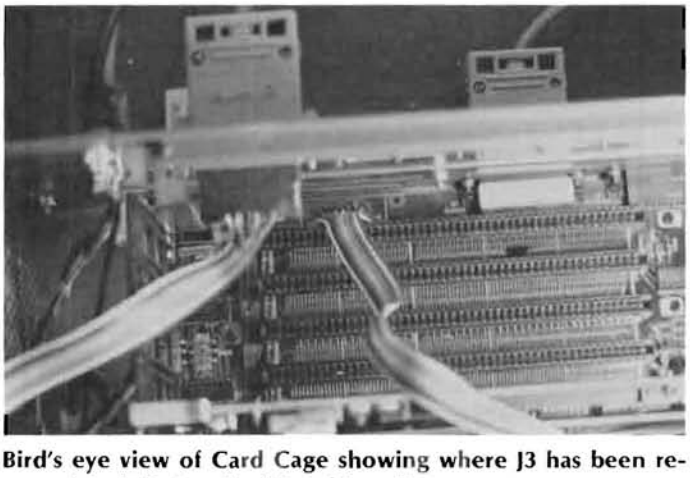
Choose a technique you are happy with and get busy! You need only have edge contacts for a few of the pins (from among pins 1, 20, 49, 50, 51, 70 and 100) on the S-100 bus connector. See the S-100 specification booklet which came with your H/Z-100 (Technical Manual, Vol 2, Appendix A1), and the X ray diagram of the Floppy Disc Controller is handy for checking your positioning of the contacts if you are making your own PCB.

The pins supply:

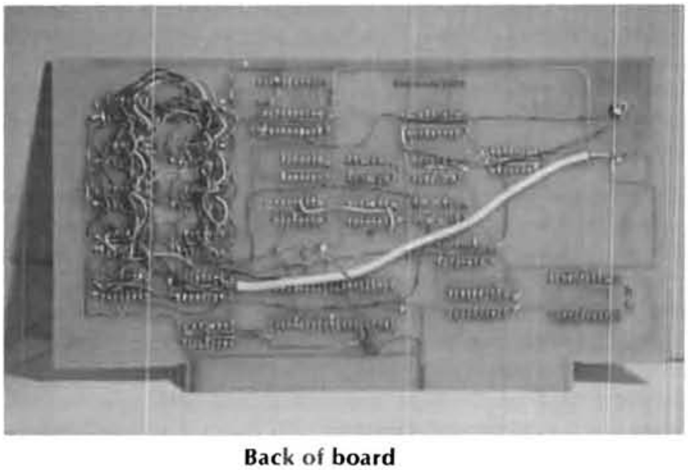
1 (and/or 51)	8V
20 (and/or 70)	GND
50 (and/or 100)	GND
49	2 MHz clock

It is up to you how many and which ones of the four possible GND pins you want to use, but it would be wise to have lots of wide GND traces with solid interconnections to minimize noise and ground loops. I used only pins 20 and 50 myself.

If you are making your own board, trim and sand the edges, etc.,



Bird's eye view of Card Cage showing where J3 has been removed and 12-strand cable soldered in.



Back of board

Figure 1A

Clock circuit using 1/2 IC24 (74LS74)
This gives a clock of 1 MHz

```
                   o 5V.
                   |
      ---------
     |1     --|----
     |     |  |----
     |     |--|    |
     |     |11|----|-----< From Pin 49 of S1ØØ bus.
     |      --|    |
     |       9|----|-----> To pin 6 of Z8Ø (1MHz)
  ---|________|----
  |
 GND
```

Figure 1B

Clock circuit using extra 74LS00.
This gives a clock of 2 MHz.

```
                                o 5V
 From S1ØØ pin 49________       |
>----------------|1        |---
               --|2        |
<----------------|3        |
  To Z8Ø pin 6   |         |
  (2 MHz)        |         |
                 |         |
              ---|_________|
              |
             GND
```

Figure 2

Power supply.

```
                          ----------
                         |          |
 8V  ____________________|  uA78Ø5  |________________  5V
           |          IN |          | OUT
           |             |__________|
         --|--                 |
           |  1.Ø uF           |
           |  Tantalum         |
           |                   |
 GND_______|___________________|____________________ GND
```

then check that it fits properly in the card cage, and that the alignment of the contacts is correct BEFORE you etch!

The layout is not critical, but you might wish to look at the assignment of the individual buffers in ICs 9 and 15, because swapping some of them around might simplify drawing traces if you are going that route. I used the assignment in the original article, but I dare say it could be improved upon. After soldering in all the IC sockets, and before soldering in the 7805 or the reset capacitor, check all your connections thoroughly at the socket pins on the IC side for all the usual errors, omissions and mistakes. Check them again. Check for infinite resistance between 5V, GND and 8V contacts. A not too hefty 8V line is not a bad safety feature, since it will fuse before your Z-100, if you short the 5V line to ground at some stage in fiddling about with trying to get test-clips onto various pins.

Solder in the 7805, its 1uF filter capacitor, (note: polarized!) and whichever IC you are using to supply the clock to the Z80 (i.e. IC24 or the extra 74LS00). Check that a good, clean square wave of the chosen frequency appears at pin 6 of the Z80 socket. The clock signal must also be as close to its full 5V amplitude as it can be, which is the reason for the 330 ohm pull-up resistor between pins 6 and Vcc (11) of the Z80.

If things are OK so far, and the 7805 isn't hot because of an undetected short to ground, solder in the remaining components and insert the rest of the ICs, taking the usual antistatic precautions with the Z80 and the memory chips. Make sure all the pins are inserted and not bent. Some final continuity checking can be done now with a low-voltage ohmmeter between the actual IC pins, but remember that with the ICs inserted, there will be some apparent 'leakage' between 5V and GND connections. Then insert your preprogrammed 2716, check the board visually again and try it out. I used one of the further back slots in the cage, to enable me to get scope clips onto pins of interest.

CAUTION: If you are not familiar with or comfortable working with a soldering iron and printed circuit boards, **DO NOT ATTEMPT** this modification, but refer it to a qualified technician.

If you don't get anything at all, switch off the computer and check for hot chips. (Oops! a 6665 in backwards?) Use your oscilloscope or pulse-catcher to check that something is happening at all the input and output pins. Check for noise and glitches on the 5V and GND lines. (Use AC coupling and high sensitivity. If there is noise, check to see if it is no more than there is on the Z-100 lines without your board in place. There is about 10 mV of high frequency stuff on mine.) If there is noise, find out why. Check the bypass capacitors for cold-solder joints. Check the data lines for high-resistance shorts.

One problem I ran into was incorrect data being sent from the buffer to the printer on an intermittent basis. I solved this by shortening the cable to a minimum length, and installing pull-up resistors on the data output lines.

Check everything! It'll work! But if you've done all of the above, and you are still having problems, then call in your expert buddy, or crawl down to your friendly Heathkit store and talk the technician into looking at it. Eventually, you will be able to enter

```
PRINT B:BIGFILE.TXT
```

and get your prompt back in just a few seconds, while the printer churns out that 30 page article you wrote for REMark on the problems of trying to make sense of homebrew projects from computer mags.

Final Note: The program John Bono supplies to run the buffer, which is stored in the 2716 works fine, but there are many typos in the hex listing in the article. The Z80 mnemonics are correct, so you can take it from there. The statements referencing "PRACK" can be deleted as they refer to an earlier version.

I also inserted a couple of lines of code to enable a software flush of the buffer, because as it stands you could have the buffer containing 60K of stuff, suddenly realize that you don't want to print any more and there is no means of stopping it until the end, since your CTRL-C (or whatever) will only be processed after the buffer is empty. All I did was decide that I would use CTRL-E (ASCII 05), which is not assigned a function in ZDOS. A test for ASCII 05 is included in each input loop, and when the code is received by the buffer a jump is performed to 0000, effectively flushing the buffer and halting printer operations.

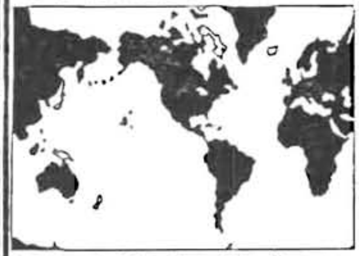
The extra code is inserted just after the "OUT (ACKHI),A" statement in the "Get Character" section, and consists of the following two lines:

```
OUT     (ACKHI),A        ;Previous line

CP      05               ;You may wish to use
JP      Z,0000           ;something else.

LD      (DE),A           ;Next line
```

It was a simple matter to write a tiny assembly language program called STOP.COM, whose only function is to send an ASCII 05 to the PRN device, so that typing STOP <RETURN> at the console will abort any current printing operation. Some people might wish to use a more complex scheme involving a two character escape code, and there is certainly plenty of room in the 2716 for it, though some ingenuity is required to do the programming using only the registers of the Z80 for storage. There's no reason why you couldn't use a bit of the buffer for data storage, and declare a stack segment for this purpose.

Think carefully about the code you assign to flush the buffer, because you may have a pet program which sends the same code to GRAFTRAX, and will be annoyed to find that the buffer traps it. My solution is one I can live with temporarily, until I get around to writing a program to process an "ESC @", which is the conventional reset code for the MX 80.

Good luck, and many thanks from all of us to Mr Bono. ✻



Be a HERO
with your
Computer
Buy One
NOW!
HERO1
HERO1

DOODLER

Steve Howard
8300 115th Street South
Cottage Grove, MN 55016

The scene is a common one. You are sitting at your desk. You are daydreaming about your upcoming vacation or possibly involved in a phone conversation you aren't particularly interested in. A pad of paper lies enticingly on the desk. A nearby pencil beckons you. You can't resist and you finally succumb to temptation and pick it up.

Your hand moves slowly toward the paper and begins drawing unusual shapes and pictures. You are now lost — you have become another Doodler!

Now there is a solution for those of us who are doodle addicts and like to use computers. Doodler, by Paul F. Herman of Safety Harbor, Florida, is a graphics drawing program for the Heath/ Zenith 100.

Doodler is written in compiled ZBASIC for the Z-100 (not the Z-150) series of computers and turns the Z-100 screen into an electronic sketchpad.

The program requires a Z-100 with at least one disk drive and a dot matrix printer with graphics capability, if you wish to print the graphics designs. Two versions are supplied on the distribution disk. One is for systems with 128k and another for systems with 192k or more of user RAM. All three banks of graphics RAM are recommended along with a color monitor to take advantage of the color graphics, but it will also work on a monochrome system.

The program is supplied with a function key template similar to that distributed with Lotus 1-2-3 and makes the program operation a snap. Along with the template, a menu is continually displayed on the 25th line.

There are a few basic capabilities any graphics software should have. The following points are the ones I feel are important to look for in graphics software.

1. Installation program for various hardware configurations
2. Shape drawing assistance
3. Automation of line drawing
4. Easy selection of color and change
5. Variable speed of cursor movement (step rate)
6. Graphics device input (mice or graphics tablets)
7. Ability to combine text with graphics images
8. Should make maximum use of the systems capabilities (resolution, colors, etc)
9. Area or shape color fill
10. Ability to display the drawing from your own program
11. Ability to print the drawings on a printer
12. Ability to take a drawn image and move or copy it to a new area on the screen
13. Ability to rotate or mirror image a drawing
14. Ability to save and retrieve drawings on a disk
15. Ability to look at a disk directory within the program
16. On-line Help
17. Miscellaneous features

How does Doodler stack up? Well, let's look at the program point-by-point and see how it compares to our check list.

Installation Program

Doodler comes with an Install program to assist you in selecting which printer you wish to use for printouts, and which colors you want to print when printing on a black and white printer. In other words, you specify which colors will be printed as black, and which ones will be ignored (white space).

The installation program is entirely menu driven and even allows you to custom configure the program for printers not included. You are asked for your special printer codes, etc, and when you have completed the driver installation, all the information is stored in a file on the disk. This permits the custom printer driver to be used by the program the next time you use it without requiring reconfiguring each time.

Common Shape Drawing

Drawing common geometric shapes such as squares, circles, and rectangles is provided by Doodler. Squares or rectangles, for example, require nothing more than moving the cursor to one corner of the box you wish to draw and hit F6. You then move the cursor to the diagonally opposite corner of the proposed box or rectangle and hit F6 again. Instantly, the box appears on the screen in the color you have currently selected.

Circles are drawn by specifying the center point of the desired circle with the cursor and hitting the circle function key, F7. You then move the cursor to any point you want on the circumference of the circle and hit F7 once again. The circle then appears in the currently selected color just as the box did.

Line Drawing

Two ways exist to draw lines. One is to simply draw the line by moving the cursor in the desired line direction like an etch-a-sketch. The other method is sometimes more convenient. Point-to-point drawing is done by positioning the cursor on one endpoint of the desired line and hit function key F5. You then move the cursor to the location of the other line endpoint and hit F5 again. The line will instantly appear in the color currently selected.

To change between drawing and simply moving the cursor, the 'MOVE' key F2 is used as a toggle. To draw, simply hit F2, and to 'lift the pen' or move the cursor without leaving a line, simply hit F2 again. The current state is displayed on the 25th line of the screen.

Color Selection And Change

The menu on the 25th line always displays the currently selected color. When a color change is desired, all that is necessary is to hit the F0 key. A color selection menu will appear, and you can choose from 8 possible colors by choosing the number of the color you want from the menu. The standard menu reappears, and the color you have selected appears as the background to the color section of the main menu.

Cursor Movement Speed

Cursor movement speed, or step rate, is adjustable in Doodler. When the program begins, it is set at a 'speed' of 5 which means that it will move 5 pixels for every cursor movement keystroke entered.

The legal range is from 1 to 9 and is a very handy feature for quickly moving about the screen. To change the step rate, the F3 key is used. Each time the key is pressed, the step rate is increased by one until the step rate reaches 9. Then the rate starts over again with 1. The lower step rates are useful for intricate drawings and lines, and the larger steps are quite useful for moving the cursor about the screen.

Graphics Device Input

This is the one area where Doodler falls short. There is no allowance made for graphics input. The only allowable method of drawing is through cursor movement. But to be fair to Doodler, it must be remembered that there are really no standards set on the Z-100 for graphics input and there are no particular devices which are being commonly used in the Heath/Zenith community. Therefore, it becomes nearly impossible to decide which devices to support, and how to implement them.

Combining Text With Graphics

In my opinion, this is one of the best features of Doodler. I have used most of the other graphics programs on the market for the H/Z-100, and none of them come close to doodler in this category. Doodler can not only combine text by simply typing in the desired text, but it comes with several very special and very dramatic character fonts. The sizes of the fonts can be multiplied to produce some startling effects. A standard font is supplied, along with a Roman font, a Shaded font, and a Script font called Murray Hill.

Doodler also provides for scaling of character sizes. When selecting the font to use, you are asked for the scale (size). The legal range is from one, which is the same as defined in the font table, to nine, which is nine times the normal size. The scale or font can be changed at any time. This allows you to have different sizes and different fonts all within the same word if you want.

Where Doodler shines is in displaying these characters of varying size. Not only is the size of the characters variable, but when they are displayed, they are proportionately spaced on the screen. This is very important when characters of different sizes are being displayed.

The color of the text can be all the same, or you can have every letter a different color. All fonts can also be used in an italic or backslant mode, in any of the allowable sizes.

In the remote chance you don't care for any of the supplied character fonts, Doodler comes supplied with a character font editor, so you can create your own. Through the use of the editor, you can create an entirely new font, change only a few characters in an existing font, and save all the modifications for future use in another program.

The operation of the editor is smooth and flawless and is an amazing product in itself. It is simple and fun to use, and is of the quality you would expect if you were buying just a font editor.

This font editor is only for fonts used in Doodler though, and should not be confused with the characters in the ALTCHAR.SYS file used by ZDOS.

Maximum Use Of System Capabilities

Doodler makes fairly good use of the Z-100's capabilities. Full resolution is possible, and all eight colors are available for you to use in your drawings. The only feature lacking, which is available on the competition's products, is the use of the interlace mode. However, interlace mode is of minor importance as it seems to require monitors with capabilities lacking in the most commonly used monitors.

Although interlace is available in the other graphics programs I have, I seldom find myself using that feature due to the way it looks on the monitor screens. Even with an expensive RGB color monitor, (a Z-135) the screen looks very unusual when in interlace mode.

Color Fill Of Shapes

Color fill of shapes or areas, (sometimes called painting) is well implemented in Doodler.

The first thing that must be done is to be sure that the border of whatever shape is to be filled is completely enclosed by one color border. You then move the cursor to the inside of the shape you intend to fill, hit F9 bringing up the PAINT menu, you will be asked which color to paint with and the color of the border of the object you wish to fill. After your response, the fill operation takes place automatically.

This is one operation which does require some care, however. If your border is not solid, color fill "leaks" as the program will not be able to determine where the border is and where the background is.

Use Of Drawings In User Programs

Doodler is written in compiled ZBASIC and graphics generated from Doodler can be used by your own programs. Examples are given in the manual describing how to call a drawing from a user program. The manual only describes calling procedures from within a BASIC program, but it should be very easy to use from other languages such as Assembly, 'C', or Pascal after some study of the BASIC examples.

The drawings are stored as ".PIC" files and are compatible with the BASIC 'BLOAD' command.

Printer Compatibility

Doodler uses what it calls 'Framing' to define the area you wish to operate on. This can be a very small area around a character or couple of pixels, for example. It can also be very large defining the entire screen if you wish.

Framing is the process of defining the area you wish the commands to operate on. It is just like drawing a rectangle around the area you wish to work with. This can be a very small area, or an area as large as the entire screen.

Before doing any of the operations which use 'Framing', you must 'Create' the frame. This is done by moving the cursor to a corner of the frame you wish to create and hit F10. Then select '1' for create, and move the cursor to the diagonally opposite corner of the frame and hit F10 once more. The frame is now defined, and all frame operations such as print, save, clear, etc., will work using this area.

Doodler is compatible with most of the popular dot matrix graphics printers commonly in use in the Heath/Zenith community.

Before using the print feature, you must install a driver for your particular printer. This is accomplished through the install procedure for doodler. If your printer is not supported, provision has been made for you to custom configure your own printer driver.

Printing a drawing in doodler is not as straight forward as some. You don't simply just issue the 'Print' command for whatever drawing happens to be showing on the screen and expect it to go to the printer.

Once you have defined the frame area of the picture you wish to print, you select the frame key F10. Select print, and you are then asked for line spacing (height of the picture — 1 is normal), width (wide, normal, or narrow), and left margin (margin in inches). If any of the parameters you give are out of range, the program will beep and await proper answers.

While this method does give quite a bit of flexibility in printing, I found it annoying after a few times to go through all that for every picture I wanted to print. However, I don't normally print that many pictures, so I feel this is a minor distraction.

Picture quality is good, and aspect ratios seem to be proper making for very nice reproductions of the drawings.

Moving And Copying Images

Moving and copying images on your drawing is done with the 'Frame' command just as in printing.

You must first create a frame defining the area you wish to copy or move. At that point, you move the cursor to the upper left corner of the location you want to move or copy the image to and select the frame move or copy from the frame menu.

You are presented with the ability to XOR, PSET, PRESET, AND and OR the data in the process. The result of this operation is just as it is described in the ZBASIC manual under the section describing "action verbs". Essentially it allows you to place one object behind another, in front of another, or completely cover another, etc. depending on your command.

Image Reflection And Rotation

Image rotation is accomplished through the use of a frame as in the printer command. Once the frame has been defined, it can be reflected top to bottom, bottom to top, left to right, and right to left.

Although there is no provision for rotation in the normal sense of the word, the frame can be reflected 90 degrees at a time as previously described. This allows for almost the same flexibility as true rotation for most graphics development needs.

File Archiving

Doodler has provisions for saving your drawings on a disk for later retrieval and display.

As previously described, file saving and retrieving uses the 'Frame' command. Once you have created (defined) the frame you want to save, the frame is written to disk. When retrieving, the frame is loaded and redisplayed on the computer screen when finished.

The defined frame can be just a portion of a screen, or an entire screen saved as a frame. Files can be from the default drive, or any of the other legal drive addresses. The only requirement for file storage or retrieval on a drive other than the default is to specify the drive designation in the program name. The file will then be stored or retrieved from the proper drive.

Disk Directories

By selecting the 'Files' option from the menu, you can then look at a directory of picture files, playback files, or all files. Drives other than the standard default drive can be looked at with this command by preceding with the drive designation.

On-Line Help

This is another weak area in Doodler. Although there is a very good menu on the 25th line, a further description is sometimes necessary to decide what you want to do. It is not convenient to continually page through a manual.

For example, it took me a while of experimenting to find that when I had begun a menu command and changed my mind, the only way to get out of the command is to hit the 'Help' key. It was not overly obvious in the manual. Methods of aborting commands in mid-command should be made very clear. In Doodler's case, neither Control-C nor 'Escape' would do the trick.

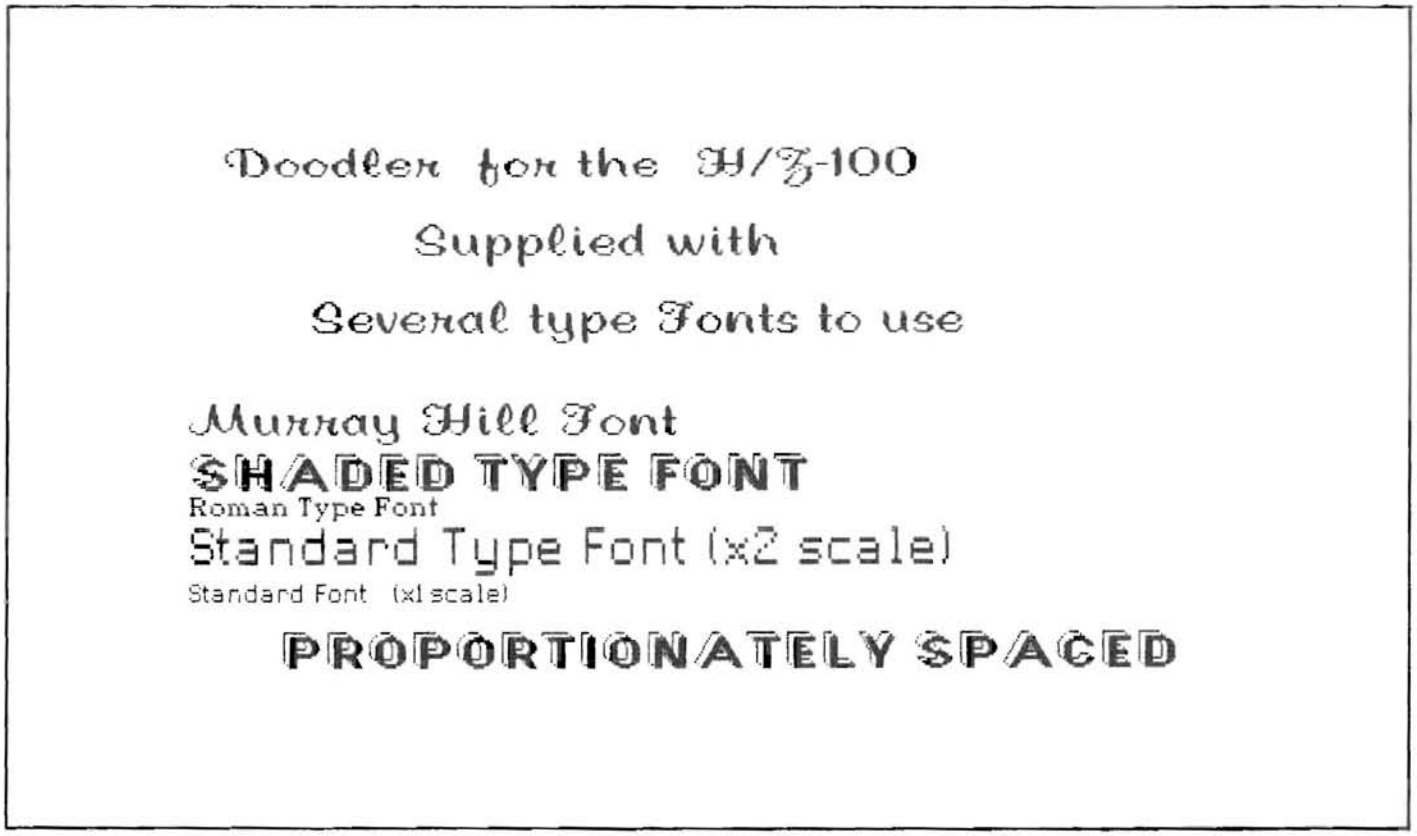
An on-line help with command descriptions would be a great help, especially for first time users.

Miscellaneous Features

Doodler has a few features of its own which are not really covered in my definition of minimum requirements for graphics software.

One of the most fascinating features of Doodler is the 'Playback' mode. Playback allows you to redraw your drawing up to any point you wish. While you are in Doodler, all your keystrokes are stored in a special string called playback. At any given point you can enter playback mode and completely redraw your image.

Playback comes in handy when you have been entering a complicated drawing and realize that you have misspelled a word, missed an operation you wanted to do, etc. Normally, this would mean you would be forced to redraw the entire picture. With playback, you can let Doodler redraw your picture and stop at any point prior to the error. You then correct the picture and



Doodler for the H/Z-100

Supplied with

Several type Fonts to use

Murray Hill Font

SHADED TYPE FONT

Roman Type Font

Standard Type Font (x2 scale)

Standard Font (x1 scale)

PROPORTIONATELY SPACED

continue.

Playback will operate at normal computer speed, or upon command will go into a single step mode where you can slowly step through the redrawing process. It is very handy and can even be used to create animated sequences by moving an object around the screen and using playback to redisplay the sequence of moves!

Finally, the character font editor must be mentioned as a plus. Although other editors are on the market, none I have seen are as flexible as the one in Doodler. I especially like the ability to create characters larger than the standard 8 × 9 size of the altchar.sys file. After all, we are doing graphics so why should we restrict our imagination.

Impressions

I must say I was rather surprised by Doodler. I was not all that excited about it when I first bought it. But after a few hours use, I really got to like it with the playback feature and the versatile character fonts.

Partially due to my snobbish attitude toward BASIC, I was very surprised with Doodler. However, Paul Herman has shown that BASIC is a serious contender for graphics applications.

On the negative side, I missed the graphics input device support. I would eventually like to see some sort of a joystick or graphics tablet support. It gets tiring pounding on the cursor movement keys for drawing pictures. Also, a tablet or joystick is a much more natural way of drawing.

I wasn't terribly excited about the way cursor movement was implemented. Doodler uses the cursor movement keys which allow you to move only in an up and down, left to right motion. There is no provision for diagonal movement.

I feel I must point out the excellent support Mr. Herman gives his product. Since I am also a programmer by profession, I try to give any software I buy a real workout (I try to crash it!).

I was able to find an obscure bug in the program (only one!) and it crashed the program requiring the system to be rebooted when it occurred. I mentioned it to Mr. Herman and in less than a week I had a new disk with the bug fixed.

Would I recommend Doodler? I think I would have to say that it is certainly a graphics program worth looking at. The text capabilities and proportional spacing of varying sized text is a very strong feature of Doodler. The Font editor is very nice and certainly adds to the package. However, it is the same or higher price as some other programs which offer better speed, more flexibility in image rotation, and better cursor movement. Also, there apparently is no text compression on the files saved, so a lot of disk space is used for pictures you save.

I would recommend that if you are looking for a graphics program, visit your local Heath store and try out the different programs you are interested in. They all have their strong points and weak points as well. Only you can tell which are the features most important to you.

So remember the next time that urge to doodle gets too strong for you, just grab your copy of Doodler and Doodle in full color!

DOODLER
Paul F Herman
P.O. Box 206
Safety Harbor, FL 33572

Available from Paul Herman and many Heathkit centers around the country. List price is $79.95.

More On The H/Z-100 Hard Disk Implementation

William M. Adney
P.O. Box 531655
Grand Prairie, TX 75053

From the mail that I've received, many of you were very interested in the discussion of the hard disk implementation on the H/Z-100 which appeared in the April issue. Because of this interest, I thought it would be appropriate to add some additional thoughts which did not make it into that column because of length.

Backups

One of the key items that did not get mentioned in the last column was the fact that you should NEVER use the same disk twice in a row to backup your hard disk. Always keep two sets of backup disks for your hard disk. If anything should happen during the current backup of the files, at least you will have your previous backup set to restore that partition. By rotating your backup disk set this way, you will always have the primary files in the partition on a backup.

One of the keys to establishing a good backup schedule is to determine the frequency of updates to data files on your hard disk. I'm assuming that you keep all of the programs in a single partition, which need not be backed up until you add some new software to that partition. Data files should be kept in separate partitions (or separate floppy disks) so that they are much easier to keep track of. The frequency of updates to these data files must be considered when you establish your schedule.

For example, my schedule is to backup all updated data files every week by simply using the CP/M PIP command or the MS-DOS COPY command. Those files are copied to a separate disk which can be considered as an "interim" or "incremental" backup. All data partitions are completely backed up with the BACKUP command every month which, in practice, amounts to the first weekend following the first of every month. This procedure is not my idea . . . it's been used for years at many large data centers. These data centers typically take a complete disk pack backup every week followed by the incremental backups on a daily basis. Like us, they frequently cannot afford the time and expense to backup all disk packs on a daily basis. These data centers may have several hundred disk packs, which makes it easy to see why they use this technique.

In addition to that schedule, I usually keep a disk in my 8" drive for the purpose of backing up a file as soon as I have completed a major update. While you can argue that this kind of process is too time consuming, I would argue that it is very efficient. It also allows me to experiment with various wording (i.e. editing) and still be able to quickly restore the original.

Image Copy Backups

One of the items that we really need for the H-100 is the streaming tape drive. Although Zenith has provided the best disk backup and restore software that's available, there is a significant advantage in being able to make a complete "image copy" of the hard disk. In this case, I'm using "image copy" to indicate a complete copy of the files on the hard disk without regard to partitions or operating systems. In addition, the software should have the capability to take the "incremental backups", that I discussed earlier, by checking the archive bit to see if the file has been updated since the last backup. That technique requires that all software that can update files be able to set the archive bit, so that the backup software can determine which files have been updated. The backup software also will have to reset the archive bit when the file has been backed up. I have heard that a streaming tape drive and software is available for the IBM PC. I hope that one of our vendors will be making that available to the H-100 owners.

Notes On The Z-150 And IBM PC

One of the more interesting things that's come to my attention lately is that the Z-150 is so IBM compatible that it's incredible. That's particularly remarkable since some of the IBM line is not upward compatible. I'm hearing more and more comments about the fact that the AT will not run some of the standard PC software. Reports are that there is a new ROM in the AT, as you would expect, and it apparently has been changed somewhat from the PC/XT version. Looks like compatibility is even a problem within IBM!

Compatibility involves many issues, but Heath and Zenith are doing a remarkable job of maintaining the IBM compatibility. I just became aware of a compatibility problem with MicroSoft Chart. It would not run on the Z-150. A simple phone call to ZDS revealed that they were not only aware of the problem, it had been fixed by a new ROM, version 2.

Some of you may be wondering why a version 3 of MS-DOS isn't available for the Heath/Zenith systems yet. Although I haven't confirmed this with the factory, I think that you should know that the current version 2.13 provides virtually everything that's available in PC-DOS version 3.

New PC-DOS commands for version 3 include ATTRIB, LABEL, SELECT, and SHARE. The ATTRIB command allows you to set a file as read-only or change it back to read/write similar to the CP/M STAT command. The MS-DOS FLAGS command, provided in the Programmer's Utility Pack, allows you to do that and more. You can use FLAGS to set the archive bit (mentioned above), mark a hidden (from the DIR command) file, set the file as read-only or mark a system file. And, of course, FLAGS will also reverse those attributes if desired. The only limitation is that FLAGS cannot change the attribute of the volume label (currently added in the FORMAT program), but that's relatively simple to do with DEBUG.

The LABEL command in PC-DOS version 3 does allow you to add, change or delete a label from a volume. The SELECT command provides the capability to select the keyboard layout with the date and time format you want to use. This capability is generally implemented in MS-DOS through the use of the COUNTRY parameter in the CONFIG.SYS file. The SHARE command allows you to install file sharing, which is used in a multi-user system and requires that appropriate software provide for the file sharing.

That's about all of the news on PC-DOS version 3. I don't see much there that isn't already available in the current MS-DOS, except for the file sharing.

The S-100 Bus

One of the things that I've recently noticed is that I don't have any more slots left in my H-100. I've got two disk controllers (floppy disk and hard disk), two 256K memory boards, and a Z-204 Multiport Board. Looks like I'm in deep trouble if I want to add another board! In short . . . HELP! I can't believe that there isn't an S-100 bus expansion unit available. Although I don't have any current plans to add any more boards to my H-100, I suspect that it's just a matter of time. For those hardware vendors who read my column, I will be glad to review an S-100 expansion unit that's compatible with the H-100. Although I don't expect to find any electronic compatibility problems, I'm interested in finding something that looks nice, too.

Keeping Up

One question that I get asked most frequently is how I manage to keep up with all of the new things in the Heath/Zenith user community. The answer is: "A lot of reading". In addition to REMark, there are three publications that I subscribe to: BUSS, H-Scoop, and Sextant.

BUSS is a newsletter which contains all kinds of information about Heath/Zenith hardware and software. It's published 20 times a year by the Sextant Publishing Company. Many new product announcements are included in BUSS, and I find it highly informative. Other information of general interest to Heath/Zenith users is also printed, as well as commentary on the current state of Heath Company and Zenith Data Systems.

H-Scoop is another newsletter which is available to Heath/Zenith users which has a slightly different approach. More technical items are printed, such as patches to WordStar. H-Scoop also carries new product announcements, as well as various reports on hardware and software. Henry Fale is the editor and publisher of H-Scoop, as well as being the President of Quikdata, Inc.

Sextant is a good magazine for the Heath/Zenith community. It carries items of interest to all users: technical information on hardware, reviews of new software, and other useful articles. It's also published by Sextant Publishing Company, and Charles Floto is the editor and publisher.

If you're looking for a way to keep up-to-date on all of the developments in the Heath/Zenith community, I can recommend all of the above. I have subscriptions to all of them, and each provides a lot of useful information.

I'd be remiss if I didn't mention that you should also have a subscription to REMark. Aside from the magazine, don't forget the membership in the Heath Users' Group (HUG) which is extremely valuable, since your HUG membership enables you to get discounts on computer hardware and software. I assume that you already have a subscription to REMark, since you are reading this issue.

Change Of Address

Now that things have settled down after my move to Texas, I also have a new mailing address as shown at the beginning of this column. If you have any questions about something in my column, just send a note or letter. It's very helpful if you leave some space for an answer, however, that's not a requirement. If you would like a personal answer, be sure to enclose a stamped, self-addressed envelope. Please note that you must enclose the envelope — my postage expenses were over $200 for last year, and with the latest increase in rates, I have to keep those expenses down. Since I answer all mail in the same week that I receive it, people have told me that it takes a total of about two weeks which allows for the time in the mail.

An Oversight

I mentioned Studio Computers in my last article and suggested that you might want to ask for their catalog. Unfortunately, I forgot to add their address at the end of the column. My apologies to Ray Massa. Please note that Studio Computers' address is properly listed at the end of this article.

Next Month

I still have a lot of items to review, primarily software, and we'll continue with that next month. For those of you who have not seen WatchWord yet, I'll review that. It's an absolutely dynamite word processor that seems to be able to provide all things to all people. There is also a lot of other great software, particularly in the communications area, that is crying for comment. Access and Pro-Driver will also be reviewed in this column, and I'm impressed with both.

Products Reviewed

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

BUSS (newsletter, 20 issues) $28.00/yr
Sextant (magazine, 6 issues) $14.97/yr
Sextant Publishing Company
716 E Street, S.E.
Washington, DC 20003

H-Scoop (newsletter, 12 issues) $20.00/yr
2618 Penn Circle
Sheboygan, WI 53081

MS-DOS Version 2 (OS-61-8) $150.00
MS-DOS Programmer's Util. Pack (CB-5063-16) $149.00
Heathkit Stores
Heath Company Parts Department
Hilltop Road
St. Joseph, MI 49085
(616) 982-3571

WatchWord $100.00
S & K Technology
4610 Spotted Oak Woods
San Antonio, TX 78249
(512) 492-3384

WatchWord Macro Utilities $20.00
EASYware
Small Computer Systems Software
7202 Faro's Court
San Antonio, TX 78233
(512) 657-7109

My Favorite Subroutines

Attention: My Favorite Subroutines

Here's a simple Screen Dump Subroutine for text output to the printer. Save the program in ASCII mode, then the routine can either be merged and run with MERGE" DUMPTEXT and RUN 50000 or it can be incorporated into your programs and called up with a GOSUB 50000.

```
50000 '<< DumpText >> by G.Holt 1984
50010 A$-INKEY$:IF A$-""THEN 50010
      ELSE IF NOT(A$="p"OR A$="P")THEN RETURN
50020 WIDTH LPRINT 80:FOR ROW=1 TO 24:FOR COL=1 TO 80
50030 CH=SCREEN(ROW,COL):LPRINT CHR$(CH);:NEXT:NEXT:
      RETURN
```

George Holt
403 2nd Street West
BAFB, LA 71110

Attention: My Favorite Subroutines

Reference the "Force Upper Case" subroutines by John Pierrel and Ed Lovett in the October '84 issue. The following incorporates the best of both John's and Ed's routines.

```
2 LINE INPUT"Word(s). ";W$:GOSUB 4:PRINT W$:GOTO 2
4 FOR I=1 TO LEN(W$):L=ASC(MID$(W$,I,1))
6 IF L>96 AND L<123 THEN MID$(W$,I,1)=CHR$(L-32)
8 NEXT:RETURN
```

George Holt
403 2nd Street West
BAFB, LA 71110

Make Neat Flowcharts Fast With This Symbol Library

Clement S. Pepper
12938 Orangeburg Ave.
San Diego, CA 92129

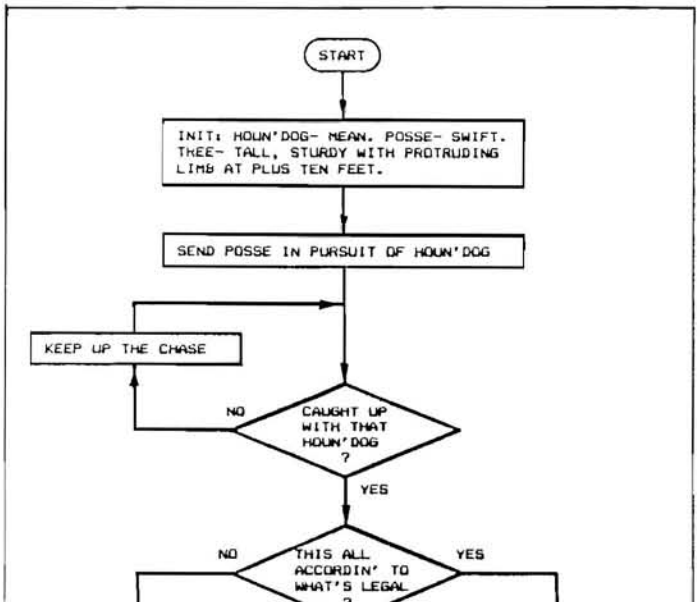

START
INIT: HOUN'DOG- MEAN. POSSE- SWIFT. THEE- TALL, STURDY WITH PROTRUDING LIMB AT PLUS TEN FEET.
SEND POSSE IN PURSUIT OF HOUN'DOG
KEEP UP THE CHASE
NO
CAUGHT UP WITH THAT HOUN'DOG ?
YES
NO
THIS ALL ACCORDIN' TO WHAT'S LEGAL ?
YES
LET THE PORE DOG GO
DO WHAT IS TO BE DID
END

Figure 1. The finished flowchart.

I find flowchart construction to be a dynamic exercise. I typically begin with sketching the major elements on the backs of discarded printouts. Very soon there is the ever present demand for additions. My flowchart becomes a flotilla of smudgy shapes on a sea of nearly unreadable scribblings.

That is how it used to be. Not any more though, because now I simply move to the keyboard of my H-89.

For the past couple years, I have used PIE and WORDSTAR for flowchart preparation. The result, as you see in Figure 1, is quite professional in its appearance. While appearance has its place, time is often of greater concern. Both were enhanced when I assembled the symbol library.

Although I have been doing flowcharts on the keyboard for some time, the library is new. While constructing a lengthy flowchart one evening by repeated "cutting and pasting", I asked myself why I was working in such a cumbersome fashion. Right then, I dropped everything and put together the library.

Figure 1 is not the printer output. Figure 2 is. The finished product of Figure 1 evolves by drawing over the dots and dashes. For this I used a black razor point pen, a straight edge, and a circle template.

Figure 3 is the Library. The first step is to get it on your disk exactly as shown. What is done next depends on the editor you are using. That is, there are editors which provide for disk read/write operations, and editors which do not.

PIE is one that does not. To use the library with this or a similar editor, we must first create a flowchart file. To it, we then append the library file.

An example will be helpful. Suppose we wish to write a flowchart for a program we have named HANGDOG. Our first step, then, is to create a file for the flowchart which we do with the command line:

PIE HANGDOG.FCT <cr>

After entering formatting and heading information, we exit from HANGDOG.FCT. Now we append the library using:

PIP HANGDOG.FCT=HANGDOG.FCT,FCHRTSYM.LIB <cr>

The first library entry is START. As a first step, delete the line containing START.SYM. Next, drop the cursor to the line beginning with END.SYM and insert a half dozen new lines. This to create a gap for working space.

Suppose the initialization for HANGDOG requires a 3ROWBLK-.SYM. Using our editor's forward search procedure, we advance the cursor to the row containing 3ROWBLK.SYM. With PIE, we would next ENTER 9 (for 9 lines to be remembered) and press the H-89's RED function key. Using the backward search function, we return the cursor to the dangling '┤ ", press the WHITE function key and there 'tis: the 3 row block in place and waiting for text to be entered. This procedure is repeated, as often as needed, with any of the symbols until the chart is completed.

A natural step at this point would be to attach the END and delete the remainder of the library. A major advantage of the library has yet to be realized, however, which is entering in the inevitable changes. When printing with TEXT a Xw separating the two files

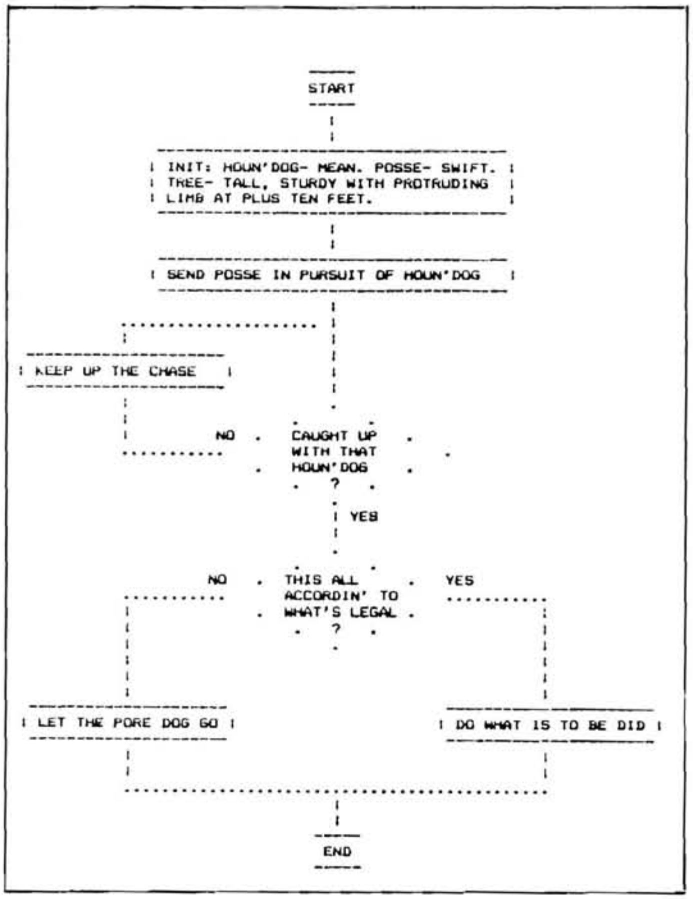
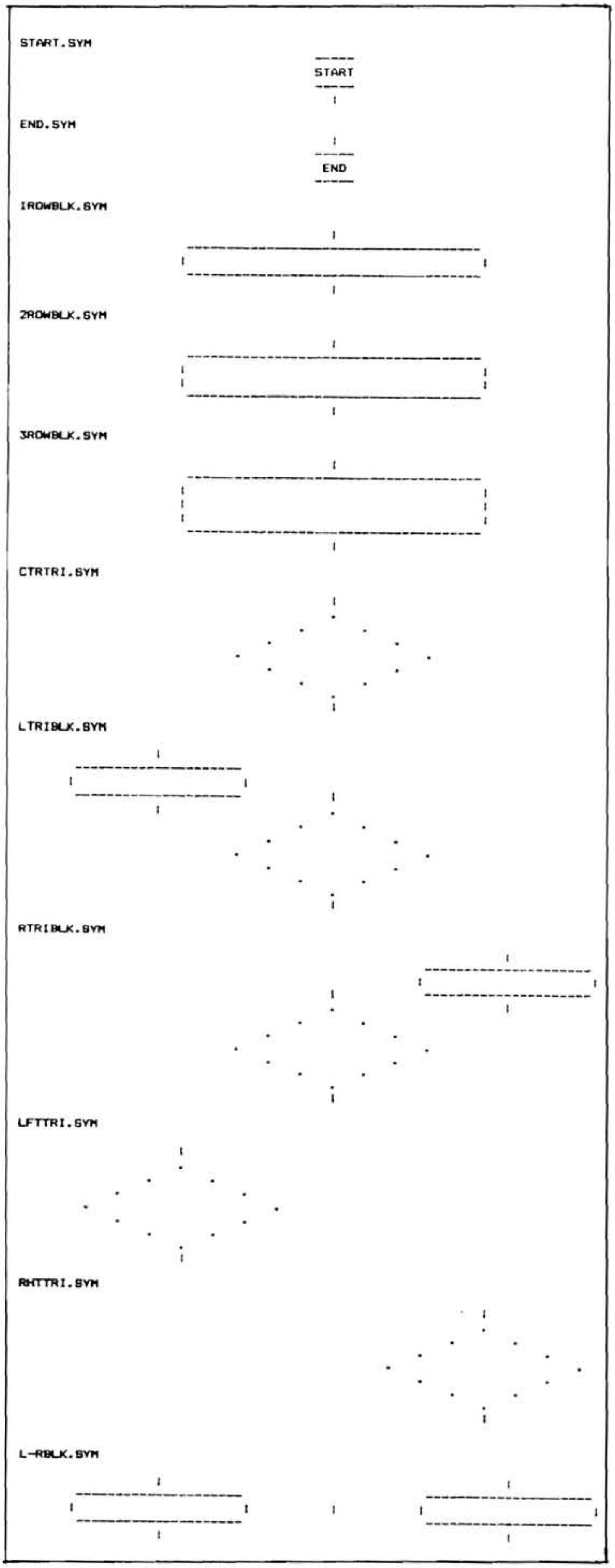

START
INIT: HOUN'DOG- MEAN. POSSE- SWIFT. TREE- TALL, STURDY WITH PROTRUDING LIMB AT PLUS TEN FEET.
SEND POSSE IN PURSUIT OF HOUN'DOG
KEEP UP THE CHASE
NO
CAUGHT UP WITH THAT HOUN'DOG ?
YES
NO
THIS ALL ACCORDIN' TO WHAT'S LEGAL ?
YES
LET THE PORE DOG GO
DO WHAT IS TO BE DID
END

Figure 2. The flowchart as it comes from the printer.

Figure 3. The flowchart library. Copy exactly as shown with your editor.



will stop the printer after the first. So the library can remain appended for as long as there is a need for it.

Now, suppose we wish to create HANGDOG.FCT with WORDSTAR or a similar word processor that permits disk read/write operations from the editor. As a first step, break up FCHRTSYM-.LIB into its component symbols, creating an individual file for each. Write each to disk under its SYM title. You now have each symbol under its own filename, which can be read into your flowchart file whenever it is needed.

Again, we can employ HANGDOG.FCT as an illustrative example. After making formatting and heading entries, request a READ of START.SYM. Delete the title line and move the cursor to the line below the dangling '| ." Request a READ of 3ROWBLK-.SYM. And just like that you're on your way.

A Review Of Condor Database Management System

Stephanie Butzbach
923 Wayne Street
Saint Joseph, MI 49085

If you are wondering how to organize and use a collection of information about one or more subjects, a data base may be just what you need.

Data storage and interfacing data with software routines is an important part of the computing process. Before the concept of the database, information was stored in a file or a program and if any changes were made to the data, a new program would have to be generated to access the data. With a data base, a collection of information may have more than one use, without writing new routines each time new information is required. A data base allows data records to have multiple "views".

One of the data base packages offered on the market is the Condor Database Management System, developed by Condor Computer Corporation of Ann Arbor, Michigan.

It requires a Z80 microcomputer or an 8080-family microprocessor with at least 48K bytes of RAM (random-access read/write memory) under the CP/M-80, ZDOS, and MP/M-80 operating systems. The Condor also runs on the 8086 and 8088 microprocessors under the MS-DOS and CP/M-86 operating systems. A 24-line by 80-column display terminal with screen-erase, line-wrap, and cursor-addressing capabilities is needed. Two floppy disk drives with a total capacity of at least 300K bytes of memory are needed. Hard-disk drives can be used if they are supported by the operating system. A printer is very useful, preferably one with 132 column print capability form feed, and form length control.

Condor frees the programmer from having to code, test and debug many data manipulating routines. Some of these generalized routines are: sorting, query retrievals, report writing, data insertion, data deletion, calculation of averages, totals, counting, and searching for records.

When large amounts of data are kept, it is important to minimize data redundancy. If multiple copies of data are kept, space is wasted, time is wasted when information needs to be changed, and data integrity is lost. Condor helps the user minimize data redundancy by allowing the user to form different data files from the existing data base files. The user may combine two datasets, attach selected fields of two datasets, update fields of one dataset with those from another, and form a new dataset with selected fields of a dataset. All of these features form what is called a RESULT dataset and can be used as a dataset or a data file.

Condor provides communication with the outside environment with two commands: read, which transforms an ASCII file to a Condor dataset; and write, which transforms a Condor dataset to an ASCII file. These features allow for data independence, which is the ability to use the data base without knowing the internal data representation details. Data appears to be organized differently for each application program. The conversion of data files allows Condor to interface with any of the popular computer languages.

One of the best features of Condor is the ability to alter or reorganize your datasets by adding, deleting, or changing the physical definition of the fields. This is especially helpful when you have accumulated hundreds of records and realize you need to store another piece of information, or no longer need to keep certain information. I am sure it would really make you happy to know you do not need to retype hundreds of records over again!

With Condor, arithmetic is made simple. The operators are: addition, subtraction, multiplication, and division. The operands may be either integers or decimal numbers, text (literal constants) or field values (variables). The compute commmand has two forms. The single compute, which may contain a search condition, calculates the formula and places the result in a specified field. The multiple compute command, which also may contain a conditional search, allows for eight different computations. Each computation may contain 32 operators, 32 operands, and 16 sets of parentheses. Using the stax and tabulate commands allows you to compute and produce a report for selected fields without changing any fields.

Getting Started

The first step in creating a data base is to study the situation you wish to represent in the computer. Then decide what information you wish to store, create field (data) names and definitions of the physical nature of the data, such as: size, alphanumeric, numeric, a date, or dollar and cents representation. Condor allows the user to define these fields and create a formatted screen to allow easy data entry. Even the novice data entry person will be able to easily use the formatted screen. You can easily go back to previous fields to correct your mistakes. Lines are used as a guide to indicate how many characters are expected in each field. When formatting the data fields, the user may request boundaries and expectations from the data that will be entered in the data field. This feature can be very useful in error checking during data entry time. When the same information is being entered into a field consistently, the auto-fill and auto-repeat features will save a great deal of time. Auto-fill allows the user to set a default value for a selected field, and skips the field during data entry. The data operator may override this default by going back to the previous field and retyping the new value. The auto-repeat feature inserts and displays information from the previous record. This option can also be overridden.

After the data has been entered, Condor provides many routines to manipulate the data base files. Using the Condor commands, the user can create routines to handle many situations, such as: accounts payable and receivable, inventories, and scheduling applications. Command files, which are preprogrammed routines, can be written to accomplish the procedures automatically. Command files are easily created using If and nested If structures, and Condor commands. The logical operators, (equal, not equal, greater than, less than, greater or equal, less or equal) are provided in the conditional directives.

Command files are very important for efficient operations. Commands that are routinely issued can be placed in a command file to save operation time. Long commands can be issued without typographical errors or omitted field names. Using a set of automated procedures allows persons unfamiliar with the data base to perform the work. The procedures implemented will be consistent from day to day, operator to operator, if the commands are placed in a command file. In conjunction with the command file, Condor allows the user to create help menus to aid the operator to step through a procedure correctly and easily.

Condor contains an excellent and very powerful report writer. Reports are user formatted using a few simple screen editor commands. The reports are generated exactly as you plan it on the screen. To take full advantage of report writer features, you must be willing to wade through a few pages of complex material, but it is definitely worth the effort. Calculations can be performed inside the report, and break points can be established using selected fields. The data base files can be represented in many ways using the report writer, such as: mailing lists, invoices, statements, and specialized reports.

Documentation for the Condor is one of the finest I have seen. It contains two manuals. One manual is a reference guide, which allows the experienced Condor user to look up information in a dictionary format using a well constructed index. A summary of commands and a glossary of terms is also provided. The second manual is a text that instructs the user in the concepts of a data base, how to implement and maintain it. The manual is divided into five sections. The first section is a tutorial explaining start up procedures, data entry and data manipulation commands.

Section two explains the importance and implementation of back-up copies and entering data from a temporary data base to a master data base to avoid data entry errors. Section two also shows the creation of command files and help screens. Also in this section is instruction on Condor's query capabilities. Condor allows queries of the data base through a set of easy-to-learn English-like commands.

Section three explains how to provide maintenance to the data base. It explains the importance of back-up copies, destroying unused data sets and data fields, and recovering space from deleted records. The manual provides a guide to the user to manage the allotted space on the disks. Section three also explains error recovery. Error recovery is both good and bad in Condor. Messages are sent to the user if an error is encountered, and these errors are thoroughly covered in the manual. That is the good news. The bad news is that in some cases the only recourse in recovering from an error is to go to a back-up copy and start over again.

Section four contains instructions for the report writer, which is quite complicated. However, sample reports are shown to clarify the instructions. And Section five contains a very good index.

Condor includes a data dictionary which allows the user to define the variables that are used in the datasets. Condor's data dictionary is limited in some areas. It does not provide for security of the data base, and allows anyone who has access to the data base to see all of the data base. The data dictionary does not allow for storage of the relationships between the variables. It does not allow the data base to audit itself, which would be helpful for monitoring the performance and tuning the data base.

Some of the good features of Condor are: sorting on keys, creating an index on a field pointing into the data base for fast retrieval, and use of multiple file access.

Condor is one of the best data base management packages on the market. It is fast and will handle most any application. Condor is particularly effective in accounting functions and personnel record processing. Condor uses the very efficient B-tree-index structure, and sorts the data files externally. Condor is truly a variable data base, which means it only stores the fields with data within them to save space. Condor has the ability to handle a relational data base structure. A relational data base is linking two unrelated datasets with a third dataset. The relational structure eliminates the need for a huge dataset containing all the information, which is more efficient and saves disk space.

Condor Data Base File Specifications

Records per data base file:	32,767 Maximum
Bytes per record:	1024 Maximum
Fields per record:	127 Maximum
Characters per field:	127 Maximum
Largest number:	+/−2,148,373,647
Smallest number:	+/−0.01
Numeric accuracy:	9 digits
Index key length:	127 bytes maximum
Terms in compute command:	32 Maximum
Sort keys:	32 Maximum
Sort file size:	128K Maximum
Conditions in select command:	32 Maximum

Vectored from Page 12

template for the cabinet of your device. This should be taped on the cabinet while you drill and cut holes, then removed. Next, use the plotter to draw and label a front panel. Once finalized this panel can be drawn on a sheet of flexible opaque plastic using a pen designed to plot on transparencies.

The inexpensive plastic is available from most office-supply stores in large sheets. Once the panel is drawn to satisfaction, the plastic can be carefully cut with a guide and a razor blade, and glued to the cabinet.

Thank you,

Walter M. Scott, III
7608 Luscombe Drive
Knoxville, TN 37919

MBASIC Program To Control SCRNCLK

Dear HUG,

The following is a short MBASIC program to control the 'SCRNCLK' program (HUG part #885-1237). It is quite useful in resetting the clock after excessive disk access. It also has the options of enabling or disabling the clock display. It can be saved with the 'A' option, and merged with any program requiring repeated disk access.

```
1ØØ CLKADR=PEEK(1Ø)*256+PEEK(9)
11Ø PRINT CHR$(7)
12Ø PRINT CHR$(27)"E"
13Ø PRINT "OPTIONS:";TAB(25);"S=SET"
14Ø PRINT TAB (25);"E=ENABLE"
15Ø PRINT TAB (25);"D=DISABLE"
16Ø PRINT TAB (25);"X=EXIT"
17Ø PRINT:PRINT
18Ø PRINT "OPTION: ";
19Ø O$=INPUT$(1)
2ØØ PRINT
21Ø IF O$="D" OR O$="d" THEN POKE CLKADR-1,Ø
22Ø IF O$="E" OR O$="e" THEN POKE CLKADR-1,1
23Ø IF O$="S" OR O$="s" THEN 25Ø
24Ø IF O$="X" OR O$="x" THEN 36Ø ELSE 35Ø
25Ø PRINT
26Ø INPUT "CURRENT TIME: ",TIME$
27Ø IF LEN(TIME$)=7 THEN TIME$="Ø"+TIME$
28Ø IF LEN(TIME$)<>8 OR LEFT$(TIME$,2)=>"24"
      THEN PRINT CHR$(7):FOR A=1 TO 5Ø:NEXT A:
      PRINT CHR$(7):PRINT "INVALID TIME (ENTER H:MM:SS)":
      GOTO 25Ø
29Ø POKE CLKADR-14,ASC(MID$(TIME$,1,1))
3ØØ POKE CLKADR-13,ASC(MID$(TIME$,2,1))
31Ø POKE CLKADR-11,ASC(MID$(TIME$,4,1))
32Ø POKE CLKADR-1Ø,ASC(MID$(TIME$,5,1))
33Ø POKE CLKADR-8,ASC(MID$(TIME$,7,1))
34Ø POKE CLKADR-7,ASC(MID$(TIME$,8,1))
35Ø GOTO 1ØØ
36Ø END
```

Michael W. Reed
75 West 250 North #46
Clearfield, UT 84015

RECOVER.BAS

Dear Editor,

I am enclosing a program for use in REMark. I recently had the bad experience of not being able to copy or print out, using TEXT.COM, a text file after I had saved it once. I kept on getting BAD SECTOR errors. I could not use PIP, ED, or any technique that I knew of to recover the 12K file. Finally, I hit upon a solution. I knew that I could read it with MBASIC into a memory array and then I could print it out in a new file. Here is my solution. It worked for me. I share this in the hope it may help other people in similar circumstances. After I had recovered the file, I tried reformatting the disk and found that it was a corrupt disk. I call my short program RECOVER.BAS.

```
5  REM RECOVER.BAS
1Ø INPUT
   "HOW MANY LINES SHOULD I RECOVER FROM YOUR FILE ? ", L
2Ø LINE INPUT "NAME OF FILE TO RECOVER..",N$
3Ø DIM A$(L)
4Ø OPEN "I",#1,N$
5Ø FOR I=1 TO L
6Ø INPUT #1,A$(I)
7Ø PRINT A$(I)
8Ø NEXT I
9Ø OPEN "O",#2,"NEWFILE.TXT"
1ØØ FOR I=1 TO L
11Ø PRINT #2,A$(I)
12Ø NEXT I
13Ø PRINT "FILE RECOVERY COMPLETE..."
14Ø END
```

Sincerely,

Don Talkington
1018 Freda Lane
Redding, CA 96003

Modified HRUN

Dear Pat,

After talking with you today, I decided to go ahead and modify HRUN to account for the currently logged CP/M disk. I made these changes and they seem to work fine. I thought I would send you a copy of the changes that were made. If you think there is enough interest, you might want to print them in REMark. I tried to comment, as clearly as possible, as to what's going on so maybe some readers would gain a little extra insight into assembler programming.

This change causes any reference to SY0: either by explicit name or the lack of an HDOS drive number to be the currently logged CP/M disk. If you are logged to drive A: all drive assignments stay as default. (i.e.: SY0:=A: SY1:=B: etc.) However, if you are logged to any other CP/M drive, the currently logged disk becomes SY0: and any other HDOS disk number begins with SY1: as A: on up. (e.g. logged to drive C: SY0:=C:, SY1:=A:, SY2:=B:, etc.)

Thanks for your time and I hope these changes are helpful.

Sincerely,

John J. Real, Jr.
P.O. Box 1123, Suite 218
1407 24th Avenue South
Grand Forks, ND 58201

```
IN THE DECODE FILE NAME SECTION
MAKE THE FOLLOWING CHANGES OR ADDITIONS

CHANGE AS DESCRIBED

DECODE5 DB      SCALL.xVERS       ;GET VERSION NUMBER
        STA     DCVER             ;PUT IT IN
        LDA     DCDEV+2           ;GET UNIT
;*********    NEXT 3 LINES ARE THE OLD CODE REMOVE AND
```

```
                REPLACE WITH THE
; *********      FOLLOWING LINE   CALL   DECODE9 *********
;       ORA     A
;       JZ      DECODE6
;       SUI     '0'
;*********************************
;            REPLACEMENT
        CALL     DECODE9
;               END REPLACEMENT
DECODE6 STA      DCDEV+2           ;DEVICE MUST BE BINARY
        LXI      D,DCDEV           ;POINT TO DECODED DEVICE
        LXI      H,DEVTBL          ;AND DEVICE TABLE
        LXI      B,102H            ;B = OPEN FLAG, C = # CHARS
**************************************************

ADD THE FOLLOWING CODE AS SHOWN
******************************


DECODEE POP      B                 ;RESTORE BC
        MVI      A,ECxIFN          ;ILLEGAL FILE NAME
        STC
        RET
; MY DECODE FOR DEFAULT CPM DISK
;**************** NEW CODE INSERTED HERE **************
;*****************************************************
DECODE9 ORA      A                 ;ANY UNIT SPECIFIED
        JZ       D9                ;NO GET CPM DEFAULT
        LDA      DCDEV+2           ;IF SPECIFIED UNIT GET
                                    IT BACK
        CPI      '0'               ;IS IT SPECIFIED AS SY0:
        JZ       D9                ;YES THEN GET CPM DEFAULT
        STA      CPMDEF            ;SAVE DRIVE NO IN BINARY
        CALL     D9                ;GET CPM DRIVE NUMBER
        MOV      C,A               ;MOVE IT TO C
        LDA      CPMDEF            ;GET CPM NO. BACK
        CMP      C                 ;SEE IF WE ARE CALLING
                                    THE DEFAULT
        JZ       D9                ;YES GET CPM DRIVE NO
        LDA      CPMDEF            ;DO IT AGAIN WITH ASCII
                                    DRIVE SPEC
        SUI      '0'
        STA      CPMDEF
        CALL     D9
        MOV      C,A
        LDA      CPMDEF
        CMP      C
        JZ       D9
        CALL     D9                ;FIND OUT WHICH CPM DRIVE
                                    WE ARE LOGGED TO
        CPI      0                 ;IS IT DRIVE A ?
        JNZ      D8                ;IF NOT SUBTRACT ONE FROM
                                    SPECIFIED DRIVE
        LDA      DCDEV+2           ;GET BACK DRIVE REQUEST
        SUI      '0'               ;REMOVE ASCII BIAS
        JMP      D10               ;RETURN
D8      LDA      DCDEV+2           ;GET BACK DRIVE REQUEST
        SUI      '1'               ;SUBTRACT ASCII + 1 FOR
                                    REMAP OF DRIVE
        JMP      D10               ;RETURN
D9      MVI      C,19H             ;BDOS CALL FOR LOGGED DISK
        CALL     BDOS              ;DO IT
D10     RET                        ;RETURN
CPMDEF  DS       1
;*****************************************************
;   ***************   END NEW CODE   *******************
;
DCFLG   DS       1
DCDEV   DS       3
DCNAME  DS       8
DCEXT   DS       3
        DB       0                 ;PROJECT
DCVER   DB       20H               ;VERSION
DCDVT   DW       0                 ;DEVICE TABLE ADDRESS
OPFLG   DB       0                 ;OPEN FLAG (USED LATER)
```

Help With A FORTRAN FORMAT

Dear Editor:

In the August REMark, on page 65, William Crocker asked for help with a FORTRAN FORMAT problem I'd struggled over, without success, several years ago. However, both FORTRAN and my knowledge have improved since then and maybe I can help. The critical lines in Mr. Crocker's program were:

```
        WRITE (1,50) ANGLE, TIME
        WRITE (7,50) ANGLE, TIME
50      FORMAT ('X',2F10.5)
```

To write the carriage control character 'X' to the CRT and a line feed to the disk, change these lines to read:

```
        WRITE (1,50) 'X', ANGLE, TIME
        WRITE (7,50) CHAR(8), ANGLE, TIME
50      FORMAT (A1, 2F10.5)
```

The slash is the standard newline carriage control character; so a simpler change may work even better. (I no longer have access to FORTRAN, so I cannot test these.)

```
        WRITE (1,50) ANGLE, TIME
        WRITE (7,50) ANGLE, TIME
50      FORMAT (/2F10.5)
```

Actually, the problem I faced was using the same FORMAT to READ and WRITE. But apparently, READ is READ, and WRITE is WRITE, and never the twain shall meet.

In the "My Favorite Subroutine" section on page 10 of the same issue, Carl Edwin Lovett, Jr. writes of a BASIC routine to calculate Julian dates from Gregorian, and conversely. These routines get into some rather large floating-point numbers, and cannot be used with languages limited to two-byte integers, such as HUG's Tiny Pascal or Software Toolworks' C80 (ver. 1.5). Last Spring, I developed two integer versions for these languages. The first, in Tiny Pascal using only positive integers, covers 32765 days, some 89 years. The other, in C80, is a little more comlex, but extends the range through the negative integers covering about 177 years. Both really calculate pseudo-Julian dates, being based on arbitrary zero dates. The only restriction on the base year is that it must be a leap year; and if it's earlier than 1888, a correction must be made for the leap year 1800. Otherwise, they work as-is, since correction is made for 1900, and 2000 is a leap year. The listings follow.

Sincerely,

Lansing E. Tryon
29 Fairhaven Road
Rochester, NY 14610

```
{File JULIAN.PAS ver. 26-Feb-84}
{  Converts to consistent Julian Date from 3/1/BASYR to
     11/15/(BASYR+89) }

CONST   BASYR = 1920;

VAR     YEAR,MONTH,DAY  : INTEGER;

{----------------------}
FUNC CTOJ(YEAR,MONTH,DAY);

VAR     Y,M    : INTEGER;

BEGIN
    Y := YEAR - BASYR;
    IF MONTH > 2 THEN M := MONTH -3
    ELSE BEGIN
        M := MONTH +9; Y := Y -1
    END;
    CTOJ := 365*Y +Y DIV 4 +3Ø*M +(3*M+2) DIV 5 +DAY -1
END;

{-------------------------}
PROC    JTOC(JDATE);

VAR     D,M,Y             : INTEGER;
{       DAY, MONTH, YEAR : GLOBAL INTEGER: }

BEGIN
    Y := (JDATE - (JDATE +1) DIV 1461) DIV 365;
    D := JDATE +1 - Y*365 - JDATE DIV 1461;

    YEAR := Y + BASYR;
    M := (5*D -3) DIV 153; D := (5*D -3) MOD 153;

    DAY := D DIV 5 +1;
    IF M < 1Ø THEN MONTH := M +3
        ELSE BEGIN
            MONTH := M -9; YEAR := YEAR +1
        END
END;

{----------------------}

/*-----------------------*/
/*
 *  File JULIAN.C -- Lanse Tryon -- ver. Ø4-Mar-84
 *  Converts to and from pseudo-Julian Date from 3/1/1891 to
      11/15/2Ø69
 */

#define CURR_YR 1984      /* Default if year not specified */
#define CURR_CENT 19ØØ    /* Permits 2-digit year entries  */
#define BASE_YR 198Ø      /* Sets range to 11/15/2Ø69      */

#include "std.h"

int      year, month, day;
int      baseyear;

/*-------------------------*/

ctoj (year, month, day)
{
/* global baseyear */

int     y, m;

    y = year - BASE_YR;

    if (month > 2 ) m = month -3;
    else
      { m = month +9; y = y -1; }
    m = (365*y + m*3Ø + (m*3+2)/5 + day -1);

    if  (y >= Ø OR year EQ CURR_CENT)
        return m + y/4;
    else
        return m + (y-3)/4;

}       /* ctoj() */
/*---------------------*/

jtoc (jdate)
int     jdate;
{
/* global month, day, year */

int     d,m,y, baseyear;

   if (jdate >= Ø) baseyear = BASE_YR;
     else     /* offset base to give positive jdate */
     { baseyear = BASE_YR -88;
       jdate = jdate + 32142;  /* Days in 88 years */
     }

    y = (jdate - (jdate +1)/1461) /365;
    d = jdate +1 - y*365 - jdate/1461;

    year = y + baseyear;
    m = (5*d -3)/153; d = (5*d -3) %153;

    day = d/5 +1;
    if (m < 1Ø) month = m +3;
        else
          { month = m -9; year =+ 1; }
}       /* jtoc() */
/*---------------------*/
```

Need A Modem?

Dear HUG,

I would like to thank all the HUG members who have sent in letters with the problems they have encountered while trying to interface different equipment to the Heath/Zenith computers. This information has been very helpful. I would now like to give some help.

With the various modems available on the market, and the high prices on 1200 baud modems, the Anchor Signalman Mark XII is the lowest priced auto answer/dial modem that I have found. List price $399.00 compared to Hayes $695.00. The Mark XII is a 300/1200 baud modem, with Auto answer and dial features and Hayes compatible. The following is the interface to port J2 on the H/Z-100.

It comes with a male db25 cable, so a gender change cable must be made with 2 female connectors. The pin connections are as follows.

Mark XII side		H/Z1ØØ Port J2
1		1
2		2
3		3
4		4
5		5
6		8
7		7
8	NC	6

the remaining pins straight thru

With the $300 difference in cost, the Mark XII is well worth considering if you want an auto 300/1200 smart modem. I hope this will be of some help to anybody in the market for a modem.

Sincerely,

Donald Sutherland
Deckaa Electronics
111 W. First Street
Fosston, MN 56542

Orphaned, For Using Cassettes

Dear HUG:

Hopefully, this letter will be read. It's written on paper, not delivered on an approved disk. My only magnetic medium is no longer acceptable to HUG in spite of the fact that my computer system is all Heathkit. Of course, it's two and a half years old. Long enough, evidently, to disappear into prehistory. But, the antique is still working. The system is an H-8/H-19/H-14 combination with two cassette decks. All that's missing is participation in user group support.

Certainly, it's reasonable to devote most of REMark to the newest

Heath/Zenith products. That serves the majority interest. But how can you justify the special notice, "HUG will no longer accept program submittals for the software library for the H11 or cassette systems." (REMark, v5, i4, April 1984, p11) and then say, "We can't print it if nobody writes it," in response to Hubert E. Roy's letter? (REMark, v5, i9, September 1984, p5). Failure to buy a new computer every few months must be a crime punishable by symbolic summary execution. The only excuse for paying dues to a user group is the expectation that minority interests will be served. Otherwise, Personal Computing would be as useful as REMark.

My H-8 is a latter day version with gold flashed pins on the mother board and 64K dynamic RAM on a single card. I'm not about to retire it just because I wasn't quick enough to buy disk drives before they ceased to exist — along with source listings for HDOS, subject of another special notice. When the July, 1983 issue of REMark announced an "Odds and Ends Sale," I tried to buy an H-17 subsystem, but they sold out before I picked up the phone. Somehow that doesn't sound like a lack of demand.

Those of us with cassette systems or H-11's have been orphaned. I don't mean to critize, I just wanted to get your attention for a moment. Would you, please, let us back in the Heath Users' Group?

Don Keller
1330 Eden Valley Road
Port Angeles, WA 98362

Complaints About The 100's Name

Dear HUG,

As a relatively new member of HUG, I have only received a few issues of REMark magazine. However, the February issue contained a letter from D.C. Shoemaker complaining about the decision to lump all H-xxx computers under the heading 'H/Z-100' series. I can clearly see his point and wish to add my complaint to his.

I own the Zenith Z-150 (or H-150 if you so choose) with a Cogito 10 meg hard disk, Techmar Captain multifunction board, Mouse Systems PC Mouse, Hayes 1200 baud Smartmodem, Texas Instruments Model 855 printer, and a total of 640K RAM.

I use both MS-DOS and IBM's PC-DOS. Since my Zenith is different than the H/Z89/90 and H-110 and H-120 series, I am finding it confusing to determine which software programs, and which articles appearing in REMark, apply to my unit when the heading states "H-100 computers." Wouldn't is be simpler and less confusing to simply state the specific model the software or article refers to (i.e. H-150/H-160 computers)?

So far I've seen little in the way of software for the H-150 listed in your catalogs and REMark magazine. I assume that's because there are not many HUG H-150 owners. I sincerely hope that will change in the future.

In closing, I will say that the Zenith Z-150 is an excellent machine. The only problem I've had was with the Z-133 color monitor, which I recently had repaired under warranty. Prior to the Z-150, I've owned Radio Shack's Model I, II, & III and most recently an IBM PC. All my IBM programs (with the exception of those written in BASICA) run flawlessly on the Z-150, including Microsoft's Flight Simulator, Lotus 1-2-3, Volkswriter DeLuxe, Power-Base, Clout, The Smart System, and Symphony — also IBM's own products — Word Proof and Personal Communications Manager.

Sincerely,

Clifford W. Coughlin
30 S. Kirklyn Avenue
Upper Darby, PA 19082

Omission In Credits

Dear Walt,

I was so pleased to see my article in the REMark. You really made my day.

However, there was an omission in the credit line. At the time I wrote the article, I was the Director, School of Nursing, Methodist Medical Center of Illinois, Peoria, IL 61636. Any requests for information about the programs or hardware mentioned in the article should be directed to the school.

I retired at the end of June and moved down here to the Ozarks. The closest computer club I have found is in Branson. Which means a 50 mile round trip, at night and in these mountains, to attend meetings. There are a number of people around who are interested and doing something with computers of all varieties. I won't mention that mine is a C-64. One of these days I'll have my own Z-100.

As you can see, I do have my computer up and running. I need to get busy and write the research paper for which all the data is stored in boxes.

Last month, I gave a short seminar to other SCORE members (I am a member) on the introduction of computers into a small business.

Thank you again for printing that article.

Sincerely,

Louise B. Guest
Star Route 1, Box 85
Lampe, MO 65681

BIOS Error On B: R/O

Dear HUG:

A few days ago, my H-89 (using CP/M 2.2.03) hung up with an error message something to the effect "BIOS error on B: R/O". At the time, I had a fairly long letter in memory, but I wound up rebooting the computer and rewriting the letter as best as I could.

At the time, I had a write-protected systems disk in A: and a disk in B: with the Software Tookworks' PIE.COM and various R/W letter files on it. Using B: PIE I finished my letter and used Control E to put it on the B: disk. At the bottom of the screen the usual message "SAVING FILE xxx" appeared, but at the same time, the above error message appeared at the top of the screen. No key had any effect at all, including Control Z (which takes care of a similar hangup in the entry line for SUPERCALC).

After rebooting, I ran A< STAT B:*.* and verified that all the previous files were still on B: and still R/W.

The rewritten letter "SAVED" in normal fashion.

This long explanation leads up to my simple question — is there any way to save the memory and get rid of the error message? I have a complete set of REMark magazines, and would certainly appreciate a reference to any material which has been published on this problem. Do you have any theory as to how it happened?

Very truly yours,

J. (Jack) D. Miner
316 South Rosedale Avenue
Lima, OH 45805

Anxious To Hear From Members

Dear HUG:

I'm very anxious to hear from all members of HUG who are physically disabled and handicapped, and all others who are interested and concerned with these matters.

I'm handicapped and physically disabled, and like so many others, I'll never be able to obtain any form of regular or conventional employment. It's rough duty, but unlike cigarette lighters and ballpoint pens, there are no lifetime guarantees for continued good health and physical fitness.

As the result and consequences of my own near disastrous experiences, and subsequent disappointments and frustrations encountered in the labor and employment markets, I've recently become actively engaged as the originator and guiding force in the organization and formation of:

The *Creative Achievers. A non-profit corporation and association of physically handicapped and disabled persons, of all ages and professions, organized and dedicated to mutual self-help for purposes of becoming productive and contributing members of society by means of some form of beneficial and constructive occupation, employment, or self-employment; and by performing good works wherever and whenever it's practical for its members to accomplish. The members and associates will recapture and reestablish their self-reliance and confidence by helping themselves, and by helping others.

*A euphemism for Common Sense & Chutzpah.

With the advent of high technology, computers and word processing systems, and telecommunications, the challenges and opportunities for the beneficial occupation of our time and wits is so truly great that it boggles the mind just to think about the possibilities. for the first time in history, the majority of the handicapped can easily and actively participate and become constructive and contributing members of society.

Also, the very nature of our reduced physical circumstances coupled with the enhanced means and opportunities for our intellectual collaboration and interactive partnerships, without the hassles, bother, stress, and distractions, of our physical and individual immobility and lack of proximity, irrevocably places us on the leading and cutting edge of the revolution and the accelerating evolutionary changes just starting in the work places, and in the work habits, customs, and practices, of our society.

It's a wonderful and exciting world we have, and the opportunities and challenges are growing and multiplying so dang fast it's impossible to identify and catalog them. I could go on for reams of pages with a million more ideas and thoughts, but for the moment, I think I've said enough; and I hope you and your readers will understand and grasp both the enthusiasm and the sense of what I'm attempting to convey.

The body may be weak, but I'm far from being helpless, my intellectual powers and wits are unimpaired, and I still have my original and undiminished enthusiasm, imagination, drive, and determination. I'm a perennial and upbeat student of life and our times, and I'm looking forward to the challenges and opportunities of my last and most successful career change. Even though it's been an involuntary change, it's quite possible that it'll be the greatest and most rewarding opportunity and adventure of my life, and I'm going to give it my best shot.

Any and all expressions of encouragement, moral support, help, guidance and direction, and constructive criticism, either you or your readers care to volunteer will be greatfully accepted and genuinely appreciated.

All letters will be answered.

Your courteous attention and thoughtful response is cordially invited and will be most especially welcome.

Thanks and sincere best regards,

A. Lloyd Freeman
6116 Everest Way
Sacramento, CA 95842-2898

Heath/Zenith
Users'
Group

HUG Club Update

Maryland Z100 Special Interest Group
c/o HEC
5542 Nicholson Lane
Rockville, MD 20852
Contact Person: Jerry Howritz
Phone: (301) 384-1040 (not after 10:00 pm)
Meet first Monday each month at 8:00 pm at Rockville HEC
Club just starting - looking for members!

EDZELL HUG - Scotland
Box 517 NSGA EDZELL
FPO New York, NY 09518
Contact Person: Lt. Dave Smith
Phone: EDZELL-7254
Group size: 3 and looking for members!
UK/International mailing address: Quarters S, Denstrath Road, RAF EDZELL, Brechin, Angus, SCOTLAND DD9 7XH

The Heath Users' Group of Calgary, CANADA has recently reorganized. The address and phone are the same. New contact person is Bill Jones. They have 20 members and meet the first Tuesday each month at 7:00 pm at the Calgary HEC.

MOHUG (Mobile, AL HUG) has a new contact person, Bob Small. New address and phone: 354 Teakwood Drive, Satsuma, AL 36572, (205) 675-9742. They now have a membership of 22. Meetings are on Fridays, usually the end of the month.

Vectored from Page 33

Part Number	Description of Product	Selling Price	Vol. Issue
CP/M			
885-1219-[37]	CP/M Navigational Program	20.00	31
	AMATEUR RADIO		
HDOS			
885-8016	Morse Code Transceiver Ver 2.0	20.00	42
CP/M			
885-1214-[37]	CP/M MBASIC Log Book (64k)	30.00	23
885-1234-[37]	CP/M Ham Help	20.00	49
885-1238-[37]	CP/M ASCIRITY	20.00	57
885-8020-[37]	CP/M RF Comp. Aided Design	30.00	44
885-8031-[37]	CP/M Morse Code Transceiver	20.00	57
	COMMUNICATION		
HDOS			
885-1122-[37]	HDOS MicroNET Connection	16.00	37
CP/M			
885-1207-[37]	CP/M TERM & HTOC	20.00	26
885-1224-[37]	CP/M MicroNET Connection	16.00	37
885-3003-[37]	CP/M ZTERM (Z100 Modem Pkg)	20.00	34
885-5004-37	CP/M 86 TERM86 and DSKED	20.00	56
885-5005-37	CP/M 86 16 Bit MicroNET Conn.	16.00	61
885-5006-37	CP/M 86 HUGPBBS	40.00	62
885-5007-37	CP/M 86 HUGPBBS Source List.	60.00	62
885-8005	MAPLE (Modem Appl. Effector)	35.00	29
885-8012-[37]	CP/M MAPLE (Modem Program)	35.00	34
885-8023-37	CP/M 85 MAPLE	35.00	45
ZDOS			
885-3019-37	ZDOS 16 Bit MicroNET Connect.	16.00	61
	MISCELLANEOUS		
885-0004	HUG Binder	5.75	
885-1221-[37]	Watzman ROM Source Code/Doc	30.00	33
885-4001	REMark Vol. I Issues 1-13	20.00	
885-4002	REMark Vol. II Issues 14-23	20.00	
885-4003	REMark Vol. III Issues 24-35	20.00	
885-4004	REMark Vol. IV Issues 36-47	20.00	
885-4005	REMark Vol. V Issues 48-59	25.00	
885-4500	HUG Software Catalog	9.75	
885-4600	Watzman/HUG ROM	45.00	41
885-4700	HUG Bulletin Board Handbook	5.00	50
885-3015-37	ZDOS SKYVIEWS	20.00	55

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.

We'll Be Seeing You at the

Fourth Annual International HUG Conference

Hyatt Regency O'Hare
August 9, 10, 11, 1985

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

HUG ID Number: ______________________

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	
U.S. DOMESTIC	$20 □	$17 □	
FPO/APO & ALL OTHERS*	$35 □	$30 □	U.S. FUNDS

* Membership in France and Belgium is acquired through the local distributor at the prevailing rate.

Heath/Zenith Users' Group
Hilltop Road
Saint Joseph, Michigan 49085

POSTMASTER: If undeliverable, please do not return.