



REMark®

Volume 7, Issue 7 • July 1986

P/N 885-2078 Issue 78

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

Introducing Bill Adney
Contributing Editor Starting August 1986

\$2.50

INTERNATIONAL H86 UGCON





Controlled Data Recording Systems Inc.

Quality Products and Support for the Heath/Zenith Community

New for the H/Z89-90 Computer Users: The Super RAM 89 Package

Now get the speed and power of a high capacity Ram Drive System at a reasonable price.

Using the 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.

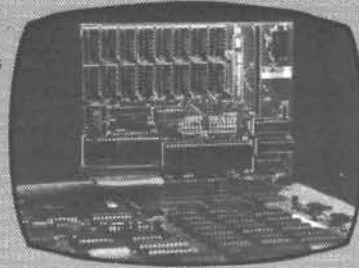
Now get RAM 89
HDOS DRIVER Software Only \$35

RAM 89 DVD supports RAM SY0: and SY1:, added autoload features, plus more!

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 attaches to any of the versions of CP/M 2.2 bios used in the H/Z89-90.

GET A FULL MAGABYTE OF RAM FOR YOUR H/Z89-90

Board 1 includes hardware manual and ram drive software with no ram: \$190.00. Each 256k bank, add \$56.00. Board 2 (must have board 1). With no ram, no clock, no SASI \$90.00. Each 256k bank, add \$56.00. Ask about clock, SASI pricing.



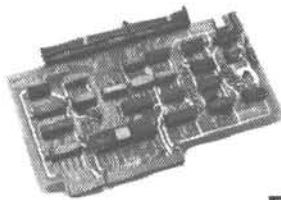
Ram 89 comes as a two card set that plugs into the left hand (16k expansion) side of the computer. No computer modifications required.

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.

FOR THE H/Z89-90 COMPUTERS

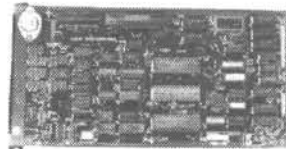
**THE ONE CONTROLLER
FOR 8"
& 5.25"
DRIVES
THE
FDC-880H
PRICE: \$395**



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 sector controller

**THE FDC-H8 DOUBLE DENSITY 8"
AND 5.25" CONTROL-
LER FOR THE H8 COM-
PUTER PRICE \$495**



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

- Direct memory access (DMA) data transfer.
- Hard sector controller (H17) incorporated on the board.
- Runs with the standard 8080 CPU card and with Z80 CPU upgrades.

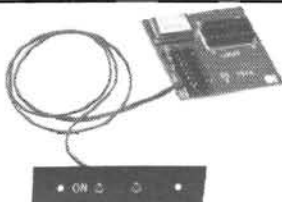
MODIFY89 Allows owners of the FDC-880H or the FDC-H8 to read/write to the following disk types:

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., Xerox 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 PRO-
GRAMS 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.

SUPER DRIVES 85

A powerful floppy drive package for the Z100 or any computer that can use 8" floppy drives.

Two extended technology 5.25" drives in a package with interface cables, case and power supply, that look exactly like standard 8" floppy drives to the computer. Now have 2.4 megabytes of floppy file storage capacity in one 5.25" package.

ONLY \$695 LIST
Dealer inquiries invited.



For information about these products contact:
Controlled Data Recording Systems Inc.
7210 Claremont 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.
New - C.D.R. Systems Bulletin Board (619) 560-8929

NOT PROTECTED

Genie™

Software Magic from Advanced Software Technologies



Genie is a memory resident application. This means that once you load Genie it is always available for you to use. Just hit the magic keys and Genie will appear (Shift-Shift: No function keys lost.) You can have Genie perform various tasks, and when you finish Genie goes away and you are back where you started.

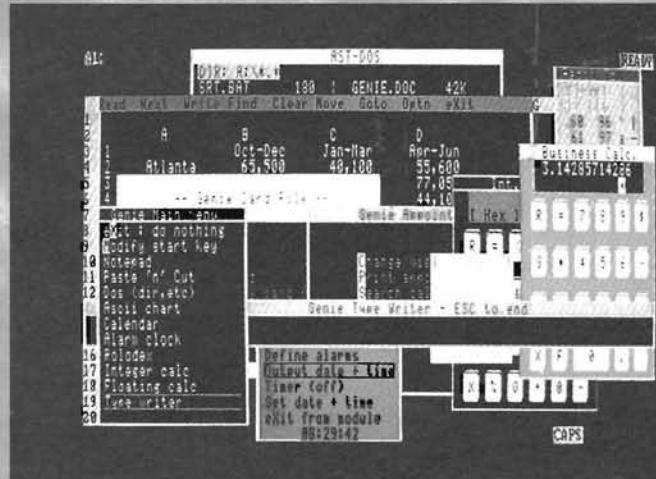
Here is what you get with Genie:

- **KEY MAPPER** - Redefine any key on the fly, store long commands in a single key, and save many maps on disk.
- **TrueDOS™** - Run any MS-DOS program inside the window. FORMAT, SORT, BACKUP, compile, all without leaving your application.
- **CALCULATORS** - you get two calculators, one regular floating point calculator, and programmer's calculator for base conversions and bit operations.
- **NOTE PAD** - you'll never have to hunt for paper and pen again. Simply call up Genie's Note Pad and jot down what you need. Or expand the buffer to 56K: Instant editor.
- **CUT AND PASTE** - Cut text from any place on the screen and output it later. Cut long commands off the screen and into your KEY MAPPER, move data from your spread sheet to your word processor. Instant integration.
- **CALENDAR** - schedule appointments for any year up to 9999 keep track of expenses, search and print the calendar.
- **ROLODEX** - a name address and telephone number list that you can search any time. No limit on the number of cards. You can even output an address directly to your word processor.
- **ALARM CLOCK** - Have your Genie remind you of appointments, set alarms to ring at any time on any day. A window appears with a reminder of what each of the 8 alarms is for.
- **ASCII Table** - programmers never have to leaf through big books to find the ASCII value of a character.
- **TYPEWRITER** - Knock out quick memos any time, even send ESCAPE codes to your printer. Cut a portion of a spread sheet and paste into the typewriter for quick printouts.
- **SCREEN SAVER** - Automatic phosphor protector for your tube. Genie will even let you blank the screen manually to discourage peepers.
- **COMMAND STACK** - Lets you access up to the last 2K worth of commands you typed.

Best of all with Genie you only have to load what you use. If you don't need to use an ASCII chart don't load it. Need extra alarms? then load two alarm clocks. Your own personal Genie.

And now Genie user's have the advantage of Genie accessories, which can be installed and accessed like any Genie function - any time:

- a **Scientific Calculator** more powerful than most real calculators. Has a full range of math, science and statistical functions and very high accuracy.
- a **Spelling Helper™** To help you find the correct spelling without leaving your word processor.
- a **Print Spuffer™** (spooler/buffer) to speed and manage all your printing.
- a development pack to let you write your own pop up programs, including many routines for windowing, and source code for our Rolodex. Writing pop ups is a snap. (Please call for details)



Shown here is Genie "popped up" on a Z-110 running Lotus 123. From the left are: The Genie main menu, the Genie rolodex style card file, the Genie notepad containing data cut from Lotus, the Genie DOS performing a directory command, the Genie alarm clock (at the bottom,) the Genie typewriter, Genie calendar, Genie Cut and paste, Genie Calculators, and the Genie ASCII table.

Only Genie Gives you so much for so little.



**ADVANCED
SOFTWARE
TECHNOLOGIES**

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

ORDERS ONLY CALL

1-800-437-4400 Ext 800

Yes please rush me the following software for my

_____ Z110/120 _____ PC compat.
 _____ Copies of Genie @ \$54.95 _____
 _____ Spelling Helper(s) 29.95 _____
 _____ Print Spuffer(s) 24.95 _____
 _____ Scientific calc. 19.95 _____
 _____ Genie pack (I want it all) _____
 Saving \$40 \$89 _____

Sub total _____
 Check encl shipping 4.00 _____

M/C NY Tax (8.25%) _____

VISA TOTAL _____

Card Number _____

Expires _____ Your Tel Num _____

Name _____

Address _____

Company _____

City/ST/Zip _____

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

Production Coordinator Lori Lerch
(616) 982-3794

Secretary Margaret Bacon
(616) 982-3463

HUG Bulletin Board (616) 982-3956

Contributing Editor William Adney

Contributing Editor Joseph Katz

Printer 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 is a registered trademark of the Heath/Zenith Users' Group, St. Joseph, Michigan.

Copyright (C) 1986, Heath/Zenith Users' Group.



HUGCON86

Bob Ellerton 7

SEEK.COM: A CP/M Assembly Language Learning Adventure — Part 1

M.D. Zapolski, Sr. 13

Mainstream Computing

Joseph Katz 17

Heath/Zenith Related Products

Jim Buszkiewicz 22

Z-GRAPH-100 Graphics Package Review

Paul W. Simmons 23

A Program For Upgrading The H-89 With A 16-Bit Single-Board Computer

Phillip L. Emerson 27

Packet Radio Description And A Simulator — Part 1

Luis E. Suarez 33

Paranoia Reigns Supreme Or Is Copy Protection Really Necessary?

Jim Buszkiewicz 39

Buggin' HUG

..... 43

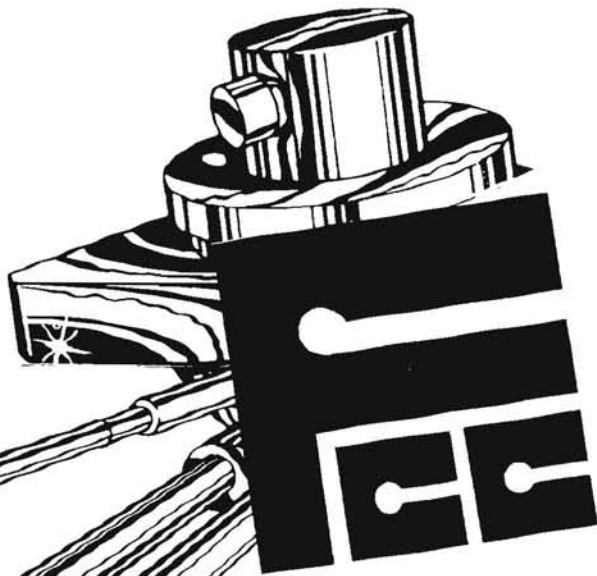
Index of Advertisers

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

For Experts Only: A Review Of The Seidl Make Utility <i>Pat Swayne</i>	45
More RAM And A RAMDISK With MegaRAM-150 For The H/Z-100 PCs <i>Richard L. Mueller, Ph.D.</i>	47
Keep Your H/Z-100 Blinking For MS-DOS 2.0 Or Greater <i>M. D. Holmberg II</i>	51
ZPC Update #7 <i>Pat Swayne</i>	59
Keyboard Codes, Menus, And TURBO Pascal <i>J. R. "Jeff" Jeffrey</i>	63
Patch Page <i>Pat Swayne</i>	67
On The Leading Edge <i>William M. Adney</i>	69
HUG Price List	75
A Winchester For The '89 — Part Six <i>Peter Ruber</i>	77

<i>Advanced Software Technologies</i>	3
<i>Advent Products</i>	62
<i>American Cryptronics</i>	41
<i>Analytical Products</i>	68
<i>ByteSize Concepts, Inc.</i>	50
<i>CDR Systems, Inc.</i>	2
<i>FBE Research Company, Inc.</i>	32
<i>First Capitol Computer</i>	6
<i>Heath Company</i>	46
<i>Intersecting Concepts</i>	81
<i>Jay Gold Software</i>	38
<i>Micronics Technology</i>	16,26
<i>New Orleans General Data Services</i>	26
<i>Paul F. Herman</i>	81
<i>S&K Technology</i>	32
<i>Secured Computer Systems</i>	82
<i>Seidl Computer Engineering</i>	83
<i>SigmaSoft & Systems</i>	42
<i>Software Wizardry</i>	58,66
<i>UCI Corporation</i>	76
<i>Veritechnology Electronics Corp.</i>	35
<i>Barry Watzman</i>	84
<i>Wolfpac Technology</i>	58

On The Cover: Featured at HUGCON86 in Chicago, IL (August 15, 16, 17) will be these Heath/Zenith computers, along with a whole lot more. See you there!



First Capitol Computer

NEW!

Z-148 38 Mb System

\$1999



**First Capitol
Computer Exclusive**

- * blazingly fast system
- * 8 mHz with no wait states
- * new high speed disk controller
- * 640 K RAM

- * amber graphics monitor
- * 38 Mb disk and 1 floppy
- * serial and parallel ports
- * color graphics adaptor

also available separate 38 Mb
hard disk upgrade kit for the
Z-148 or Z-158

**new lower prices on Z-158 systems
CALL FOR PRICES**

**First
Capitol
Computer**

ZENITH | **data
systems**

AUTHORIZED SALES AND SERVICE

First Capitol Computer
1106 First Capitol Dr.
St. Charles, MO 63301
(314) 724-2336

Single-Step Through The Fifth International Heath/Zenith Users' Group Conference

Bob Ellerton
 HUG Manager

A Grand Tour! This simple phrase could be used to describe the events of the Fifth International Heath/Zenith Users' Group Conference scheduled for August 15-17, 1986. Although it's difficult to write about an event that's two months down the road, I can say with confidence this year is going to be a winner! With the International Conference still in the planning stages, 80 percent of the available floor space for exhibits is already being used! And, as you will see by the Discussion Group Schedule, we are loaded with speakers (some old and some new) to cover just about all aspects of personal computing. But, before I get carried away, let's take a "single-step tour" of the Fifth International Heath/Zenith Users' Group Conference.

As you arrive at the Hyatt Regency O'Hare on Friday, August 15, 1986, you will observe Margaret Bacon and her crew of hard working HUGgers manning the HUG Registration Booth just to the left of the main entrance. Be sure to check in with Margaret! She has some special information for you that will make life a lot easier during the busy weekend. Margaret's area will be invaluable for the duration of the Conference as she has the scoop on the Hyatt and the proper people to contact for any questions you may have on the facility.

Next, visit the Exhibit Area located on the lower level of the Hyatt. Again, the entire lower level will be used to house the many loyal Heath/Zenith exhibitors you have become familiar with through the pages of REMark. Drop in on the folks at the various booth locations. Look at the new products, talk about your experiences with some of the familiar products and most of all, meet the people who make the International Conference enjoyable and possible.

Exhibits aren't everything! Friday evening brings the Grand Opening of HUGCON86 and some important news from Chas Gilmore, Vice President of Product Development for Heath

Company. Last year Chas teased us a bit with the front cover from the 1986 Heathkit Catalog. This year he's back with the actual products developed by his group over the past year. Be prepared to ask some questions, give him your suggestions and needle him about 1987.

Marketing Research is important to any company. Miles Hoffman, Manager of Marketing Research for Heath Company, will give us a look at the typical HUGger (I can tell — this is gonna be good). Miles has the numbers! He will give you a look at the industry from Heath's viewpoint and possibly answer a few questions for all of us on the direction of Heath's engineering efforts. Again, be prepared to ask some questions and supply your input. Both Miles and Chas are open to suggestions, questions and comments.

Good morning! It is now Saturday! The Registration Booth opens at 7:30 am. Now it's time to visit the booth locations you missed the night before. The Exhibit Area and the first Discussion Groups begin at 9:00 am. As you can see, you've got to make a decision as to whether your mind is in gear to absorb all that technical data or whether you want to make some preliminary visits with the people at the booths. In any case, you're going to be busy.

The Discussion Groups for 1986 number 23 with 25 speakers prepared to answer your questions on just about any subject you could imagine. Scan the list! Talks for the beginner, graphics, operating systems and hardware "bull" sessions are only a few from a large complement of subjects available to you. The Discussion Group Schedule is arranged to repeat some of the popular sessions up to three times during the weekend. Be sure to carefully review the Conference Program to select those topics you are most interested in. Sessions will be available for users of the H8, H89, H/Z-100 and the PC-Compatibles. Hopefully, there

will be something for everyone. Heath/Zenith Technical Consultants will be on hand for all your hardware questions. (We'll have to see how they come up with a busy signal in person.)

Gosh! Now you've had a long day visiting the Exhibitor Area and the various Discussion Groups and we're only beginning. Dinner time! Get a couple of drinks at the Dinner Warm-up scheduled for 7:00 pm. At 8:00 pm, grab the nearest seat for a great meal served by the Hyatt. After dinner, we will be hearing from Mr. Lou Frenzel. Lou was the Product Line Manager for Heath Company about the time Heath introduced a thing called the H8 (somewhere between the Stone Age and PC-Compatibles). He was the man responsible for another invention he called HUC (sound familiar). Lou will be giving us a brief history lesson along the way to some very interesting and new technology.

Dinner on Saturday night has been a tradition for the Heath/Zenith Conference since its beginning five years ago. Another tradition started back then also — Grand Prize Drawing!. In the past, we have given short and obvious clues as to the prizes cooked up for each conference. This time, however, we are going to vary a bit from this policy and just tell you that there will be a number of lucky individuals walking out of the party with a tremendous amount of computing power. We are planning something a little special for Saturday evening and I would have to urge all of you to be present at this special function. So — "Come on down!" (PLEASE NOTE: You must be present to win.)

Good morning! It's Sunday and I would hope that we are all still moving and breathing by this time. We continue the Fifth International HUG Conference at 9:00 am sharp with more Discussion Groups and another chance to get at the Exhibitors for some last minute purchases and questions. Discussion Groups continue up to noon and the Conference will close again for another year at 2:00 pm.

Well, that about wraps it up — did I forget anything? Oh yeah! Each year, most of our Exhibitors donate a number of door prizes to be given away during the Conference activities. No exception this year! Prizes donated by the Exhibitors will be selected at random for a special prize drawing each hour the Exhibit Area is open. Numbers of the winners will be displayed at the HUG Booth for the duration of the Conference. The prize and the name of the winner will be announced as each winner collects the prize.

Anything else? Yup! If you are selling, trading/wanting or in need of some special assistance, bulletin boards will be placed in a convenient location within the Exhibit Area for personal messages only. Vendors should contact the Heath/Zenith Users' Group (HUG Booth #1) for commercial advertising space or call Margaret Bacon at HUG ((616) 982-3776) for arrangements to have your material made available for display to the attendees.

Wanted/Trade — This bulletin board will be available for those of you wishing to purchase or trade items.

For Sale — This bulletin board will be available for those of you wishing to sell computer related items.

Messages — This bulletin board will be available for general messages of any nature.

One more thing! The guys from the Heath/Zenith Computers and Electronics Centers are preparing some of those great deals you have either heard about or enjoyed at earlier HUG Conferences. This year, the Heath Store will occupy the entire BOAC room as you enter the Exhibit Area. Word has it, that many of the Exhibitors will follow this lead by making some fantastic offers to users attending HUGCON86. So, make sure your plastic is ready for the trip to the Hyatt.

What else? I know I've probably missed a few details. I hope this brief description of HUGCON86 will help you decide to join in the fun this year. I'll be looking forward to meeting as many of you as possible at the Fifth International Heath/Zenith Users' Group Conference.

HUGCON86 In Review

August 15-17, 1986 promises to be a busy exciting weekend. The Staff of the Heath/Zenith Users' Group is working hard to insure a great time for all that choose to join with fellow users at the fifth anniversary of the International HUGCON. The Hyatt Regency O'Hare provides an excellent atmosphere for an exciting, yet relaxing weekend. If you have not attended previous HUG Conferences, this is a good time to get into the swing of what the Heath/Zenith Users' Group is all about. The "old pro" at HUGCONs will find many new items of interest along with some new topics at the Discussion Groups currently scheduled.

NOTE: The schedules, exhibitors and activities of the Fifth International Heath/Zenith Users' Group Conference are subject to change. A final program of events will be available at the HUG Registration Booth on the days of the Conference. Changes that occur during the Conference will be announced within the Exhibit Area during normal hours. Tickets for those of you who have pre-registered are now being mailed every two weeks. If you have not received your tickets, please contact Margaret Bacon, HUG Secretary, at (616) 982-3776. If you are planning to attend for one day only, you may purchase tickets at the HUG Registration Booth. One-day tickets may be used for Discussion Groups and access to the Exhibit Area, however, one-day ticket holders will not be eligible for door prizes selected in the Exhibit Area or at the HUG Dinner Party. The \$25.00 regular ticket entitles the holder to attend all activities of the Fifth International Heath/Zenith Users' Group Conference (you must be present to win prizes at the HUG dinner party on Saturday night). We cannot guarantee seating space for all attendees to each Discussion Group. Seating will be assigned on a first-come, first-served basis.

SPECIAL NOTICE: Dates furnished on the Hyatt Regency Room Registration Cards have the Conference listed from August 14-17, 1986. Be sure you fill in the dates that you will be staying at the Hyatt so there is no mistake about your reservations and room requirements. Remember, actual Conference dates are from August 15-17, 1986.



Fifth International Heath/Zenith Users' Group Conference

Schedule of Events

Friday, August 15, 1986

- 2:00 pm Registration Booth Opens
- 3:00 pm Exhibit Area Opens
- 7:00 pm Exhibit Area Closes
- 8:00 pm Registration Booth Closes
- 8:00 pm Grand Opening Warm-up (Rosemont)
- 9:00 pm Grand Opening and Introductions (Rosemont)

Saturday, August 16, 1986

- 7:30 am Registration Booth Opens
- 9:00 am Exhibit Area Opens

Morning Discussion Groups Start Times

9:00 am 10:30 am

Afternoon Discussion Groups Start Times

12:00 pm 1:30 pm

3:00 pm 4:30 pm

- 5:00 pm Exhibit Area Closes
- 5:30 pm Registration Booth Closes
- 7:00 pm Dinner Warm-up Open (Rosemont)
- 8:00 pm Dinner, Keynote Address, Awards and Prizes (Rosemont)

Sunday, August 17, 1986

- 7:30 am Registration Booth Opens
- 9:00 am Exhibit Area Opens

Discussion Groups Start Times

9:00 am 10:30 am

- 12:00 pm Registration Booth Closes
- 2:00 pm Close of Fifth International HUG Conference

1986 International HUG Conference

Discussion Group Schedule

Saturday August 16, 1986

Time	Room	Subjects	Speaker
9:00 am	Ozark	DeskTop Publishing, or The Dot Matrix Printer And Beyond	Arthur Seebach
	Pan Am	Z-200 Firmware	Brian Barnes
	Allegheny	HUG Software...What You're Missing	Jim Buszkiewicz
	Eastern	Trouble Spots For The Beginning 'C' Programmer	Jack Purdam
	American	Introduction To Computers	Ron Hackney
	TWA/NWO	HERO 2000	Ron Johnson
	Forum	Hardware 'Bull' Session For Old And New Computer Users	Bruce Denton
	United B	Introduction To AutoCAD And Its Enhancements	John Roach
10:30 am	Ozark	Z-120 Emulation On The Zenith PC's	John Guenther
	Pan Am	Business Graphics For The Z-100	Janet Hirsch
	Allegheny	Z-100 And Z-100 PC Secrets Revealed	Pat Swayne
	Eastern	ZLOGO: Learning, Graphics, And A/I For The Z100 And Z100 PC's	Peter G. Halverson
	American	Tharting With LISP	Walt Bilofsky
	TWA/NWO	Starting With XENIX	Scott Cutshall
	Forum	Software Workshop	Bill Parrott
	United B	Advanced Programming In 'C' Graphics, Utilities, Portability	Dave Haskell
12:00 pm	Ozark	HyperAccess For The Z-150, Z-170, & Z-240	Matt Gray
	Pan Am	Introduction To Local Area Networks (LAN)	Steve Hesterman
	Allegheny	Device Drivers, TSR Routines, And Device Configuration For MS-DOS	Bill Rothman
	Eastern	Goal Oriented Data Bases — What To Do Versus How To Do	Dov Wiezman
	American	MS-DOS Version 3 For H/Z-100 And H/Z-100 PC's And Windows	Bill Adney
	TWA/NWO	Introduction To Computers For The Completely Intimidated	Susan Hayes
	Forum	PC Hardware Workshop	Bob Harris & Rick Simpson
	United B	Introduction To Computer Based Instrumentation	Jim Lytle
1:30 pm	Ozark	DeskTop Publishing Or The Dot Matrix Printer And Beyond	Arthur Seebach
	Pan Am	Z-200 Firmware	Brian Barnes
	Allegheny	HUG Software...What You're Missing	Jim Buszkiewicz
	Eastern	Trouble Spots For The Beginning 'C' Programmer	Jack Purdam
	American	Introduction To Computers	Ron Hackney
	TWA/NWO	HERO 2000	Ron Johnson
	Forum	Hardware 'Bull' Session For Old And New Computer Users	Bruce Denton
	United B	Introduction To AutoCAD And Its Enhancements	John Roach
3:00 pm	Ozark	Z-120 Emulation On The Zenith PC's	John Guenther
	Pan Am	Business Graphics For The Z-100	Janet Hirsch
	Allegheny	Z-100 And Z-100 PC Secrets Revealed	Pat Swayne
	Eastern	ZLOGO: Learning, Graphics, And A/I For The Z100 And Z100 PC's	Peter G. Halverson
	American	Tharting With LISP	Walt Bilofsky
	TWA/NWO	Starting With XENIX	Scott Cutshall
	Forum	Software Workshop	Bill Parrott
	United B	Advanced Programming In 'C'; Graphics, Utilities, Portability	Dave Haskell

Time	Room	Subjects	Speaker
4:30 pm	Ozark	HyperAccess For The Z-150, Z-170, & Z-240	Matt Gray
	Pan Am	Introduction To Local Area Networks (LAN)	Steve Hesterman
	Allegheny	Device Drivers, TSR Routines, And Device Configuration For MS-DOS	Bill Rothman
	Eastern	Goal Oriented Data Bases — What To Do Versus How To Do	Dov Wiezman
	American	MS-DOS Version 3 For H/Z-100 And H/Z-100 PC's And Windows	Bill Adney
	TWA/NWO	Introduction To Computers For The Completely Intimidated	Susan Hayes
	Forum	PC Hardware Workshop	Bob Harris & Rick Simpson
United B	Introduction To Computer Based Instrumentation	Jim Lytle	

Sunday, August 17, 1986

9:00 am	Ozark	Z-200 Firmware	Brian Barnes
	Pan Am	Software Workshop	Bill Parrott
	American	HUG Software...What You're Missing	Jim Buszkiewicz
	TWA/NWO	PC Hardware Workshop	Bob Harris & Rick Simpson
	Eastern	Starting With XENIX	Scott Cutshall
10:30 am	Allegheny	Introduction To Computers	Ron Hackney
	Ozark	DeskTop Publishing, Or The Dot Matrix Printer And Beyond	Arthur Seebach
	Pan Am	Device Drivers, TSR Routines, And Device Configuration For MS-DOS	Bill Rothman
	American	Z-100 And Z-100 PC Secrets Revealed	Pat Swayne
	TWA/NWO	Hardware 'Bull' Session For Old And New Computer Users	Bruce Denton
	Eastern	MS-DOS Version 3 For H/Z-100 And H/Z-100 PC's And Windows	Bill Adney
Allegheny	Introduction To AutoCAD And Its Enhancements	John Roach	

Conference Vendor List

Advanced Software Technologies	New Orleans General Data Services
AI Davis	Newline Software
CDR Systems, Inc.	Public Brand Software
Condor	QuikData — H-Scoop
D-G Electronics	S&K Technology
Disk Movers	Sextant
EWDP Software	SigmaSoft & Systems
Floppy Disk Services	Software Toolworks
Gemini	Software Wizardry
Graphnet Systems	UCI Corporation
Graymatter Application Systems	Veritechnology Electronics Corporation
Heath Users' Group	Barry Watzman
Hilgraeve, Inc.	Zeducorp
Kres Engineering	Zenith Data Systems
MPI	

**I
N
T
E
R
N
A
T
I
O
N
A
L**
H86

Heath/Zenith Users' Group • Hilltop Road • St. Joseph, Michigan 49085 • (616) 982-3463

**INTERNATIONAL
HEATH/ZENITH USERS' GROUP
CONFERENCE**

Official Conference Registration Form

**O'Hare Hyatt Regency
Rosemont, Illinois
August 15, 16, 17, 1986**

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 15, 16, and 17, 1986. 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, 1986. Cancellation will not be accepted after this date.
Sorry, We cannot accept purchase orders**

SEEK.COM

A CP/M Assembly Language Learning Adventure

Part 1

M. D. Zapolski, Sr.
226 N. West Avenue
Bridgeton, NJ 08302

Introduction

How many times have you written a BASIC program only to find it took too long to run, or that BASIC didn't give you the program control you needed? What can you do to improve the program's operating characteristics? Generally, you re-analyze the program, rewrite its routines, try to compress its length, and the like. If you're fortunate enough to own a compiler, the speed problem is solved. But then you're stuck with a good size runtime module (16k) that absorbs some of the available programming memory. Even though this is an improvement over the size of the MBASIC interpreter (25k), there are alternatives to compiling and BASIC program revision. Why not use assembly language? Yes, I know its instructions are somewhat cryptic, and it is less forgiving when programming errors are made. However, consider its advantages — extremely fast execution, and total program control over the computer's operation. As important as these points are, there is a cost consideration. The tools needed to use assembly language (hereafter referred to as ASM) are usually distributed with the purchase of the CP/M operating system. So, the price is right!

Learning to use ASM can be a trying experience. There are few well written courses (or books) on this subject. Those that exist often use examples that are machine specific and can't be used directly on the H/Z-100 computer. Likewise, explanations of example programs frequently lack insight into how they were developed, or why they had to be structured in a particular way. Through this series of articles, I hope to fill these gaps, and provide the reader with an understanding of the mechanics of transforming an idea into an operating 8080/85 ASM program for the H/Z-100. At the same time, I intend to explain the program development steps and problems encountered in this process.

Before continuing, I should say that by profession I am not a programmer. As a result, the programming methods I use may not

be the most efficient by others' standards. However, in an effort to give you a starting point to learn from, efficiency may be sacrificed for comprehension's sake. For those of you who have ASM experience, please bear with me when I elaborate on familiar details. I'll thank you in advance for your patience . . .

References

This series will refer to other documents which, if reviewed, may be helpful. These references will be noted in the following manner: [ref 1, pg. 41]. This example states there is more information in reference #1 (Z100 User's Manual) starting on page 41. Part 1 of this series includes a Master Reference List (Table 1).

Table 1
Master Reference List

1. Z-100 User's Manual
 2. CP/M-85, Vol. 1
 3. CP/M-85, Vol. 2
 - a) CP/M System Interface Manual, pg. 91
 4. REMark Magazine
 - a) Issue 40, pg. 36
 5. Sextant
 - a) Spring 1983, pg. 69
 6. 8080 Assembly Language Course (Heath P/N EC-1108)
 - a) Table of Opcodes (plasticized placard)
-

The .COM File

Before proceeding with the main article, I believe a review of what a .COM (COMmand) file is, and how it's created will be beneficial. A .COM file is a machine language file (binary 1s and 0s) which can be directly executed by the CP/M Operating System (hereafter referred to as OS). At the OS prompt (e.g. A>), the user types the .COM file name (without the .COM extension). The OS then:

1. Looks for the file on the default drive.
2. If found, it loads the file into the Transient Program Area (TPA) starting at address 0100H.
3. Executes the program's instructions in the order specified.
4. When finished, the OS redisplay its prompt awaiting further user response.

The .COM file is created by first preparing an ASM file. This requires the use of a program known as a Text Editor, or simply an editor. Essentially, the editor allows you to use your computer like a typewriter and a blank sheet of paper. The ED.COM program [ref 2, pg. 2-100] provided with CP/M-85 may be used for this purpose, but it has severe drawbacks. It has an "invisible" character pointer that is difficult to use, and its commands are cryptic. Although other editors exist, I use PIE.COM. While I'm not associated with the Software Toolworks, I recommend PIE — its inexpensive, versatile, and is easy to work with. Regardless of which editor you use, it's important that it does not imbed control codes in the text file. Those codes can inhibit proper file assembly. However, to be useful, your editor should at least have insert/delete lines and character capabilities.

Once you've prepared the ASM file (also called a source file), it is assembled. In other words, converted to a .HEX file through the use of CP/M's ASM.COM program [ref 2, pg. 2-3]. This file is printable (may be listed on a printer or the CRT screen). It contains the INTEL hexadecimal equivalents of the ASM instructions in ASCII form. Hence, INR B (increment the B register) would equal 04H [ref 6a]. Then the .HEX file is converted to a .COM file after submission to CP/M's LOAD.COM program [ref 2, pg. 2-141]. As a .COM file, the program is now directly executable. This means no additional program (other than the OS) is required to run it. By comparison, MBASIC.COM is required to run a .BAS program. Thus, a .BAS program is not directly executable. In summary, four items are needed to create a .COM file:

1. Your idea expressed in ASM mnemonics (instructions).
2. A text editor.
3. ASM.COM — The 8080/85 Assembler.
4. LOAD.COM — The Loader.

The Idea

After writing my first computer article, I became curious about how many lines, words, and sentences it contained. The notion of determining these attributes manually was simply out of the question. Why not let the H/Z-100 do it? Great idea! I opted to use ASM. After all, how hard could it be? Indeed, I was about to embark on a learning adventure that would make me appreciate the labors of professional programmers. So, we want a program that will take a text file, and examine it to count the number of lines, words, and sentences. Accordingly, it should "seek out" this information. Hence the program name SEEK.COM. Now, let's outline the 5 basic steps in the process of transforming our idea into an ASM program:

1. Separate the idea into function modules creating a Basic Flow Chart.
2. Develop specific program details for each module.

3. Combine the detailed modules into a composite program.
4. Analyze the program and resolve any conflicts between the modules.
5. Assemble the program and thoroughly test its operation.

Step 1 — Basic Flow Chart

Without too much effort, a simple task may be made complex if looked at in too much detail. Consequently, you should separate your idea into discrete, summary level functions (modules) and develop a basic flow chart. For example, our program will count words. Accordingly, that task will be one of our modules. Try not to clutter your mind with the mechanics of "how do I get the program to count words." That will be dealt with later. Just focus your thoughts on the main objective — the BASIC flow chart. When you've finished, you should have come up with 3 to 5 modules. This depends on whether you considered the support function modules. They perform activities like loading the disk file into memory, and printing the results of the counting modules on the video screen. Finally, arrange all the modules in a logical sequence to form a Basic Flow Chart (Chart #1). Note, there is also a Start (Module #1) and End point on the chart. Don't be concerned if you've interchanged modules 3-5. It makes little difference to the program.

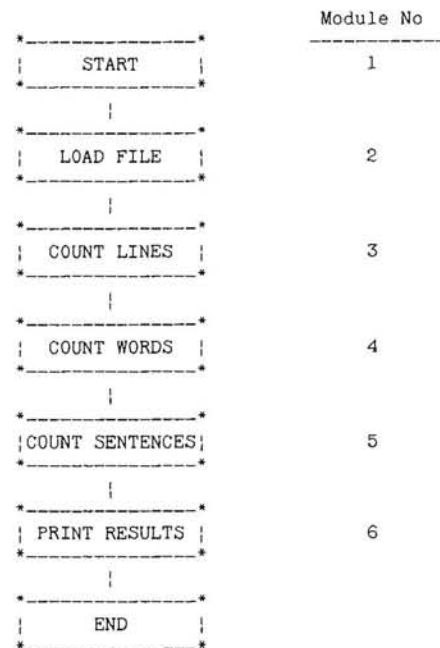


Chart 1
SEEK.COM Basic Flow Chart

Step 2 — Develop Specific Program Details

This will be a lengthy section involving many details. As a result, it will be subdivided by the module numbers shown on Chart #1:

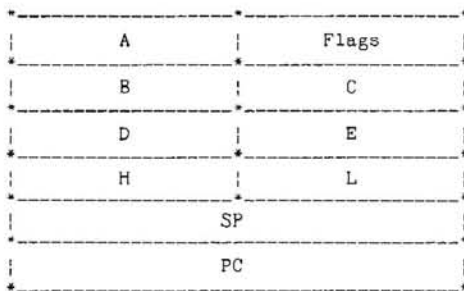
1. Module 1 (START) — Review 8080/85 registers, ASM file organization, text files, & CP/M functions.
2. Module 2 (LOAD FILE) — Open the disk file, load it into RAM, and error trapping.
3. Module 3-5 (COUNT...) — Program steps to count the lines, words, and sentences of the text file.

4. Module 6 (PRINT RESULTS) — Take the data from modules 3–5 and display them on the video screen.

The END block is simply a way of showing the program's end, and needs no further explanation.

Step 2 — Module 1 (Start)

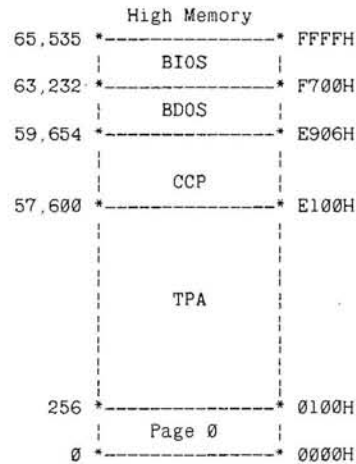
From an operational viewpoint, the 8080/8085, 8 bit microprocessors are essentially the same. They have the same registers (Figure 1), but the 8085 has two extra opcodes (instructions): SIM — set interrupt mask, and RIM — read interrupt mask. The 8080/85 can directly address 65,536 bytes of memory. The CP/M OS organizes (or maps) this memory into specific areas for its use (Figure 2). Of these areas, only the BIOS should vary from one computer system to another. We are currently interested in the Transient Program Area (TPA). Since that is where SEEK.COM will be loaded. After loading, CP/M will jump to location 0100H and start executing the program's instructions. Normally, all CP/M application programs are limited to this area (0100H to E100H, about 57k). Experienced programmers sometimes intrude upon the CCP area, but we will not. The operational details of CP/M's CCP, BIOS, and BDOS are beyond the scope of this series. Suffice it to say that they coordinate the activities of the computer system's disks, keyboard, etc.



Register	Size(bits)	Purpose
A	8	Data I/O to data ports, Math and logical operations, counter
B,C	8	Double add, counter, RAM pointer
D,E	8	Double add, counter, RAM pointer, HL exchange
H,L	8	Double add, counter, RAM pointer, DE exchange, interplay with PC and SP
SP	16	Data exchange, address storage
PC	16	Contains the address of the next instruction to execute
Flags	5	Holds the current status of the condition flags

Figure 1
8080/85 General Register Information

The effective organization of an ASM file is a simple matter, yet quite important. Advanced programmers may intertwine program instructions with messages and data bytes (DB), but this is confusing to beginners. The file structure I use is shown in Figure 3. At the beginning is a place for the date, program name, assumptions and general notes. These are entered in the form of comments with a comment marker (;) prior to any characters. To separate this, and other areas, I use an "area separator" (blank line). This serves as a visual aid which sets apart the major ASM file work areas.



Basic Input/Output System (BIOS) — This area contains the customized routines that operate the computer's peripherals.

Basic Disk Operating System (BDOS) — This area contains the general programs for the operation of the peripherals.

Console Command Processor (CCP) — This area contains resident CP/M programs like DIR, ERA, REN, SAVE, TYPE, and USER.

Transient Program Area (TPA) — This is the working area of memory where your executable programs & data are stored.

System Parameter Area (Page 0) — This area contains important information such as the BIOS and BDOS jump vectors, default File Control Block (FCB), current drive and user #, IOBYTE, and the restart area.

Figure 2
H/Z-100, CP/M-85 (Ver 2.2.103) Memory Map

First, the Equate Area. An equate is an assembler directive used to assign numeric values to labels. To this end, BDOS EQU 05H tells the assembler to use the number 05H anytime it finds the label BDOS.

(These addresses may not be accurate for other CP/M versions. Obtain a copy of TELL.COM from a CP/M Bulletin Board. It will provide the values for your system)

```

<date> <program name>
;
; <program assumptions, general comments>
;
; * Equate Area
; BDOS EQU 05H
;
; * Program Instruction Area
; ORG 0100H
<label> <operator> <operand> ; <comment>
BEGIN CALL OPNRD ; Open file & read
; into RAM
; <last program instruction>
;
; * DB,DW,DS, Message Area
MSG1 DB 'This is an example message',0
;
; END BEGIN

```

Figure 3
Example ASM File Organization

The Program Instruction Area follows the Equate Area and always starts with the ORG directive. It tells the assembler where (RAM address) to start the 1st line of the machine code. The next line is the 1st program instruction. Added to this 1st line should be a label, such as START or BEGIN. Labels and comments are important! Please don't underestimate the value of using them in ASM programs. If you found them useful in BASIC programming, you'll really see their utility in ASM programs.

The final area is located after the last program instruction. It is reserved for assigning storage for messages, addresses, counters, and memory work areas. The END directive is inserted after this area. It tells the assembler when to stop assembly. Adding an expression (BEGIN) in this directive is accepted convention for ASM programs. The reason for this [ref 2, pg. 2-16] does not apply for CP/M-85. The distributed LOAD.COM program produces .COM files that automatically execute from 0100H regardless of any END directive.

Using your editor, create a file named SEEK.ASM. The .ASM extension is required by ASM.COM to assemble the file. For PIE users, type PIE SEEK.ASM. Next, enter the comment lines for the program name, and the date. If you wish, add any general notes. Insert area separators as you feel the need. Take note, extra comments in ASM files don't consume valuable RAM as they do in BASIC. They are there for your needs. The assembler will simply ignore them. Now, tab over 2 times and enter the equate BDOS EQU 05H. The basis for this will be covered later. Did you notice the 0 prior to the 5? When using hexadecimal numbers, it is generally a good practice to precede them with a 0. This will avoid assembly time errors. An error would occur should the assembler encounter a number like E500H instead of 0E500H. Enter the directive ORG 0100H, and then insert the instruction BEGIN CALL OPNRD. Don't concern yourself over the meaning of this instruction as it will be covered in Part 2. Complete the file by entering the directive END BEGIN. The file should now look similar to Figure 4.

```

; 9/19/85                SEEK.ASM
;
; SEEK loads a text file into memory, counts the # of
; words, lines, and sentences in the file, the displays
; the results.
; ----- By: M.D Zapolski, Sr -----
;
;           BDOS    EQU    05H
;
;           ORG     0100H
BEGIN  CALL  OPNRD    ; Open file & read into RAM
;
;           END     BEGIN

```

Figure 4
SEEK.ASM — Stage 1

For the most part, a text file is treated by CP/M as a contiguous sequence of ASCII characters. Each "line" of the file is ended by the combination of a carriage return and line feed (i.e. 0DH followed by 0AH). The end of the file (EOF) is denoted by a CTRL-Z character (1AH). CP/M reads a file in 128 byte (80H) chunks called records. In CP/M these files can be as large as 8Mb! So, everything in our text file will be represented by a value whose ASCII equivalent is a printable character, printer/cursor control code, or EOF marker. These facts will be frequently used throughout this series.

To our benefit, CP/M was designed with some 40 interface functions [ref 3a, 4a, and 5a] that make ASM programming much easier, and the resultant programs "portable" (capable of use on different versions of CP/M). Their existence, allows the programmer to center the programming effort on the application at hand, rather than the details of data I/O. If we want to print a character on the screen, it is put in the E register, a CALL is made, and voila! It's on the screen. Effectively, these functions work in a similar manner. First, entry parameters are established — function # in the C register; and, if required, other registers are loaded with data. Second, a CALL is made to RAM address 0005H. Upon the return from this CALL, the function is executed (i.e. data moved, etc.). This series will use 4 of the 40 functions:

1. Console Output — function #2.
2. Open File — function #15.
3. Set DMA address — function #26.
4. Read Sequential — function #20.

That's it for Part 1. I recommend you review the noted references. Especially, the CP/M functions, ASM.COM, and get familiar with the use of your editor. Next month we'll learn how to load a disk file into RAM — making sure it's not too big to fit. See you then ...



Want New & Interesting Software? Check Out HUG Software



H-89 & H-150/160 SPEED DOUBLER

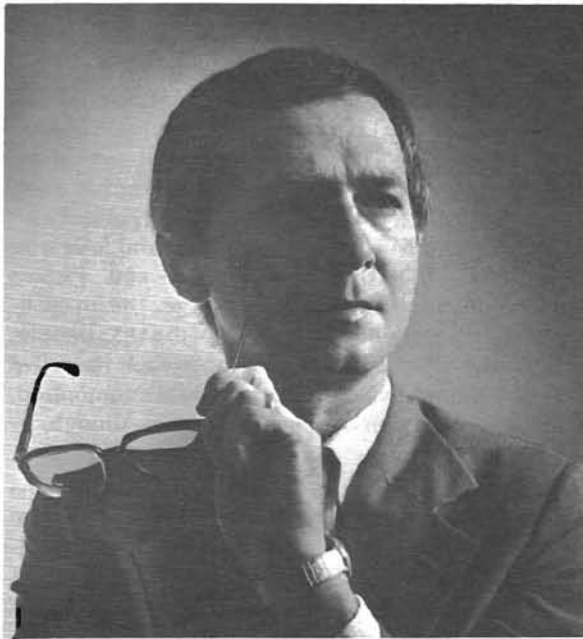
Stop Waiting! Increase your computer's speed NOW!
Easily installed. NO trace cuts or permanent mods.

H-89 Software select 2 or 4 MHz. Supports all H-89 CP/M and HDOS. 4 MHz Z80 included. Assembled and Tested \$34.95, Kit \$24.95, Bare board \$20

H-150/160 Runs at 6.67 or 4.77 MHz. Socketed crystal for easy speed increases. Hardware reset included FREE. Assembled and Tested \$34.95

Micronics Technology (904) 897-4257

449 Barbados Way, Niceville, FL 32578
VISA, MC, checks accepted. \$2 shipping.
Ask about Perfect Printer and Paycheck!



Mainstream Computing

Joseph Katz

103 South Edisto Avenue
Columbia, SC 29205

At The Fair

PCs, XTs, and ATs — mainstream computers — dominated this spring's COMDEX, the computer trade fair, even more than they had the last. Apple wasn't even there. Microsoft wasn't either, however, and Ashton-Tate had only a table at the Softsel booth — next to Fox & Geller, appropriately, who make dBASE add ons. The fair was smaller this spring than it was last spring. But IBM was big as life right across from the Zenith Data Systems booth, and Burroughs and AT&T and Tandy all had big booths, too. Scattered around the periphery were tiny little stalls at which one could order Brand-X compatibles and peripherals at tiny little prices. I found it easy to resist the temptation to buy one or the other. Maybe I will when disposable computers cost about as much as disposable pens and repairs are beside the point. Not now though.

What I sensed about this show was that now seems a time of consolidation, perhaps before a big surge forward. It's the feeling of watching a group of people who have recently realized they inhabit the same small world and are just getting into the final stages of working out the rules. If I'm right, prices in the XT world probably are about as low as they will get and just about everything in it is being perfected and polished. I don't think I saw anything really new. Everything seemed a refinement. IBM, for example, introduced nothing world-shaking. Exemplifying the situation are characteristics of its next XT line: half-height instead of full-height floppy diskette drives, availability of a 20MB Winchester, and the ability to take up to 640KB of RAM on a revised motherboard. That sounds like a scaled-down H/Z-158 to me. There's nothing at all wrong with that, but it's nothing new.

Also available for this refined XT is an optional 3-1/2" floppy diskette drive. You may have seen its diskettes for the Macintosh and a few other machines outside the mainstream. They hold 720KB, as opposed to the 320KB that can be put on a standard 5-1/4" diskette formatted double-sided with 9 tracks. IBM recently declared its irritation with makers of compatibles, because these

have been selling so well, and promised innovations that will make compatibility slower to achieve. This high-density diskette format, a good idea, presumably is one of those innovations. When it catches on, many of the present compatibles probably will become more or less incompatible. Don't you worry: IBM made its announcement on April 2; on April 28, Zenith announced that the disk controllers in its existing compatibles already supported the 3-1/2" format. So what's new?

One thing new was that this COMDEX turned into very much Zenith's show. News of those big government contracts you've been reading about had sunk in. When suppliers of software and aftermarket hardware made the connection between ZDS computers and my press badge (green was the color this season, perhaps some COMDEX humor), I became oddly attractive and I don't attribute it to my natty gray suit, striped shirt, and neatly-figured tie. At any rate, I was made offers that should keep you in even better touch with what's going on. Has this new-found popularity gone to my head? Of course.

Dilworth Speaks

It's appropriate, therefore, that Robert Dilworth, President of Zenith Data Systems, was the keynote speaker. Dilworth said many good things, but what I found most interesting was his implied recognition of HUG's importance to Zenith's success. "Zenith, as well as many of our competitors," he said, "has seen office automation develop from the bottom up on a piece meal basis. The decision makers are almost sneaking in the computers because they are becoming less and less expensive."

In a very real sense, of course, the "decision makers" are us: members of the Heath/Zenith Users' Group. For historical reasons, the core buyers and users of Heath/Zenith computers are a computer-sophisticated lot of people who have proven their ability to help place Zenith equipment. "Hey, Major Joe, does the darned thing really work?" "Well, Colonel Jim, I've been happy with every one I've owned." We validate not only the equipment but also the company, lending credibility in ways a salesman can't. I'm not sure a manufacturer could build so vast a

group of unpaid sponsors today. Even were that possible, the cost would be prohibitive. Zenith got us as part of Heath's good will and we may well be Zenith's single most valuable asset. The saying isn't "Nobody ever got fired for buying Zenith," so I'm glad that Dilworth evidently hasn't turned the news of those contracts into a local myth of corporate invincibility. In his maiden speech at HUGCON last, Dilworth gave me the impression of a sensible man not easily misled by reading his own publicity. That still seems so. Zenith Data Systems, therefore, looks to have a profitable future.

For those reasons, we should continue to have a unique kind of association with computer vendors — Heath and Zenith — that recognize the commercial value of healthy symbiosis. Hugs and kisses to us all.

PC Storyboard Glitters

At last spring's COMDEX, you could always count on finding an admiring crowd gathered in front of a "slide" show running on a computer at the IBM booth. Those shows were generated with PC Storyboard by, of course, IBM.

It's a splendid package of its kind. I have to admit that I appreciate PC Storyboard particularly because of the way Janet uses it to generate quick-and-easy instructional graphics for the University of South Carolina's College of Business Administration. What she does is produce image frames (the so-called "slides") on her XT there or her 150 here, arrange them into a sequence, and dump the sequence to a camera that attaches to the RGB (Red/Green/Blue) output of the computer. It in turn records each image on a frame of ordinary 35mm color film for processing into real slides, which can be shown with a slide projector to support classroom teaching, or on negative film for processing into real prints, which I can use for print publication.

(By the way, I was so impressed with that simplified description of that process after proofreading this column that I've just ordered the camera, too. It's a ComputerMate: \$2,500. I'm not sure I can afford to continue reading this column.)

There is competition to PC Storyboard, of course. Some packages are cheaper or shareware or in the public domain, and some of those are really pretty good. But none of them we've seen so far is in the same ring as PC Storyboard, although some of them are useful adjuncts to PC Storyboard in allowing image reduction, enlargement, and rotation — none of which can be done with PC Storyboard itself. Other packages can produce higher quality images — for commercial television and film, where higher quality is appropriate — but they are more expensive and usually require more hardware or hardware modifications. The quality of PC Storyboard's images is limited to the quality of the microcomputer's imaging capabilities, but it's much more than just acceptable for instructional purposes and in-house graphics. At \$275 (up from the former price of \$250) it's reasonably priced for what it does. If you know of something in the same price range with better capabilities, I really would be grateful to hear about it. So would Janet.

The PC Storyboard package is built around four separate programs that may be used to take a show completely from beginning sketch to final product: Picture Maker, Picture Taker, Story Editor, and Story Teller.

Picture Maker is a full-screen editing program used to create and edit each frame. It's about as good as a keyboard-operated program can be: there's no obvious way to use a graphics pad, not even a mouse, but the keypad keys can be used with fair effec-

tiveness for gross, intermediate, and fine movements. In addition, Picture Maker has a zoom feature that allows retouching any area of the screen, so English professors with patience can substitute patience for skill — although not for talent.

In fact, one area in which Picture Maker really shines is in its editing capabilities. You can alter foreground, background, drawing, fill, and shadow colors with the tap of a key, always able to preview the final result. If you don't like that result, moreover, erasing and undoing are equally easy. Picture Maker has good cut-and-paste capabilities too, which are particularly useful with the two image libraries (PM.LIB and PMALT.LIB) and four type faces (ROMAN.FAC, BOLD.FAC, THIN.FAC, and Picture Maker's own PM.FAC) supplied with the package. The image libraries have several screens, including a map of the U.S., people, vehicles, and various dingbats; the type faces each have five sizes. You can create your own type faces, if you are committed. It's much easier to create your own image libraries, though, by using Picture Taker.

Picture Taker is a useful adjunct to Picture Maker. It is a ram-resident program that captures a screen to a disk file so it can be processed with Picture Maker. Since PC Storyboard uses a modified BLOAD format, graphics images created by other programs that produce files with that format may be brought into Picture Maker by capture with Picture Taker. (Note, however, that Picture Maker cannot process text files from other programs.) But Picture Taker is useful because Picture Maker has no way to reduce, enlarge, or rotate an image. Such manipulations can be done with other programs and transported into Picture Maker, either directly or with Picture Taker. Do all necessary manipulations in the other program before loading the captured file into Picture Maker: whatever it saves to disk has a header unacceptable to other programs. In other words, files can be imported into Picture Maker, but not exported from it.

With Picture Taker, I've built libraries from samples that come with other programs. Snap the picture (by pressing SHIFT PRT SC when the desired image is on the screen), and it gets dumped to the disk file I've named when I ran Picture Taker. Then on to the next screen, which is dumped to the same file. And so on and on I go. When I'm finished, I have a file called something like DUMP.CAP (the "CAP" extension is automatically added by Picture Taker). What I do next is enter Picture Maker, call up DUMP.CAP as if it were a library, cut and paste each DUMP screen onto the main editing screen, edit out garbage like the cursor some programs won't conceal, and dump the edited screens to a new file with Picture Taker. Then, I rename the file to give it the extension "LIB," and I have a new library that can be loaded into Picture Maker and paged through with the F2 key. The procedure sounds gory, but it's not any big deal.

Story Editor is a sophisticated "programmer" for editing groups of files made by Picture Maker into a show. You can do a variety of effects — including dissolves, fades, and weaves — in a variety of directions, and they can be timed both for how long each image should be displayed and how long a wait should take place between images. Because images can be labeled (as, for example, "start," "waves," "end"), and because Story Editor provides a rudimentary programming language (including "goto" and "gosub"), you can create the effect of animation or do a reasonably useful programmed instructional sequence. For example, you can follow the explanation of a point with questions to test the viewer's understanding of it. An acceptable answer would branch into the next set of frames, while unacceptable answers might either repeat the explanation or go to a more detailed tutorial.

What Story Teller does is simple: it displays the show laid out in the script file generated with Story Editor. Since Story Teller, the script file, and the picture files all may be distributed under the PC Storyboard license (provided the diskette is labeled according to the instructions in the PC Storyboard manual), you can distribute the finished show as you like with no royalty payments to IBM.

Of course, neither PC Storyboard nor any other software will make me able to produce professional graphics, just as no word processing software will turn Janet into a professional writer. But if we keep a sense of proportion on our own talents and have a realistic understanding of our limitations in the alien field, the right tools help us according to our abilities. PC Storyboard lets me do good slides of a simple kind for use in class, while in her hands it is a valuable professional tool for media development.

Crosstalk And HyperACCESS Sing

An immediate effect of this COMDEX will result in modernization of my computer communications. A few months ago I had to jettison my old communications software because those programs simply weren't reliable any more. The only factors I know that have changed since I started using the MS-DOS version were the breakup of AT&T and our decision to use an alternative long-distance company. Whatever the reason, I couldn't count on getting my work to publishers by phone anymore and that was losing me money — which is not what computers are supposed to do.

Bless Mark Huth, who responded to my complaints on the CompuServe HUG SIG with a copy of PC-TALK III. I tried it for a while and thought it all right but not much more than all right. PC-TALK is a compiled BASIC program that I think takes up too much space for what it does. Bless Jim Buzskiewicz too, for sending HUGMCP (HUG Modem Communication Program). I consider it a real find: it's a reliable communications program that does all the fundamentals without wasting any space. Fast in, fast out, and apparently bomb proof, HUGMCP seems worth having if only as a backup to one of the more full-featured programs.

Several members of the SIG responded with recommendations of full-featured communications software, and three packages seemed to be dominant: Microstuf's Crosstalk XVI, Hilgraeve's HyperACCESS, and Microsoft's Access.

The Microsoft product is copy-protected, which means that Microsoft wants to protect its profits at my expense. One trouble with copy-protected software is that the protection devices sometimes malfunction. After all, they're there in the first place to protect the vendor, not the user. A few months ago stories surfaced about what happened to some folks when Microsoft Access malfunctioned: "The weeds of crime bear bitter fruit" appeared on their screen along with a threat to trash their hard disk. I enjoyed listening to "The Shadow" on the radio when I was a kid, and a few tapes are in my collection of classic radio shows, but still I am not amused. I am so unamused that I now do not trust other Microsoft products. I do have confidence in the professionalism of IBM, Heath, and Zenith, so I am not worried about whether similar time bombs are in their versions of MS-DOS or Microsoft products. But I sure ain't gonna put Microsoft Access on one of my hard disks.

So the recommendations worth pursuing really came down to Crosstalk XVI and HyperACCESS. Most reluctantly, I pursued them. My reluctance was based on my feeling that com-

munications software is essentially boring — for the user, that is. It's a tool, after all, similar in function to a remote control tuner on a television set. Great, it works: now, what's on?

Well, I was not right. These two software products are absolutely amazing. I had grown up on communications programs for CP/M and just stopped developing my knowledge when I found something that worked for me there. All I did later on was get the MS-DOS versions, which made me miss at least one entire stage of evolution in communications software.

Both Crosstalk XVI and HyperACCESS are major league products that can be used for major league applications. Each can be used with one disk drive or the biggest hard drive in your computer. Each can use "scripts" to automate communications, although HyperACCESS is more complete in containing features at \$149 that would require both Microstuf's Transporter at \$295 (which includes Crosstalk XVI) for unattended fully-automated communications (to transfer files after hours, for example, or unload a remote mailbox) and Microstuf's Remote at \$195 (for use as a host computer accessed by modem from remote computers). On the face of it, HyperACCESS looks like the bargain.

The Microstuf and Hilgraeve products are in the same general league, however, and it is the major league. I'm trying to learn both systems before I settle down to one — as I suppose I must. HyperACCESS comes with a tutorial disk, denominated "CAI" (Computer Assisted Instruction), to make learning easier. Crosstalk doesn't, but it's not hard to pick up the major features quickly without a tutorial disk. Part of the trouble I have with HyperACCESS is that it is completely menu-oriented: doing almost anything requires slogging from menu to menu, and I am both impatient and very easily confused. If you like menus, you'll probably prefer HyperACCESS to Crosstalk; if you swing the other way, Crosstalk ought to please you more than HyperACCESS.

If you own an H/Z-100, as well as a compatible, you might want to take into account the desirability of having the same communications software on both your machines. You can get a Z-100 version of HyperACCESS but not, so far as I know, of Crosstalk. (My standby HUGMCP is programmed to recognize which MS-DOS machine it's in, so one size fits all: run it on the Z-100 or the Z-100 PC or the Z-200 PC and it will figure out what it's supposed to do. If people keep writing software like that, columns like this would be much shorter.) If you have an Easy PC emulator on your Z-100, there is an Easy PC version of HyperACCESS, too — but, again, not of Crosstalk. (I understand from Dave Cheung, President of UCI Corporation, makers of the Easy PC, that he has patches to make HUGMCP work on the Easy PC. If you own HUGMCP and an Easy PC, write Dave for a copy and tell him Joe sent you.)

Norton's Tool Works

Just before we left for COMDEX, version 3.10 of The Norton Utilities arrived. For those who may not know, this package is a collection of utility programs that keep squirrels from nesting in a computer. There are eighteen of them and I'm not going to list them all here. The three programs for which I owe Peter Norton a drink any time we meet are the appropriately-named NU (which I assume would be "NU?" if MS-DOS didn't use the question mark as a wildcard) for disk snooping and diddling, SI for system information that includes an index of a computer's performance relative to the IBM PC (my 150 rates 1.0, my 158 at 8 MHz is 1.5, and my 241 is 6.6, facts I wield like a rapier at the drop of a gauntlet), and my real irreplaceable favorite DS. It sorts my hard disk direc-

ories so I don't go bonkers when looking for something. Every hacker likes doing utilities like these, and I'm no exception, so there are many public domain programs that do one or another of these jobs. The Norton Utilities has them all in one place, with comprehensible instructions, and the added feature of reliability. In other words, they have one small advantage over many free programs: they work.

Version 3.10 is a minor revision, so if you already have 3.0 you may not want to upgrade. Then again, you might. In addition to useful revisions of each program, there are two new programs that might save you from getting a little behind in your work: QU, for quick unerasing of some accidentally erased files (if QU can't succeed, you can go to the more powerful NU — which has never failed me); and UD, for unremoving a subdirectory you killed with the MS-DOS RMDIR. Here's what I think: the real reason you would want to bring back a removed subdirectory is that you next want to unerase the files in it, which means either that you are terribly green at working with MS-DOS subdirectories and didn't read the manual or that you have indulged in a Jovian orgy of carelessness. In either case, you might need version 3.10 of The Norton Utilities to salvage your dignity, as well as your files. Of course, if you don't already have version 3.00, you have the perfect excuse for getting 3.10: version 3.00 was a major revision.

Bad Chips

There is, or was, a flap in some circles about a bad DMA controller chip in the Z-150. The chip controls Direct Memory Access, hence the abbreviation. Rumors about the bad chip have circulated widely and, apparently, with the most satisfactory and unnecessary panic. Since every instance of the rumor I've seen has reference to someone else who first learned about the problem when trying to install Fastback, the hard disk backup program, on a 150, I decided to get a copy of the program to see what really was going on. Fifth Generation Systems makes Fastback; but, since there usually was a crowd around its COMDEX booth, I waited to get a copy until the fair was over and we were back home.

Argh! My 150 had the dreaded bad DMA chip! Had I been harboring a secret microphage in my bosom? I went on in that paranoid vein for a few moments, then I decided to actually read the message displayed by Fastback's installation program. I must say, it was a bit of a letdown. You can read it in Figure 1 and decide for yourself.

Unlike the IBM PC, the 8237 on the H/Z-150 is socketed. So I passed up the offer of the "DMA-Fix" board at \$40 and ordered an NEC 8237 chip from Software Wizardry for \$20. I really wouldn't have bothered had it not been for this column and the sense of responsibility I feel as a result. Until the replacement 8237 arrived, I used Fastback in the slow mode — as the installation program put it on the hard disk — with no trouble whatsoever. Compared with the way Fastback works on the 158 in the fast mode (there's no 8237 chip there to worry about), the slow mode is indeed slow. Without that comparison, Fastback in the slow mode seems fast enough to me. Age makes one savor every remaining moment. When the 8237 came a couple of days ago, I reinstalled Fastback and it now works fine in the fast mode. Of course, there was a chance of getting a similarly-bad replacement chip, according to the message, and of course I hadn't had any trouble to start with — at least so far as I knew. Nevertheless, the patient survived the operation. I don't know if the operation was really essential, but I do think the flap isn't. Maybe I've used computers long enough to have become acclimated to the con-

ditions of living with relatively innocuous bugs. As long as they don't bite me, I guess I'm really quite tolerant. Squirrels, of course, are another matter entirely.

Fastback itself seems to work nicely. Not only does it back up quickly, but also it restores quickly — by individual files, if you'd like, or by groupings if you have to restore an entire subdirectory or even an entire disk. I have a special fondness for backup programs that provide reliable restoration too: some of them, in my experience, don't. That kind of misses the point, I think. Fastback not only restores files reliably, but also makes restoring files easy. I like all that.

There are, however, things I don't like about Fastback. One is that it gains its remarkable speed in part by taking over the process of formatting, as well as of backing up. There's nothing inherently wrong with the procedure except that the backup disk is not readable except with FRESTORE — the Fastback restore program. I also don't like Fastback being copy-protected. It's the key disk scheme, which requires you to have the distribution disk in Drive A when you start to run the program, and I hate having to search for a distribution disk when I want to use a program. Since I also hate backing up my hard disks, anything that is a burden makes me postpone the job. I already postpone it too long too often. I also hate copy-protected software, and I think it bizarre that a backup program is copy-protected. Don't these guys recognize the importance of having backups? (There is a way to get an un-copy-protected version after you buy the copy-protected version of Fastback, but you have to sign things and stuff.) The third thing I don't like is the price: \$179 seems awfully pricey to me when the cost of inexpensive streaming tape backup units is low and seems to be getting lower. Nevertheless, Fastback works.

After The Fair

It's only about three hours or so by car from downtown Atlanta to our house in Columbia; but at 3:30 A.M. of the morning before our last day at the show, the Downtown Holiday Inn broadcast a fire alarm. We snapped out of sleep and ran for the street. Okay, "snapped" and "ran" were boasts: we were exhausted after a long day of work, so we more or less staggered half dead downstairs and onto the sidewalk. Just after we got there, the public address system said there was no fire, only a false alarm. So we stumbled back to bed from our questionable haven and then, just after we had drifted off to sleep again, the blasted hotel's public address system — an infernal machine, in my opinion — awakened us AGAIN. This time it explained that the alarm had been "a computer malfunction." I worked part of the day in a stupor, stumbling from exhibit to exhibit in the Georgia World Congress Center examining computers that didn't malfunction, then gave up and pointed the car homewards. Janet drove the rest of the way. Women are stronger than men.

Products

PC Storyboard (Version 1.0) IBM Entry Systems Division Box 1328 Boca Raton, FL 33432	\$275
The Norton Utilities (Version 3.1) Peter Norton 2210 Wilshire Blvd., #186 Santa Monica, CA 90403 213/826-8032	\$ 99

HyperACCESS (Version 3.10) \$149
Hilgraeve, Inc.
P.O. Box 941
Monroe, MI 48161
313/243-0526

Transporter (Version 1.4)
and Crosstalk XVI (Version 3.61) \$295
Remote (Version 1.3) \$195
Microstuf, Inc.
1000 Holcomb Woods Parkway
Roswell, GA 30076

FastBack (Version 5.03) \$179
Fifth Generation Systems
7942 Picardy Avenue
Suite B-350
Baton Rouge, LA 70809
800/228-6127

Doctor Katz's Mailbag

Screen Dumps On The Z-150

Dear Mr. Katz,

Your article in the May REMark magazine was delightful in every way. Your knowledge, wit and clear concise style make an impressive combination. You really have my situation, and my frustrations, clearly identified. Ever since I bought my Z-150, I have had continual hardware and software problems caused by that "1% incompatibility" the salesman mentioned.

Your column is a priceless addition to REMark.

I would like to see an article about the graphics function of the Z-150 compared to IBM. I have had no end of problems getting "compatible" programs to print out on my printer. Supposedly, my Star SD-15 is IBM compatible. However, I have not been able to get them to work together. Perhaps I am not IBM compatible.

So far, none of the manufacturers has been able to help, except to tell me that Zenith put the equivalent of GRAPHICS.COM in ROM. Therefore, some programs cannot find it, because they are looking in DOS.

Because of this problem with "DRAW-IT" (Paperback Software), I have been reluctant to buy any software unless it is guaranteed, in writing, to work on my Zenith . . .

My usage is 100% business. I am a consultant and teacher of customized technical courses. My professional experience is engineering, marketing, and teaching.

Sincerely,

Jon S. Wilson
San Juan Capistrano, CA

Thanks for the kind words. I appreciate them.

Well, look on the bright side: you may have been given enough free hogwash to let you open a swine laundry.

DRAWIT (a drawing program from Paperback Software) has no printing functions whatsoever, so it doesn't "look" at DOS or ROM or anywhere else — except maybe at you while it waits for you to issue a command for something it can do. The way you print a DRAWIT screen is to dump it to your printer. It's a pretty primitive way to print when you think about it.

The method is called a "print screen" or "screen dump." The way a screen dump works on a compatible — any compatible — is through a BIOS interrupt. "BIOS" is the "Basic Input Output System"; an "interrupt" is a kind of trap that waits for a well-defined trigger, at which point it produces an event. Generally speaking, if the BIOS interrupt, the event, and the circumstances that trigger it are all the same on two different machines, the computers are to that extent compatible. It makes sense, I think.

Well, the BIOS interrupt that triggers the "print screen" event on the IBM and Heath/Zenith's compatibles is the same: INT 5. It is triggered by the same circumstances: receiving a keyboard scan code of 5500, which you send when you press "SHIFT PRT SC" (hold down a SHIFT key and the PRT SC key simultaneously). And the result is the same: whatever is on the screen gets dumped to the printer.

Figure 1
Fastback's Bad DMA Diagnostic Message

```
***** GREETINGS *****
The DMA controller chip (8237, on the motherboard) failed. There is a documented bug in the 8237 DMA controller which affects a small percentage of IBM-PCs and compatibles [sic]. The problem is not specific to any brand of computer. However, the 8237 manufactured by Advanced Micro Devices seems to be especially bothersome. The bug only occurs when 2 or more channels of DMA are used at the same time (the 8237 is a 4-channel device). Most high-speed tape drives and some network interface cards will be affected. Regrettably, so is FASTBACK(tm). This copy of FASTBACK(tm) is being temporarily configured to operate in a slower, safer mode until the hardware is repaired. This is the only diagnostic we know which will find the problem, since other diagnostics test one channel at a time. You have the following options: 1) Purchase a 'DMA-FIX' board from us ($40.00, including Fed-Ex shipping). This little circuit card fits underneath your 8088/8086 CPU and corrects the timing problem. It does NOT slow your computer. Also, consider replacing your CPU with an NEC V-20 or V-30 CPU at the same time ($30.00 extra, comes pre-installed on the 'DMA-FIX' board). It's plug compatible [sic] with the 8088/8086 and 30%-50% faster (about the same as an 80188/80186) with no increase in clock speed. We've used them in-house since 7/85 with no problems. Call 1-800-228-6127 or 504-767-0075 for ordering information. 2) If your computer has an AMD 8237 and it is in a socket, try replacing it with the Intel part. Make sure the speed rating matches your computer. Our experience shows a success rate of about 25%-50%. If you have an IBM-PC, don't bother looking; the chip is soldered in. UNDER NO CIRCUMSTANCES SHOULD YOU DESOLDER A CHIP FROM A MULTI-LAYER BOARD!! 3) Continue to operate FASTBACK(tm) in the slow mode and hope the problem doesn't affect your other programs. It probably has already, but the blame was put somewhere else. The symptoms vary greatly, since the DMA controller will (maybe very infrequently) put data in the wrong place. Sometimes it even writes over DOS, causing massive data loss!! 4) Kludge your motherboard (specifics are available from AMD). Very nasty, will void warranties and service contracts. 5) Contact your manufacturer. We will assist you as much as possible. If the computer is under warranty, your chances are better. In case you didn't write it down, our phone number is 800-228-6127 or 504-767-0075.
```

Joseph Katz gets just too much mail to answer each letter individually. From time to time we'll try to publish a selection of the most useful letters here. If you don't want your letter published, be sure to mark it "not for publication."

But since DRAWIT doesn't do anything in all this, or even know it's going on, why in the world should it care about any of it? In fact, why should you care?

Normally, you wouldn't — on a Heath/Zenith compatible. When you're at the DOS command line, for example, just do the print screen and watch your printer do its stuff. The only reason why you have to care at all is that DRAWIT uses graphics characters made from what IBM defined as "Extended ASCII": not the normal alphanumeric characters you and I, and printers in their standard mode, can recognize. Your Star is "IBM compatible" in the sense that it can print the Extended ASCII character set.

Even so, to do a graphics screen dump (which is what DRAWIT requires) — no matter whether you own an IBM or Heath/Zenith compatible — you have to first run a printer driver that sits in RAM and does some translating. It has to translate the Extended ASCII codes into whatever codes your printer requires to reproduce the screen characters on paper. IBM supplies one such driver, GRAPHICS.COM, which has switches for three kinds of printers sold by IBM. Heath/Zenith supply drivers for several different printers from various makers. In either case, you need to run a driver that works with your printer and you need to run it before you do the screen dump — which means, in your case, before you run DRAWIT.

Heath/Zenith's printer drivers all have names that begin with the prefix "PSC" (for "print screen") and have the extension "COM." Look for them on Disk II of your MS-DOS distribution disks. You want PSCMX80.COM. Copy it to your hard disk (or working diskette) and run it before you run DRAWIT. You only run the driver once each session: don't get carried away.

By the way, you've bumped into one of those things I was talking about in the first "Mainstream Computing," when I said that Heath/Zenith compatibles sometimes do the same job as IBM's but in different ways.

Your principle about making sure something will work before you spend your money is pretty good, though. I certainly won't guarantee anything except my own good will and reasonable honesty, but I'll try to help here and through the column itself.

As a matter of fact, here's an experiment that seems worth trying. If you — or anyone else in our community — contemplates buying software or hardware and hasn't had the chance to test it, send me a note saying so and ask me to review it. If it looks like something I can handle within my own limitations and policies and interests, I'll see if the vendor is interested in cooperating. In short, I'll try to put it in my review queue. (Ah, I love it when I talk computer!) If the vendor chooses not to cooperate, you may have learned something useful in making your decision. Be sure that I will have learned it. At any rate, call it "an experiment in custom reviewing." Seems worth trying, as long as the burden doesn't become onerous.

Joseph Katz



Heath/Zenith Related Products

Jim Buszkiewicz
HUG Software Developer

Spectro Products Inc. of North Haven, Connecticut announced the release of ZIOMATE-100 in kit form. ZIOMATE is a unique product which combines, on one S-100 board, the functions of eight channel, programmable-gain, multiplexed 12 bit analog-to-digital inputs, two 10 bit digital-to-analog outputs, a 16 line TTL compatible output port, and a 24 line TTL compatible input port. It opens the Zenith Data Systems H/Z-100 family of computers to real-time operations that include process control, instrument control, data logging, data reduction, and automated analysis. In kit form ZIOMATE-100 sells for \$975 and more information can be obtained from Spectro Products Inc., 385 State Street, North Haven, CT 06473, (203) 281-0122.

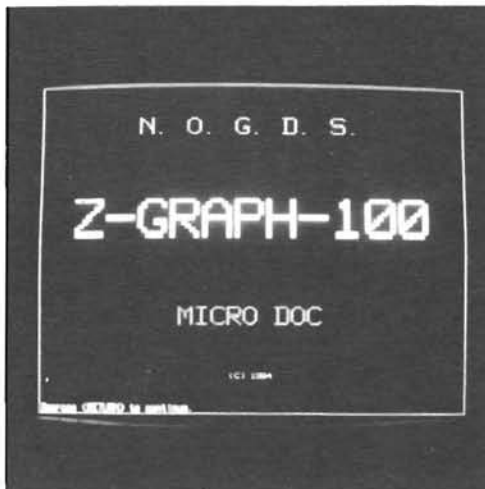
Generic Computer Products Inc. of Marquette, Michigan announces the "LAST .SCALL TO HDOS USERS". Special price discounts are being allowed on all HDOS software currently in stock. All software items are available in either Heath hard-sector or soft-sector formats. Products still available include: CARD-MASTER, CATALOG-MASTER, DECISION-AID, FOOTBALL, FOOTBALL-PICKS, GAME-PAC I, MAIL-IT II, MICRO-CABINET, SPACE-WARRIOR, WORLD-MAP, and HDOS Game Combination Special. For more information, contact Generic Computer Products Inc., P.O. Box 790, Marquette, MI 49855, (906) 249-9801.



Z-GRAPH-100

Graphics Package

Review



Paul W. Simmons
6409 Glenbard Road
Burke, VA 22015

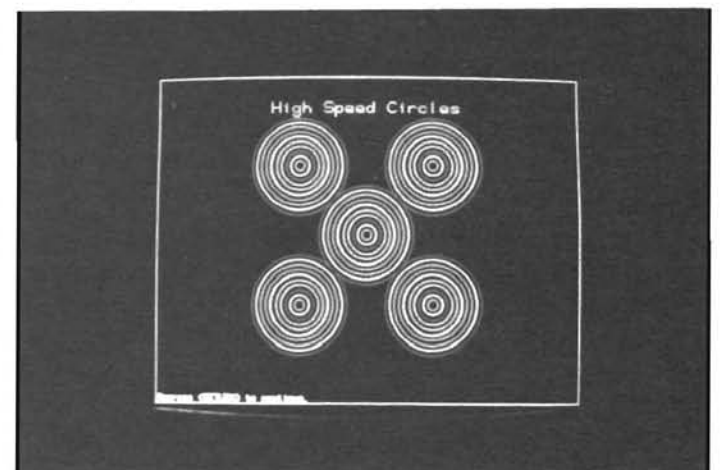
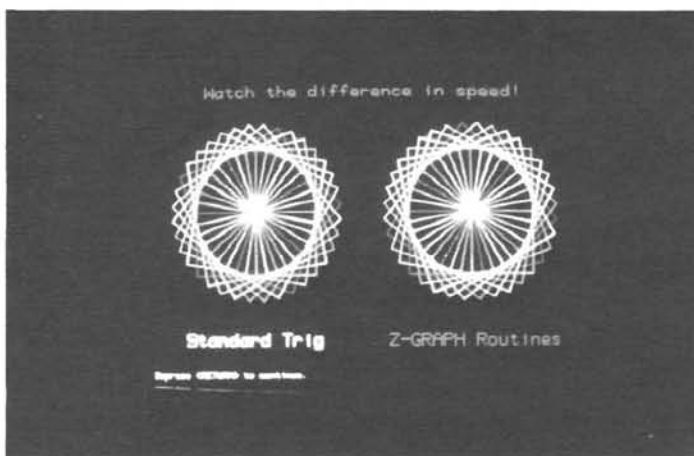
Z-GRAPH-100, by New Orleans General Data Services (NOGDS) and Micro Doc, is a library of procedures which provide high resolution graphics to a whole passel of compilers on the Z-100 computer. It's a super system for the experienced programmer or the DP professional; however, its documentation prohibits it from being readily useful to the novice. If you have a Pascal, C, FORTRAN, COBOL or ZBASIC compiler or are an 8088 Assembly Language programmer, read on; Z-GRAPH-100 may be for you.

In this article there will be no attempt to compare Z-GRAPH-100 to other products on the market. That comparison will be the subject of another article. We will discuss the capabilities of Z-GRAPH-100 and its usefulness as a graphics library.

After purchasing my Z-100, I began developing my own set of graphic procedures using Pascal, MASM, and the TM-100 Technical Reference Manuals for guidance. I developed a series of basic level graphic routines such as POINT, MOVE, LINE, CIRCLE, etc. These were interesting and educational to write, but not a very productive use of time. It's the kind of thing that is good for the soul, but not recommended for those who have a job to get done.

Because of this exercise, I became cognizant of some of the graphic power of the Z-100 hardware and painfully aware of my inability to efficiently use it. Your Z-100 has great graphics potential bottled up inside it that is totally ignored by most compilers, except Microsoft's ZBASIC Compiler. ZBASIC taps only a part of this potential. Z-GRAPH-100, on the other hand, harnesses it, and puts it into the hands of the graphics programmer.

One of the most impressive features of Z-GRAPH-100 is its speed. This speed is realized by implementing all code in assembly language and using a special Z-100 bit-mapped graphics mode. Because of this, the LINE, CIRCLE, and PAINT functions and other graphics functions are much faster than similar procedures implemented using the normal display mode. The latter is optimized for text processing whereas the Z-GRAPH-100 mode is optimized for high resolution graphics. It would be interesting to see how fast Z-GRAPH-100 would perform using the newly released Heath, HA-108, 8 MHz upgrade kit to the Z-100 computer. As it is, it easily outperforms my own graphic routines: routines that are written in 8088 assembler and optimized for speed. The difference is, I stayed with the normal display mode; they didn't.



Along with the speed and the special graphics mode (the good news) come several problems (the bad news). While in this graphics mode, normal input from the keyboard is not properly displayed on the screen. The Z-GRAPH-100 folks have attempted to mitigate this problem by supplying a KEYIN procedure and several display routines. However, the KEYIN procedure is very primitive. It retrieves one character at a time from the keyboard buffer as an integer value and the programmer must convert it to the proper ASCII character. There are no procedures to input strings of characters, integer numbers, or real numbers. It is left to the programmer to develop these routines by repeatedly calling KEYIN. To some programmers, this may be a problem or certainly a nuisance. To me it's not, since I normally verify user input one character at a time anyway. By doing this, errors can be immediately identified and special keys such as HELP, BREAK, BACKSPACE, etc. may be detected and processed.

Another problem with the special graphics mode occurs if your software aborts, as mine frequently does. You are left in the special mode and subsequent keyboard entries are not displayed on the screen correctly. Sometimes, depending on where the cursor was positioned when the program died, keyboard entries are not displayed at all. In order to get back to a normally functioning screen, you need to reboot. What an annoyance, especially since I have a system log program which executes every time the computer boots! To alleviate this problem, I wrote a short Z-GRAPH-100 program called FIX, which correctly exits the graphic mode. Now, whenever I abort in the middle of something, I immediately run FIX to restore the screen.

On the positive side, one of the very attractive features of the Z-GRAPH-100 system is its ability to link to a virtual plethora (biggest word in my vocabulary) of language compilers. At this writing it includes: ZBASIC, MS COBOL, MS FORTRAN, MS PASCAL, MASM, C86, C186, and De Smet C. It even has a universal graphic device driver which can be used by any language including Turbo Pascal. Now that's what I call flexibility! Unfortunately, I did not have the opportunity to test the universal device driver.

To multilingual programmers Z-GRAPH-100 is great. You only need to learn one graphics interface for all languages. From a monetary point of view, only one purchase of Z-GRAPH-100 is needed to use graphic capabilities with all your languages. The boys in the back room at NOGDS and Micro Doc must have fun keeping up with all the compiler releases and versions thereof. Personally, I have five compilers and Z-GRAPH-100 supports all of them. It's a programmer's dream: five languages and only one graphics library to worry about.

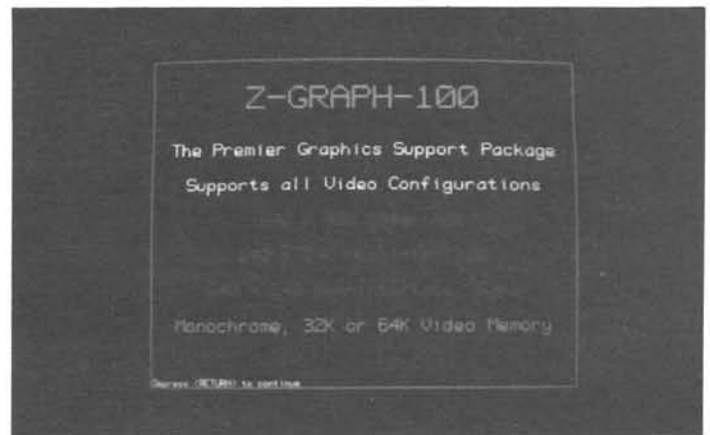
For you Tektronix fans, Z-GRAPH-100 tries to mimic the Tektronix command syntax wherever practical. This, of course, will ease conversion of existing software to the Z-100. Don't get too excited though; only the bare bones of the Tektronix routines are available and some of those have been modified to take advantage of unique Z-100 features. However, those familiar with Tektronix will have little trouble writing code using Z-GRAPH-100.

Personally, I would not expect or even desire complete Tektronix compatibility. I don't own a Tektronix. What I need is a set of routines that take advantage of the Z-100's powerful graphics capabilities. If I wanted complete compatibility with Tektronix, I'd buy Tektronix (big bucks). What's important here is that the Tektronix programming philosophy remains intact.

Another nice feature of Z-GRAPH-100 is that it takes advantage of all possible combinations of display monitors and screen memory configurations. The parameters defining these hardware characteristics are defined at run time. Consequently, the same program can run on any Z-100 (a nice feature for those programmers who are marketing their software). By the way, you are permitted, without charge, to link Z-GRAPH-100 relocatables into software you market.

Depending upon the availability of screen memory, Z-GRAPH-100 may be initialized in one of three possible modes:

- Interlace-sync and Video mode 640 X 480 (horiz. pixels X vertical pixels)
- Non-Interlace mode 640 X 240
- Interlace-sync mode 640 X 240



Interlace and sync describe how the CRT displays the picture generated by your Z-100 computer. The normal Z-100 display is non-interlace mode with a resolution of 640 pixels horizontally and 240 pixels vertically. If you have upgraded your video memory from 32k chips to 64k chips and made the necessary jumper connections, then the interlace-sync and video mode may be used with a resolution of 640 by 480. Regardless of the mode you select, the programmer always uses 480 as the vertical screen resolution and Z-GRAPH-100 adjusts automatically to what is physically available.

With the preceding serving as general background to Z-GRAPH-100, I will embark on a description of the available procedures. Because of a lack of imagination on my part, I will use the organization set out in the Z-GRAPH-100 documentation to discuss groups of routines.

The first group is made up of the Configuration Procedures. They include an assortment of routines to set parameters, manipulate the CRT, and enter or exit the special graphics mode. All programs using Z-GRAPH-100 must first make a call to INIT to initialize the special graphics mode and to set up a number of internal variables to keep the Z-GRAPH-100 routines synchronized. At this point, you define the initial foreground and background colors, the origin of the screen coordinate system and availability of a color monitor. If color memory is not available, foreground colors are mapped to green and background colors to black. To exit the special graphics mode, call ZGDONE. This will return the screen to the normal text mode. ZGDONE must be the last Z-GRAPH-100 procedure called. As mentioned earlier, if a job aborts or is interrupted before a call to ZGDONE has been made, subsequent keyboard entries and screen displays will be difficult, if not impossible, to read. The Z-GRAPH-100 people should provide a utility program to return the screen to the normal text mode after an abort.

Other Configuration Procedures include the ability to change the foreground and background colors, turn on or off the video display and set the line style. There are sixteen predefined line styles available and the user has the ability to define his own style. Setting my own style has always appealed to me.

The ability to turn the CRT off is useful when updates to the screen are distracting or confusing to the user. The graphic memory, of course, is not affected, and you have a choice of colors to which you may set the screen while the display is off. A procedure to write to the CRT controller is also provided for you Z-100 hardware experts. A discussion of this is beyond the scope of this article (and beyond me, as well).

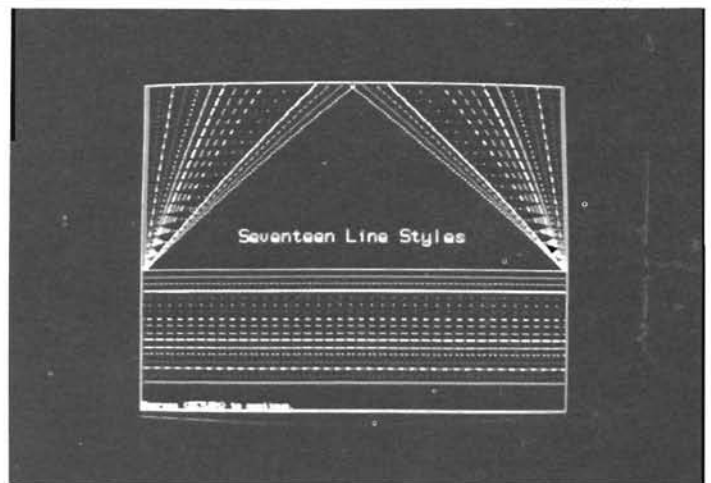
The next set of procedures to be discussed is labeled Display Control. Paramount to this set is the WINDOW procedure. It defines the minimum and maximum x, y coordinates used in your program. These coordinates are termed the "world" coordinates, and are the coordinates the programmer uses. They are in feet, inches, miles or whatever and may range from +/- 32k. Z-GRAPH-100 maps these world coordinates into screen pixels. The programmer only references screen pixels (640 horizontal X 480 vertical) directly in the VIEW procedure, to be discussed shortly. I missed this point on my first go around with Z-GRAPH-100 and consequently, spent several hours thoroughly confused. I feel somewhat betrayed by the documentation, since it frequently refers to pixels. I assumed these references were to screen pixels. Nothing could be further from the truth. With the exception of the VIEW procedure, when the documentation refers to pixels, it is referring to world coordinates. These world coordinates are mapped into screen pixels for you by the software. If you wish to work in screen pixels, you must define the world coordinate system to match the screen coordinate system (640 X 480). This, by the way, is the default condition.

Following close behind the WINDOW procedure in importance is the VIEW procedure. It defines in screen pixel coordinates on which portion of the screen the subsequent graphics will be displayed. VIEW enables the programmer to define multiple views (viewports) on the screen at the same time. Once a view has been established, all subsequent graphics are automatically scaled to fit within the view limits. Besides simultaneously displaying multiple plots on the screen, VIEW is useful in displaying menus along with graphics.

Display Control routines also allow you to specify absolute or relative plotting and to retrieve the location in world coordinates of the last point plotted. This is particularly useful when text is plotted, random numbers are used to drive the graphics, or the user is controlling the plotting.

The Plotting, Drawing and Circle routines are the first group of Z-GRAPH-100 procedures that actually display something on the screen. Points, lines, circles and arcs may be drawn. The foreground color is used and the figure is drawn in any one of sixteen standard styles. The color and style may be changed at will and the programmer may specify his own unique style. In addition, a procedure is provided to check the color of any pixel specified by world coordinates.

Also included in this group is a procedure to convert rapidly from polar to rectangular coordinates. The demo program provided with the library shows how much faster the conversion is using the Z-GRAPH-100 routines. It's impressive. This is good news for you polar coordinate gurus.



The final and most interesting procedures in this grouping are the Toggle routines. These procedures allow the programmer to plot points, lines, circles and arcs without destroying what was previously on the screen. This is handy for displaying moving objects, such as a cursor or for animation.

The Area Fill grouping of procedures is used to fill or paint closed polygons or figures, bars, rectangles, circles, etc. with a pattern defined by the PATTERN procedure. The resulting color of the filled area is based upon the foreground and background color in effect when filling takes place. A multitude of patterns are available. One can specify twelve different patterns for each of four directions of shading lines, all of which may be used in any combination with one another. To further distinguish filled or painted areas, the programmer may change foreground and background colors at will.

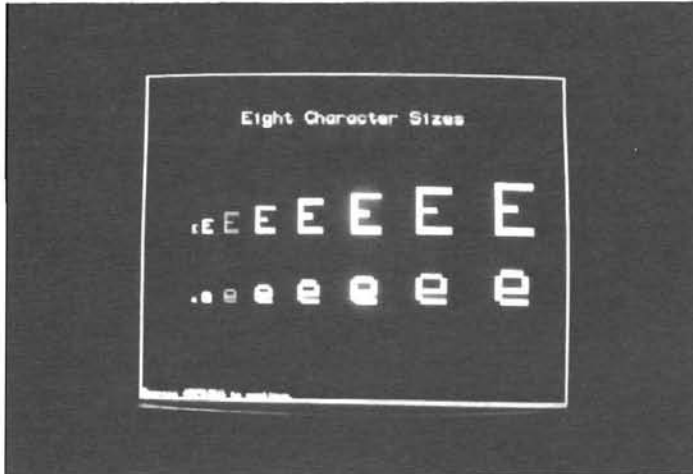
By filling with every other vertical line, the programmer blends the foreground and background colors, and effectively expands the number of available colors on the screen.

The next to last grouping of procedures involves the Graphics Text and Symbols. These routines allow you to plot text and symbols on the screen along with the graphics. You can also define your own character fonts or use the Z-GRAPH-100 standard fonts. Personally, I like the Z-GRAPH-100 standard fonts; however, for you individualists, Z-GRAPH-100 provides you the ability to use as many fonts as you care to define. You can change fonts at will from within your program. Oh yes, the Greek alphabet is provided as part of the standard font.

Once the fonts are defined, the MOVE procedure is used to define in world coordinates the location of the first character of text. The GTYPE procedure is then used to display a string of characters on the screen. The CTYPE procedure is used to concatenate text on the same line to a string of text previously plotted, without computing its beginning position. Finally, text may be plotted in eight different sizes and four different orientations.

I feel the character plotting routines are weak. Specifically, integer and real numbers must be converted to character strings before they can be displayed. A more serious problem arises from character strings being positioned using world coordinates. There is nothing inherently wrong with this, except Z-GRAPH-100 neglected to provide the programmer with the size of the character in world coordinates. Consequently, if the programmer wishes to display a series of lines of text, one below the other on the screen, there is no Z-GRAPH-100 routine to call

to determine where to move to in world coordinates to begin the second and subsequent lines of text. This problem is further complicated since text characters are bit mapped and not scaled. Consequently, when the window and view coordinates change, the numeric value in the horizontal and vertical dimensions of a character in world coordinates changes, but the physical size of the character on the screen remains the same. To alleviate this problem, I wrote a text size routine which returns the size of a text character in world coordinates based upon the GTYPE procedure character size value and the WINDOW and VIEW parameter values. A routine similar to this one should be provided by the Z-GRAPH-100 folks as part of the library.



The final group of Z-GRAPH-100 routines is made up of the Non-Graphic Support Procedures. This includes a random number generator, a pause procedure, a keyboard input procedure (discussed earlier), a keyboard input status procedure (which enables you to determine if a key has been pressed without actually doing a read), a read or write to an AUX port, and a general error check procedure.

This wraps up the Z-GRAPH-100's capabilities review. In preparation for this article, I tested all the Z-GRAPH-100 routines, except the read/write to AUX port and the write to the CRT controller. All the routines functioned correctly when used properly; that is, used in accordance with the documentation. I was able to produce errors with the circle procedure that the general error check routine was not able to catch. These errors resulted from trying to plot a circle that fell partially outside the currently defined view and window. Because of this problem, I wrote my own circle testing routine to verify that I was requesting a legitimate circle, prior to calling the Z-GRAPH-100 circle routine.

In general, I was pleased, even impressed, with Z-GRAPH-100 routines, their capabilities, and their performance. The routines work as advertised. With the exceptions I have noted, the system is complete.

My biggest gripe with the system is the documentation. Sound familiar? It should. historically, software documentation has been notoriously poor, regardless of the application, be it graphics, database, word processing, whatever. Don't get me wrong; the documentation is adequate for the experienced Tektronix programmer and for the graphics programmer, in general. It assumes you already know how and why to use the procedure. There are very few examples and some of the explanations are quite short. This is a shame since the system is sound and performs excellently.

During my use of Z-GRAPH-100 I only needed support on one occasion. The problem I was having turned out to be one not associated with Z-GRAPH-100 at all. I am too embarrassed to admit what it was; however, Fred Pospeschil of Micro Doc quickly and courteously provided assistance and resolved my problem.

With the addition of a graphics tutorial based upon the Z-GRAPH-100 routines and an improved reference manual, Z-GRAPH-100 would be an excellent graphics package for the novice and the experienced graphics programmer. Currently, I can whole-heartedly recommend Z-GRAPH-100 to only experienced programmers, but to them, at a price of \$89, it's a bargain.



H-89 20 MEGA BYTE WINCHESTER! TIRED of swapping floppies and limited space???

- 20 meg or two logical 10 meg drives.
- Uses left side memory slot.
- Completely integrated Heath BIOS.
- Western Digital Controller, Seagate ST225 Drive

INTRODUCTORY PRICE - Only \$795.
Discounts available.

ORDER NOW by writing or calling:
Micronics Technology (904) 897-4257/4218
449 Barbados Way Niceville, FL 32578
Checks, VISA, MC. Add \$20 shipping.

JOIN THE PROFESSIONALS WRITING GRAPHICS PROGRAMS

on the
Z-100 _ Z-150 _ IBM PC
with
FLEXI-GRAPH

Libraries based on TEKTRONIX and CORE graphics standards. Over 40 routines provide a wide variety of high speed graphics capabilities.

Callable From:
MS-FORTRAN — MS-PASCAL
Many C Compilers

(Includes source code for high level language interface routines)

FLEXI-GRAPH

Only \$99 (Plus \$5 for shipping & Handling)
(Discount available for registered Z-GRAPH-100 Owners)

(Extensive Demo Disk—\$3)

SEE US AT HUGCON '86—BOOTH No. 25

NEW ORLEANS GENERAL DATA SERVICES, INC.

7230 Chadbourne Drive
New Orleans, LA 70126
(504) 241-9495



A Program For Upgrading The H-89 With A 16-Bit Single-Board Computer

Phillip L. Emerson

*Dept. of Psychology
Cleveland State University
Cleveland, OH 44115*

Several suppliers offer add-on options for the H-89, such as extended-format disk controllers and memory extensions. After considering some of the add-ons that are available separately, an H-89 owner might decide to go with a single-board 16-bit computer that combines some of the separately available add-ons, with a 16-bit processor and memory. This strategy opens the door to the use of the burgeoning body of software for the popular 8088/86 machines, in addition to the continued use of H-89 software. The 16-bit board I chose was the Slicer, but others might work approximately as well. Such boards are available with almost everything that is needed to get well started. The Slicer has an 8 MHz 80186 CPU, 256 bytes of memory, two serial ports, and a floppy disk controller. The floppy controller handles up to four drives in optional mixes of 5" or 8" drives with any of the popular options of sides, densities, and formats. Available operating systems are MS-DOS and CP/M-86. In addition, this single board computer has a host interface and system software for a hard disk.

The General Setup

I first considered mounting the 16-bit board inside the H-89 enclosure, but noted that there would be little room for future expansion. Fortunately, appropriate box enclosures and power supplies are now reasonably inexpensive, and many suppliers include all the needed hardware and connectors. I bought a hefty PC-XT type box and power supply by mail order from advertisers in Byte magazine. I found everything to fit perfectly, and most all the component-mounting problems to be anticipated. The power supply came with four disk drive power cables with installed connectors ready to plug into disk drives, and two other cables for powering circuit boards. With only the single-board computer, the power supply, and two Teac half-height drives in the box presently, there is a lot of room left for expansion boards and additional disk drives.

A serial cable connects the H-89 modem port to the serial terminal port of the 16-bit board. Thus, the H-89 looks like a terminal to the 16-bit host, and the host looks like a modem-accessible mainframe to the H-89. The accompanying program implements a few of the many possible functions of such an arrangement. It is an H-89 program written to interact directly with the operating system (MS-DOS or CP/M-86) command processor of the host, at 9600 baud. Thus, it requires no special companion program for the host.

The most elementary function of the program makes the H-89 act like an ordinary terminal. What is typed on the keyboard is sent to the host, and what the host sends back is printed on the screen of the H-89. Beyond that, there are four elementary disk operations: (T) Send an H-89 ASCII disk file to the host; (R) Receive into an H-89 ASCII disk file, from the host; (U) Send (upload) a binary file to the host, translating to hex; and (D) Receive (download) a hex file from the host, translating from hex to binary.

Advantages

These simple operations take care of quite a number of the functions that one might envision for such a system. There are several advantages of the interconnection, rather than running two independent systems. First, I did not have to buy a terminal for the 16-bit system. Second, I can easily archive hundreds of old hard-sector SSSD H-89 disk files on the higher-density 800-Kb Teac drives. Third, I can use my old familiar H-89 screen editor to enter programs and data for running on the host. Fourth, I can transfer data between host MS-DOS and host CP/M-86 disks by way of the H-89. Fifth, a printer is readily shared and can be connected to either computer — or two printers can be operated simultaneously by the two systems. The available commands do not include H-89 OS functions, such as listing an H-89 disk directory or resetting an H-89 disk drive. They are not needed because

one can freely exit this program with a Control-C, perform such tasks, and then reload to resume ongoing operations with the host.

Disadvantages

Some disadvantages of the interconnection, rather than independent systems are as follows. Two people can not be doing two different things on the different systems at the same time, and the H-89 modem port is not readily available for other uses. Though independent dual processing is possible with the present program, it tends to be practical only when some long-running background task is available for the 16-bit host, with little need for intervention. The batch-processing procedures of MS-DOS and CP/M-86 can facilitate such operations.

Program Overview

Some further advantages might be achievable, and a few of the disadvantages might be overcome by modifications or extensions of the program. It is written mostly in C (Software Toolworks' C/80). Therefore, it should be easy to understand and modify. The first section contains common variables and constants. These are commented, but an overview of the rest of the program will help to provide the context. There are four following segments.

Common Utilities

The first of these four segments, "COMMON UTILITIES", contains the assembly language routines and a few simple C functions. The first two are particular to HDOS. The first, console(), configures the keyboard handler to accept one byte at a time, and not echo input bytes to the video screen. The second, getkbas(), reads a byte from the keyboard, if one is available, or returns -1 if none is available. The other assembly language routines are general purpose, and do not assume HDOS to be the operating system.

Keyboard Operations

In the next section, "KEYBOARD OPERATIONS", are the routines that handle all keyboard input. This amounts to sending keyboard input to the host, detecting a command, and initiating the operation associated with a command. The possible commands are the four disk operations (T, R, U, D) plus two others (C, E). The C command toggles between MS-DOS and CP/M-86 operating modes. The E command evokes a display of the last 64 bytes (decimal ASCII form) received from the host through the modem. It is useful for debugging.

Buffer Operations And Translations

In the following section, "BUFFER OPERATIONS and TRANSLATIONS", are routines to insert and extract bytes from the modem input and output buffers, and for the translation of characters between binary and hexadecimal, when upload or download operations are underway. For the download operation, the assumption is that a CR and LF follow each 64 hex bytes. This assumption is used in the hexin() routine to ignore any leading non-hex protocol bytes, and terminate with the exclusion of any trailing bytes that do not satisfy the format.

Polling Routines

The last section, "POLLING ROUTINES", contains three sub-routines, in addition to the main program. These sub-routines are called repeatedly by the main program, in an infinite cycle, to poll for conditions requiring input and output of any kind. The routine, servmod(), tests for and handles modem action, while chekin() and chekout() take care of video and keyboard activity, as well as access to a disk file, if one is open.

Board Command And Control

To initiate one of the six commands (T, R, U, D, E, C), the ESC is pressed. The program displays a prompt, and waits for the command string to be typed. For the disk commands, the command string is the single-character command symbol, followed by a space, followed by an H-89 file name, and a RETURN. While the disk operation is underway, a status message is displayed on line 25, containing the command string. For C and E, the command string is simply the symbol followed by a RETURN. Striking the ESC key while a command operation is in progress immediately cancels the operation, closing any open H-89 disk file.

Function Keys

The H-89 function keys have no implemented functions, and other keyboard ESC sequences are effectively disabled because of this usage of ESC. To implement such functions, an ESC-sequence recognition-dispatch procedure would need to be inserted in the escop() routine, just after the function call, getstr1(filecom).

Control Characters

Directly typed control characters are available to the H-89 operating system, and have their usual effects. Control characters meant for the host must be entered by pressing the backward apostrophe key (~ — grave accent — decimal ASCII 96), and then the key for the character to be converted to a control character. The one exception is the RETURN (Control-M), which is sent to the host when typed directly, except when it occurs as the terminator of a command string.

ASCII File Transfers

Here are some examples of usual procedures. To transfer an ASCII H-89 file, ABC.TST, to a host file, XYZ.TST, first type COPY CON: XYZ.TST to the MS-DOS command processor. Then press ESC, getting a Command-> prompt. Then, type T ABC.TST. The transfer then proceeds to completion, and Control-Z is sent automatically to cause MS-DOS to close the file and end the transfer.

To transfer a host ASCII file, XYZ.TST, to an H-89 file, ABC.TST, first press ESC, getting the command prompt, as above. Then, type R ABC.TST. Then, give the MS-DOS command, TYPE XYZ.TST. The transfer then runs to completion, but you must observe when it ends and press ESC to cause the H-89 file to be closed. The R operation is a bit messy in another way, too. Some of the command protocol gets included in the received file before and after the desired file contents, and must be edited out later if it is objectionable.

Uploading and Downloading

The U and D commands are used exactly as the T and R commands, respectively. With D, however, command protocol is automatically eliminated from the received file, and the file is automatically closed when the transfer is complete. This is possible because of the rigid format produced by U and expected by D. It is a sequence of consecutive records, where each record contains exactly 64 hex bytes followed by a CR and LF.

On Adaptations And Extensions

This program works with HDOS on the H-89, and MS-DOS or CP/M-86 on the Slicer. It is written mostly in C, so extensions and adaptations should be fairly easy.

The H-89 Operating System

The first two small assembly language routines are peculiar to HDOS, for non-buffered, non-echoed, and non-waiting key-

board input, with a return of -1 to signal no byte available. These would have to be rewritten for use with H-89 CP/M. Also, because HDOS uses the LF internally as the "newline" character, there are two other minor changes that probably would be desired. The first is the removal of the statement, if (c == LF) c = CR; in the insert() routine. This statement substitutes a CR for any LF on modem output. The second is the elimination of the condition, c != CR, in line 5 of the chekin() routine. This condition prevents received CRs from entering H-89 disk files.

Program interrupts are not used by this program for handling modem activity, for two reasons. First, the polling procedure seems adequate for ordinary purposes. Second, no special exiting "cleanup" procedure is needed to shut off input from the modem, so the simple Control-C exit works adequately. The speed of execution, under some circumstances, is not quite up to what one expects at 9600 baud, so host programs using animated video may seem sluggish. Modifications to use interrupts for greater speed would involve mainly the polling procedure, and servmod(), in particular.

I have found the following configuration of the HDOS console handler to be most "natural" for this application, as well as some others. SET TT: NOMLI, SET TT: NOMLO, SET TT: BKS, and SET TT: WIDTH 255. They provide for no lower-upper case mapping, for the use of DELETE for backspace and erase, and for hardware video line wrapping.

Host Computers And Operating Systems

Other host computers or operating systems may require some adaptations. They should be minor adaptations, though, as long as a serial terminal port is used. My host system has a terminal setup procedure that was used to select the H-19 video control conventions such as cursor addressing. A host without such options might require that a translation table be built into a program such as this one.

Hardware Handshaking

The fundamental terminal handshaking used here is based on the fairly standard RS232 RTS-CTS conventions. These two lines must be present in the RS232 connecting cable. The H-89 and host computer documentation should be consulted for the proper pin terminations. I was able to use the standard ribbon-cable convention with crimp-on connectors. To read a byte from the host, the program turns RTS "on" with the portout (MODCTRL,2) command in the servmod() routine. It, then, checks the modem buffer status flag. If a byte is available, the fastop() routine is called to read the byte and quickly turn RTS "off" to prevent the host from sending more bytes before they are acceptable. If no byte is available, the RTS is merely turned "off" by the portout(MODCTRL,0) command.

The handshaking on sending a byte to the host is basically a test if CTS is "on" (portin(MODSTAT) & 16), and a test if the modem transmit buffer is ready to accept a byte (portin(LINSTAT) & 32).

LF Triggering

The other conditions shown in servmod() for sending a byte, are as follows: The condition (outptr != outpt1) is a test of whether there is a byte to be sent. If pointer outpt1 is caught up with pointer outptr, there is nothing to be sent. The condition (!crlflag | !fileop) implements the waiting for an echoed LF trigger byte after sending a CR, when a file is being sent. The host software device drivers for the command processor assume slow manual keyboard input, and do not properly use the CTS to lock out input during the brief processing delays after each line of

input. Therefore, the program must use echo triggering after each line is sent. In servmod(), crflag is set to 1 after the end of each line. In chekin(), crflag is set to 0 when the echoed LF is received. No bytes are sent in a file transfer as long as crflag = 1. The triggering procedure is slightly different for CP/M-86 and MS-DOS, and that is the main reason for needing the two modes of operation in this program. MS-DOS always echoes a CR and an LF after it receives a CR, whereas CP/M-86 usually echoes only the CR. Some tinkering with the LF triggering may be needed for implementation with other host computers. The ESC-E command is useful in such tinkering.

Summary

Upgrading the H-89 with a single-board 16-bit computer is feasible, with some advantages and some disadvantages compared to a separate 16-bit computer system. An H-89 program is presented for such an upgrade, such that the H-89 modem port is connected to the serial terminal port of the 16-bit host. The program provides for disk file transfers between the two systems, as well as terminal access to both the H-89 and host system. The program is written mostly in C and, therefore, should be easy to adapt or expand to provide additional functions.

Listing

```

/* H89 terminal program for MS-DOS and CP/M-86 on an
88/86 type computer with serial terminal port
Phillip L Emerson, Nov , 1985
Compiles with Software Toolworks C/80 compiler
*/

/*****
COMMON CONSTANTS and VARIABLES
*****/

/* misc */
#define NOMAIL -1 /* signal for no data */
#define OUTASCII 1 /* the 4 disk ops */
#define INASCII 2
#define OUTHEX 3
#define INHEX 4
#define EOF -1 /* end file, reading disk */
#define ESC 27 /* common ASCII chars */
#define BAKSPC 8
#define DELETE 127
#define LF 10
#define CR 13
#define KBEof 26
#define BELL 7
#define CONSOL 6 /* HDOS call to set KB mode */
#define SCIN 1 /* HDOS KB read call */
#define INBSIZ 10000 /* circle buffer sizes */
#define OBSIZ 10000

/* INS8250 modem @ 216 */
#define MODEM 216
#define BUFREG 216
#define INTREG 217
#define INTRID 218
#define LINCTRL 219
#define MODCTRL 220
#define LINSTAT 221
#define MODSTAT 222
#define LODIV 12 /* 9600 baud */
#define HIDIV 0
#define MODE 7 /* 8 db ,no parity, 2 sb */

/* kb & screen control */
#define CTLKEY ''
char savcurs[] = "\033j" ;
char rescurs[] = "\033k" ;
char enab25[] = "\033x1" ;
char addr25[] = "\033Y\070\040" ,
char disab25[] = "\033y1" ;
char eraslin[] = "\033l" ;
char rubout[] = "\033D\040\033D" ,

/* bufrs, ptrs, & flags, etc */
int fileop , /* 0, or one of the 4 disk ops */

```

```

int chan ; /* OS i/o channel open */
int odd , /* odd-even vrble for hex-bin conver-
sions */
int count; /* counter used in hex-bin conversions*/
int crflag , /* line-end flag, causes wait for LF
after sent CR */
int cpm , /* 1 for CP/M-86 mode, 0 for MS-DOS mode */
int inptr ; /* 1st bufptr, input */
int inptl , /* 2nd bufptr, input */
int outptr ; /* 1st bufptr, output */
int outptl ; /* 2nd bufptr, output */
int dstate , /* 0,1,2, 4,5,6, 8, states for download
procedure */
char filecom[32] ; /* file-op command buffer */
char *inbuf,*outbuf ; /* the big modem buffers */

/*****
COMMON UTILITIES
*****/
console() { /* HDOS KB char mode, no echo */
#asm
XRA A
MVI B,201Q
MVI C,201Q
SCALL CONSOL
#endasm
;}

petkbas() { /* HDOS read KB, or -1 if no data */
#asm
LXI H,NOMAIL
SCALL SCIN
JC GETOUT
MVI H,0
MOV L,A
GETOUT NOP
#endasm
;}

fastop() { /* direct modem read with fast shut
off of RTS */
#asm
IN BUPREG
MVI H,0
MOV L,A
XRA A
OUT MODCTRL
#endasm
;}

/* general purpose read-port routine */
#asm
EXECUTI DW 0
RET
#endasm
portin(port) int port , {
#asm
POP D
POP H
MOV H,L
MVI L,333Q
SHLD EXECUTI
CALL EXECUTI
MVI H,0
MOV L,A
PUSH H
PUSH D
#endasm
;}

/*general purpose write-port routine */
#asm
EXECUTO DW 0
RET
#endasm
portout(port,byte) int port,byte , {
#asm
POP D
POP B
POP H
MOV H,L
MVI L,323Q
SHLD EXECUTO
MOV A,C

```

```

CALL EXECUTO
PUSH H
PUSH B
PUSH D
#endasm
;}

putstr(s) char *s; {while(*s) putchar(*s++) ; }

char mapup(a) char a;
{if( (a >= 'a') && (a <= 'z') ) a += ('A'-'a') ,
return(a) ; }

putint(n) int n,
{int i, if(n < 0) {putchar('-'), n = -n ; }
if(i=n/10) putint(i), putchar(n % 10 + '0') ; }

screen25(a,b) char *a,*b; /* put strings a,b, to line 25 */
{ putstr(savcur), putstr(enab25), putstr(addr25), putstr(a),
putstr(b), putstr(rescur) ; }

erase25() /* erase line 25 */
{putstr(savcur); putstr(enab25), putstr(addr25),
putstr(eraslin), putstr(disab25), putstr(rescur) ; }

/* initialize modem INS8250 for baud rate, etc */
setcom()
{int i;
portout(INTREG,0) ;
portout(MODCTRL,16) ,
portout(LINCTRL,128) ;
portout(BUPREG,LODIV) ;
portout(INTREG,HIDIV) ;
portout(LINCTRL,MODE) ,
for (i=0; i < 5000; i++) ; /* delay */
portin(BUPREG) ;
portout(MODCTRL,0) ;
}

/*****
KEYBOARD OPERATIONS
*****/
/* get KB byte, process Ctrl chrs, do escop on ESC */
getkb() {static int flag, int c, char mapup() ;
c = getkbas() ;
if( c == NOMAIL ) return(NOMAIL) ;
if( c == CTLKEY ) {flag = 1; return(NOMAIL) ; }
if( c == ESC ) {escop(), return(NOMAIL) ; }
if( !flag ) return(c) ;
flag = 0; return( mapup(c) & 63 ) ;
}

/* get a command str from KB after ESC typed */
getstrl(a) char *a ,
{int c,n; char *b; b=a; n=0;
while( ( c=getkbas() ) != '\n' ) && ( n < 30 )
{if(c != NOMAIL)
{if( (c == BAKSPC) || (c == DELETE) )
{if(a <= b) putchar(BELL) ;
else {a--; n--; putstr(rubout) ; }
}
else { a = c; putchar(*a++); n++ ; }
}
}
'a = 0, putchar('\n') ,
}

/* start command T,R,U,D,C, or E */
escop()
{char mapup() ,
if (fileop)
{fclose(chan) ; erase25(), chan = -1, putchar('\n'),
fileop = 0, outptl = outptr, return ,
}
putstr("\nCommand->"); getstrl(filecom) ,
*filecom = mapup(*filecom) ,
screen25("Command in progress->",filecom) ;
if( *filecom == 'T' )
{chan=fopen(filecom+2,"r") ; fileop = OUTASCII ; }
else if( *filecom == 'R' )
{chan=fopen(filecom+2,"w") . fileop = INASCII ; }
else if( *filecom == 'U' )
{chan=fopen(filecom+2,"rb") , fileop = OUTHEX; odd=0,
count=0, }

```

```

else if( *filecom == 'D')
  {chan=fopen(filecom+2,"wb") ;
  fileop = INHEX
  odd = 0;
  count = 0;
  dstate = 0 ;
  }
else if( *filecom == 'E')
  {putstr("last 64 from modem ");
  for(inptl=((inptr-64) % INBSIZ); inptl != inptr;
    inptl = ++inptl % INBSIZ )
    {putchar(' '), putint(inbuf[inptl]) ; }
  putchar('\n'); erase25(); return;
  }
else if( *filecom == 'C')
  {putstr("mode: ");
  if(cpm) {cpm=0; putstr("MS-DOS\n") ; }
  else {cpm=1; putstr("CP/M-86\n") ; }
  erase25(); return ;
  }
if( (chan == 0) || (chan == -1) )
  {putstr("\nerror. Command cancelled ") ,
  putstr(filecom) ; putchar('\n') , fileop=0; erase25() ;
  }
inptl = inptr; outptl = outptr ;
}

```

```

/*****
      BUFFER OPERATIONS & TRANSLATIONS
*****/

```

```

/* byte to outbuf */
insert(c) char c;
{if(c == LF) c = CR;
 outbuf[outptr] = c;
 outptr = ++outptr % OBSIZ ;
 if(cpm && (c == CR) && ((fileop == OUTASCII) ||
  (fileop == OUTHEX) ) )
  {outbuf[outptr] = LF, outptr = ++outptr % OBSIZ ;
  }
}

```

```

/* byte from inbuf, or -1 */
extract()
{int c;
 if( inptr == inptl ) return(NOMAIL) ;
 c = inbuf[inptl] ;
 inptl = ++ inptl % INBSIZ ;
 return(c) ;
}

```

```

/* byte from H89 disk file, ascii or hex converted */
getcl(chan,active) int chan, active;
{ static char bitehi, bitelo; int n ;
 if(active == OUTASCII) return(getc(chan) )
 if(active != OUTHEX) return(EOF) ;
 if(count == 64) {count=0, odd=0, return('\n') ; }
 if(odd) {odd=0; count++; return(bitelo) ; }
 if( (n=getc(chan)) == EOF) return(EOF) ;
 bitelo = (n & 15) + '0' ;
 bitehi = ( (n/16) & 15) + '0' ;
 if( bitelo > '9' ) bitelo += '0' - '9' ;
 if( bitehi > '9' ) bitehi += '0' - '9' ;
 odd = 1; count++ ; return(bitehi) ;
}

```

```

/* convert 2 in-line hex to 1 bin, but -1 if illegal hex */
hexbin(hex) char hex ;
{static int tem;
 if( (hex >= '0') && (hex <= '9') )
  hex -= '0' ;
 else if( (hex >= 'A') && (hex <= 'F') )
  hex += (10 - 'A') ;
 else return(-1) ;
 if(!odd) {tem = 16*hex, return(0) ; }
 return(tem+hex) ;
}

```

```

/* process a byte for INHEX fileop, changing states */
hexin(c) char c;

```

```

{static char binbuf[32] , char d,*p ; int k;
 if( ((dstate == 1) || (dstate == 5)) && (c == CR) )
  ++dstate; return; }
else if( ((dstate == 2) || (dstate == 6)) && (c == LF) )
 {dstate = 4;
  for(p=binbuf,k=0, k < 32, k++) putc(*p++,chan) ;
  odd = 0 ;
  count = 0 ,
  return ;
  }
if(odd == 0)
 {if( hexbin(c) != 0 )
  {if( dstate > 0 ) dstate = 8 ;
   else count = 0;
   return ;
  }
  else {odd = 1 ; return; }
  }
else
 {if( (d=hexbin(c)) == -1)
  {count = 0,
   if(dstate > 0) dstate = 8 ;
   return;
  }
  else
  {binbuf[count++] = d ,
   if(count == 32)
    {count = 0 ;
     if(dstate == 0) dstate = 1 ,
     if(dstate == 4) dstate = 5 ,
     }
   }
  }
  odd = 0 .
  }
}

```

```

/*****
      POLLING ROUTINES
*****/

```

```

/* do modem input or output, as required */
servmod()
{int c;
 portout(MODCTRL,2) ;
 while(portin(LINSTAT) & 1)
  {inbuf[inptr] = fastop() ,
  inptr = ++inptr % INBSIZ ;
  }
 portout(MODCTRL,0) ;
 if(portin(MODSTAT) & 16)
  {if( (portin(LINSTAT) & 32) &&
  (!crlflag || !fileop) && (outptr != outptl) )
  {c = outbuf[outptl]; portout(BUFREG,c) ;
  if(cpm) { if(c == LF) crflag = 1 ; }
  else if(c == CR) crflag = 1,
  outptl = ++outptl % OBSIZ ;
  }
  }
}

```

```

/* dispose of new input byte, if any */
chekin()
{char c,
 c = extract() ,
 if(c == NOMAIL) return,
 if( (fileop == INASCII) && (c != CR) ) putc(c,chan) ,
 if( fileop == INHEX )
  {hexin(c) .
  if(dstate == 8)
   {fileop = 0; fclose(chan); erase25(),
   chan = -1; odd = 0, count = 0,
   }
  }
  putchar(c) ;
  if(c == LF) crflag = 0;
  }
}

```

```

/* pipe out next byte to be sent, if any */
chekout()
{char c;

```

```

if( (c=getkb()) != NOMAIL) insert(c) .
if( !crflag && ((fileop == OUTASCII) || (fileop == OUTHEX)) )
{c = getcl(chan,fileop) ;
if( c == EOF )
{fileop = 0: fclose(chan), erase2b(), chan = -1,
insert(KBEOF), insert(CR) .
}
else insert(c) ;
}
}
main()
{ int c,
console() ;

```

```

setcom() ;
inbuf = sbrk(INBSIZ) ;
if( (inbuf == 0) || (inbuf == -1) )
{putstr("memory overflow\n") , exit(1) . }
outbuf = sbrk(OBSIZ) ,
if( (outbuf == 0) || (outbuf == -1) )
{putstr("memory overflow\n") ; exit(1) ; }
putstr("Terminal program for MS-DOS or CP/M-86 host,
v 11-85\n")
while(1)
{servmod() ; chekin() ; chekout() .
}

```



FBE Products

For the H/Z-100 Series

ZMF100a — Modification package allows installation of 256K RAM chips in older Z-100 without soldering. Full compatibility with newer motherboard. **\$65**

ZCLK — Calendar/Clock module provides on-line date and time. Software included to set system date/time on automatically on bootup. Does not use S-100 bus slot. **\$89**

ZRAM-205 — Modification package allows 256K RAM chips to be put on Z-205 memory board to make one megabyte bank switched (256K banks) memory. Includes RAM disk software. **\$49**

For the H/Z-150, 160 Series

MegaRAM-150 — Modification kit allows memory board to be filled with 256K RAM chips (1.2 MByte). No soldering. Supplied with RAM disk software. **\$49.95**

ZP640 PLUS — Replacement PAL for standard memory board allows up to 2 banks of 256K and 2 or 3 banks of RAM chips to be installed for 640K or 704K maximum memory. **\$24.95**

COM3 — Replacement PAL allows installation of three serial ports (one an internal modem). Supplied with printer driver software for 3rd port. **\$39.95**

For the H/Z-138 and 148

ZC148 — Calendar/Clock module installs on CPU board. Supplied with software to set time/date automatically on bootup. **\$70**

For the H/Z-89, 90 Series

SPOOLDISK 89 — 128K byte electronic disk and printer interface/spooler card. **\$195**

H89PIP — Dual port parallel interface card. Use as printer interface. Driver software included. **\$50 Cable \$24**

SLOT4 — Extender card adds 4th I/O expansion slot to right side bus. **\$47.50**



FBE Research Company, Inc.
P.O. Box 68234
Seattle, WA 98168
(206) 246-9815 .

S & K Technology Inc.

Quality Software for Heath/Zenith Microcomputers

WatchWord™ for the Z100

The ultimate in word processing with speed and power. See the reviews in Remark (Jul 1985) and Sextant (Jan-Feb 1985; Sep-Oct 1985). Requires 192K RAM.

Price: \$100
Demo Disk: \$ 3
(Includes the Resident Speller Demo.)

The Resident Speller™ for the Z100

A spelling checker for use with WatchWord. Check your spelling as you enter text from WatchWord. Includes a stand-alone spelling checker for ASCII files.

50,000 word expandable dictionary, requires 192K RAM (300K for use with WatchWord).

Price: \$100

(Demo included on WatchWord Demo Disk.)

The Resident Speller™ PC Version for the Z150

Check your spelling as you type from most word processors.

Includes a stand-alone spelling checker for ASCII files.

50,000 word expandable dictionary. Requires 90K in addition to memory required by your word processor.

Price: \$99
Demo Disk: \$ 2

Texas Residents add state sales tax

S & K Technology, Inc., 4610 Spotted Oak Woods, San Antonio, TX 78249, (512) 492-3384



Packet Radio Description And A Simulator

Part 1

Luis E. Suarez, OA4KO/YV5
Apartado 66994
Caracas 1061-A
VENEZUELA

Packet Radio or just packet, is an error free data communications mode now used in Amateur Radio. Probably, this means very little to most computer hackers, but is certainly a remarkable achievement, considering that radio waves are plagued with interference, noise, and fluctuating communication paths. Besides, speed (that's why computers were invented) is the star light of this communications mode. Computer modems used everywhere are able to communicate with BB's through telephone lines at high speed. Even 300 and 1200 bauds are handled without a glitch. Well... almost! Packet radio sends and receives at those speeds, in a very tough environment like the air waves. But, speed is not a constraint. Radio Amateurs have sent messages up to 300,000 bauds — That's 300,000 words per minute — without losing a single character!

What Is Packet?

Basically, packet radio is digitally encoded communications. It has some resemblance with RTTY or Telex, but has important differences which are the key to ensure error free reception, and at the same time allowing maximum use of the radio spectrum. Can you imagine accessing different computers with two or more personal computers, using your single telephone line? No, I'm not talking about frequency-domain multiplexing. Packet radio uses another form of multiplexing and does it very efficiently.

The integrity of the transmitted data is provided by packet radio through hand-shaking and a special error detection technique. Along with each transmission, a frame check sequence (FCS) is sent, which is used to detect errors. The receiving station acknowledges the error free message received, with a special acknowledgment signal (ACK). If the sending station does not receive that signal, it retransmits the message.

As I said before, several stations can share the same frequency. This is possible because each message is sent along with the destination address and with the identification of the sending sta-

tion. This permits the sharing of a single frequency by several stations engaged in independent communications. Each station just ignores the packets sent to other stations. As each packet transmission is very short, each user needs the frequency for short periods and most of the time is just "listening." The time between transmissions is available to other users. This kind of channel sharing is called time-domain multiplexing.

AX.25 Level 2 Protocol

Protocol is the rules governing the transfer of information. As the Webster says, "... correct procedure, as in diplomatic exchange and ceremonies." In fact, any communications protocol establishes the regulations of everything, from hardware to the simplest logic decision software controlled. Protocols are even used by animals and persons. The protocol you use, if you ever have the chance, to talk to the country's president is different than the one used to talk to a friend. As a must, two persons should use the same protocol to talk each other. If one's language is different, the information cannot be transferred. You cannot talk to a deaf person if you don't know Asmelan language. Computers use a protocol too, and you know what happens if you try to transfer a CP/M file to an HDOS disk. Did you ever hear about compatibles? Yes, a case of protocol.

In packet radio, the protocol used is designed to enable many users to share a single radio frequency for point-to-point communications, with optimum reliability. The rules are based in the International Standards Organization (ISO) that proposed a 7-layer model for packet-switching. If you are curious, they are contained in the Recommendations 3309 and 4335, including DAD 1&2 and in 6256 High Level Data Link Control (HDLC). This conforms with ANSI X3.66 from our old friend the American National Standards Institute.

The seven layers that make up the OSI reference model for packet radio are:

Level 1	Physical Layer
Level 2	Link Layer
Level 3	Network Layer
Level 4	Transport Layer
Level 5	Session Layer
Level 6	Presentation Layer
Level 7	Application Layer

At this moment, only the first two layers have been implemented for Amateur Radio use. However, Level 3 is almost ready and several proposals are under study for all the remaining levels, too.

Physical Layer

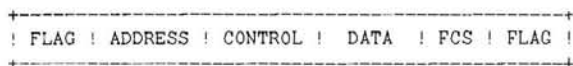
This layer is concerning the hardware environment. That's why it is also called the Physical Layer. Among other things, it deals with the voltage levels, data handshaking signals, speed of data transmission and reception, order of bit transmission and reception, type of modems and radio frequency signaling channels.

The equipment necessary to transmit packets uses a micro-processor and is named, Terminal Node Controller. This TNC is serial connected in RS-232C configuration to a terminal and to a radio via a Bell 202 style modem, for up to 1200 bauds communication speed. A personal computer may be used as a terminal to take advantage of the mass storage capability. However, a dumb terminal will suffice. In fact, the job is done by the TNC. The modem is also included within the TNC, although an out-board one is generally used for higher than 9600 baud speeds.

Briefly, the latest TNC is enclosed in a 10 by 6 inch metal box, resembling a modem, and features a Z-80 with up to 32K of ROM and 16K of RAM. A Z8440 configured as UART is used for I/O interface. The modem is built around the XR2211 and XR2206, plus a state of the art MF-10 switched capacitor filter. A non-volatile battery backed RAM stores the operating parameters. Supported baud rates are 300, 1200 and 9600 for the radio and from 300 to 9600 for the terminal. The TNC is commanded by 95 software commands from the terminal keyboard. The only switch used is the power switch located in the rear panel. The front features 4 LEDs used for signaling purposes.

Link Layer

The Link Layer is responsible for receiving and transmitting data from the higher level protocols to the physical layer. When you write a message in the keyboard, the TNC packetizes it. That's why this communication mode is named Packet Radio. The frame of a packet is represented below.



The FLAG field is a unique 1 byte sequence: 01111110. To make it unique, five or more consecutive 1 bits are found in the data, a zero bit is added after the fifth 1 bit. The receiving TNC will realize that a "stuffing" was performed and delete the bit to restore the integrity of the data received.

The ADDRESS field contains routing information for the packet. This information may include the destination station, the originating station and possible relay points.

The CONTROL field specifies the purpose of the packet. The TNC is constantly interchanging information in a true handshaking, via radio.

The DATA field is the information you wish to send. It may contain plain text or any or all of 128 ASCII characters. In fact, you can

transfer anything from ASCII to binary files or digitized voice and television frames. Anything that may be digitized, is packetized in groups of up to 256 bytes and transmitted.

The FCS field contains the frame check sequence. The receiving stations calculate the FCS of the received data and makes comparisons with the received FCS. If wrong, the packet is not acknowledged and hence retransmitted.

The final FLAG, marks the end of the packet and is similar to the first FLAG.

With the implementation of these two layers, radio amateurs are able to send and receive messages and computer files. They even use satellites to relay communications when the radio path is not reliable enough for a given contact. To extend the radio path in very high and ultra high frequencies, the packet is relayed. They send a message addressed to a given station via one or more relay stations. The TNC at the relay station stores the message in RAM and immediately retransmits the packet to the next address. This way, a cross country link may be established. While this happens, the relay stations are not aware of the retransmitted messages, unless the operator wishes to see them. It is interesting to note that each packet station is a repeater station and all may be used to relay communications in order to extend the radio coverage. When relaying, or digipeating the information, the station operates transparently for the operator, if he or she so desires. In other words, each amateur operator leaves the station unattended to help others to reach greater distances, and this can also be done simultaneously while the operator is in contact with other stations. The TNC has enough memory and "intelligence" to do the task very efficiently.

It is easy to understand now that while the operation of a packet station is simple for the operator, the TNC is very busy all the time. It must receive and send information through the terminal and radio ports. It controls the modem and watches for messages addressed to it. It must decide if the message is to be sent to the terminal or must be relayed. It must acknowledge the correct packets received, must request retransmission of those received incorrectly and must ignore all those addressed to other stations. The TNC must send its own packets taking care of all the messages coming and leaving its memory. Besides, it must time many events as to how long to wait before transmitting and how long to wait before sending and ACK. The TNC must know who it is talking to in order to tell other stations if it is busy. Besides, it must keep track of the number of times a packet has been transmitted. The TNC must know, and take care of, what stations it is able to be connected with, or not be connected with. It handles a directory of all stations contacted, including the date and time. It must handle the terminal operation also, specially if yours is a dumb one. There is a lot of work around the terminal display, like line length, to echo or not to echo a character, to print or not to print a selected character, to wrap around the end of line or not etc., etc. And, there are even more and that's why the latest TNC, released by Tucson Amateur Packet Radio has 95 commands!

What Else In The Future?

At the present stage, besides sending and retrieving files or typed messages, you can perform interesting things. For instance, I have transferred binary files and have sent synthesized voice. I have also played Chase using cursor addressing. This is unique to packet radio, an impossible task for radioteletype. Also, multiple connections are allowed, a very useful feature for traffic net operation.

Multi-jointed fully articulated arm

A unique arm and gripper mechanism with a sense of touch – patented by Heath Company.

Manipulate a variety of objects with human-like dexterity.



A hand-held wireless remote control console with full ASCII keyboard for manually controlling and remotely programming HERO 2000 from up to 100 feet away

Sonar, light, sound, touch and temperature sensors



Status display flashes condition of 16 functions



Two RS-232C DCE ports plus provision for four other back panel connectors



Real panel connector enables HERO 2000 to dock with its heavy duty charger

HERO 2000 – a complete automation learning system for exploring and testing the hardware and software used in real-world intelligent machines and robots



Series of two courses that support the HERO 2000 Automation Trainer



HERO 2000 Robot and Courses



HERO 2000 is a high-technology tool that enables you to explore and learn about controlling real-world automated systems which are intelligent machines. HERO 2000 is also an educational trainer for use in simulating industrial robotic applications. And HERO 2000 is an imagination machine with enormous capability that can stimulate, challenge and inspire the user.

Unmatched technology at
\$1999⁹⁵

- A master microprocessor plus eleven other processors that control motors, sensors and interfacing
- A multi-jointed, fully articulated arm
- A synthesized voice with an unlimited vocabulary
- Five different senses; remote wireless control

New Automation Course provides a comprehensive look at the electronics used at the component level

\$99⁹⁵

- A component by component look at the electronics used in automated industrial robotic systems
- Used with the HERO 2000 Automation Trainer to transform textbook theory into exciting, hands-on experience

Learn about the technical systems that affect all of us in the New Intelligent Machines Course

\$99⁹⁵

- Ties together computers, robotics and sensor systems
- Written so that even a novice can understand the technical aspects of intelligent machines

Come in for a demonstration today!

ZENITH data systems

AUTHORIZED DEALER

LEASING AVAILABLE FOR YOUR BUSINESS NEEDS

Fine Electronics Since 1962

HZC-219



Call 616-982-3614 for store locations or consult your Heathkit Catalog.



Your **TOTAL SERVICE** computer center
Service • Support • Software • Accessories • User Training • Competitive Prices

Heath **ZENITH**
Computers & Electronics

Units of Veritechnology Electronics Corporation

In less than a year, a new amateur satellite will be launched. It will have a 2 megabytes memory bank devoted to packet radio. This bird will be used for storage and retrieval of radio messages from Amateur Radio stations around the world. A future Space Shuttle mission will carry a Packet Digipeater to relay packet transmission from Radio Amateurs around the world.

The next development step is a radio network that will be created to link all the hemisphere. The level 3 is almost ready and will be put into practice very soon. There will be links interconnecting satellites and Earth stations. What is really interesting is the possibility to access mainframes. Not just mail boxes, but a real access to operating system level. Can you imagine accessing a mainframe, listing the directory and running a Fortran program with all the power, speed and mass storage the main frame provides? Can you imagine having access to a database stored in 20 or 30 megabytes of memory? Can you believe that Layer 6 will make it possible to correctly display your screen, no matter what the cursor addressing of the correspondent station? Your H-89 will converse with a TR-80, and both terminals will display the correct screen!

The Session Layer of the protocol, will allow several operators using the same program, or one station running one program while another is editing a document, both accessing a mainframe under MP/M II. The Application Layer will standardize the operation of a particular program by the user. It seems that there is no limits in the future of packet radio. Just use your imagination!

```

10 REM          PACKDEM.BAS VER 1.1
20 REM          November 1985
30 REM          by
40 REM
50 REM          Luis E Suarez, OA4KO
60 REM
70 REM          PO BOX 66994
80 REM          CARACAS 1061-A
90 REM          VENEZUELA
100 REM
110 REM This program simulates a limited operation of Packet
120 REM Radio The code emulates ONLY, the display output of
130 REM a typical TNC and resembles, in a limited form, the op-
140 REM eration of the TNC-2, a product developed and sold by:
150 REM
160 REM          TUCSON AMATEUR PACKET RADIO CORPORATION
170 REM
180 REM          PO BOX 22888
190 REM          TUCSON, AZ 85734-2888
200 REM          (602) 746-1166
210 REM
220 REM This program is released to the public domain for
230 REM personal use and should not be sold for profit or
240 REM used for advertising or promoting commercial products,
250 REM except by written permission of the author and Tucson
260 REM Amateur Packet Radio Corporation
270 ES=CHR$(27):CU$=E$+"Y":SAVECU$=E$+"j":SETCU$=E$+"k"
280 DIM CMD$(95),ATRIB$(95)
290 PRINT E$b":RANDOMIZE PEEK(12)
300 FOR A%=1 TO 95:READ CMD$(A%):NEXT
310 PRINT " |A":PRINT
320 PRINT"bbRAM loaded with defaults":PRINT CHR$(7)
330 FOR A%=1 TO 95:READ ATRIB$(A%):NEXT
340 PRINT"Tucson Amateur Packet Radio TNC 2"
350 PRINT"AX.25 Level 2 Version 2.0"
360 PRINT"Release 1.1.2 12/02/85 - 16K RAM"
370 PRINT"Checksum $33"

```

```

380 C0$="":C1$="":C2$="":C02$="":PRINT SAVECU$:
390 PRINT"cmd:":
400 IF INP(232)=13 THEN A$=INKEY$: IF A$=CHR$(13) THEN PRINT:GOTO 390
410 IF INP(232)>26 THEN 440
420 GOSUB 1960
430 GOTO 400
440 C$=INPUT$(1):PRINT C$: IF ASC(C$)<>13 THEN C0$=C0$+C$:GOTO 440
450 PRINT
460 FOR I%=1 TO LEN(C0$):L=ASC(MID$(C0$,I%,1))
470 IF L>96 AND L<123 THEN MID$(C0$,I%,1)=CHR$(L-32)
480 NEXT
490 FLAG1=INSTR(C0$," ") IF FLAG1=0 THEN 510
500 C01$=LEFT$(C0$,FLAG1-1):C02$=MID$(C0$,FLAG1+1)
C0$=C01$
510 IF LEN(C0$)>9 THEN 1550
520 FOR A%=1 TO 95:IF C0$<>CMD$(A%) THEN NEXT
530 IF A%>38 AND A%<75 THEN 560
540 IF A%>75 THEN 570
550 ON A% GOTO 600,610,620,630,640,650,660,670,680,690,
700,710,720,730,740,750,760,770,780,790,800,880,890,
900,910,920,930,940,950,960,970,980,1000,1010,1020,
1030,1020,1030,1040
560 ON A%-38 GOTO 1050,1060,1070,1080,1090,1100,1110,1120,
1130,1140,1150,1160,1170,1180,1190,1200,1210,1220,1230,
1240,1250,1260,1270,1280,1290,1300,1310,1320,1330,1340,
1350,1360,1370,1380,1390,1400,1410
570 ON A%-75 GOTO 1420,1430,1440,1450,1460,1470,1480,1481,
1482,1483,1484,1485,1486,1487,1488,1489,1490,1491,
1492,1493
580 GOTO 1520
590
600 GOSUB 1840:GOTO 380:8BITCONV
610 IF C02$="OFF" THEN PRINT CHR$(27)"Y8",:GOTO 600
ELSE IF C02$="ON" THEN PRINT CHR$(27)"x8",:GOTO 600:
REM AUTOLF
620 GOTO 600' AWLEN
630 GOTO 600' AX25L2V2
640 GOTO 600' AXDELAY
650 GOTO 600' AXHANG
660 GOTO 600' BEACON
670 GOTO 600' BKONDEL
680 GOTO 600' BTEXT
690 GOTO 600' BUDLIST
700 A$=INPUT$(1):
IF A$="Q" OR A$="q" THEN GOTO 380 ELSE 700' CALIBRA
710 GOTO 600' CALSET
720 GOTO 600' CANLINE
730 GOTO 600' CANPAC
740 GOTO 600' CHECK
750 GOTO 600' CMDTIME
760 GOTO 600' CMSG
770 GOTO 600' COMMAND
780 GOTO 600' CONMODE
790 IF LEN(C02$)<4 OR LEN(C02$)>9 THEN 1500' CONNECT
800 IF CALL1$<>C02$ THEN IF RND(1)>.85 THEN GOTO 1670
GOTO 380
810 IF FLAG2=1 THEN GOSUB 1560 ELSE FLAG2=1:CALL2$=C02$
FOR Y=1 TO 500:NEXT
820 GOSUB 1630
IF CALL1$<>CALL2$ THEN FLAG3=0:GOTO 1720 ELSE FLAG3=1

```

```

830 Z$=INPUT$(1):IF ASC(Z$)=VAL(ATTRIB$(18)) THEN W$="":
GOTO 380
840 IF ASC(Z$)=VAL(ATTRIB$(70)) THEN PRINT:GOTO 860
ELSE PRINT Z$;:W$=W$+Z$
850 IF LEN(W$)<>VAL(ATTRIB$(60)) THEN 830
860 FOR A=1 TO 250:NEXT:PRINT W$:W$="":GOTO 830
870 GOTO 600' CONOK
880 GOTO 600' CONSTAMP
890 IF FLAG3=1 THEN 830 ELSE 1720' CONVERS
900 GOTO 600' CPACTIME
910 GOTO 600' CR
915 IF FLAG2=1 THEN PRINT
"A stream - IO Link state is CONNECTED to ";CALL2$
ELSE PRINT "A stream - IO Link state is DISCONNECTED"
916 FOR Y=66 TO 74:
PRINT CHR$(Y)" stream - Link state is DISCONNECTED":
NEXT
917 GOTO 380
920 GOTO 600' CTEXT
930 GOSUB 1510:GOTO 380' DAYTIME
940 GOTO 600' DAYUSA
950 GOTO 600' DELETE
960 GOTO 600' DIGIPEAT
970 IF FLAG2=0 THEN GOSUB 1590 ELSE GOSUB 1660:GOTO 600'
DISCONNE
980 FOR A%=1 TO 82:
IF ATTRIB$(A%)<>"*" THEN PRINT CMD$(A%),ATTRIB$(A%)
990 NEXT:GOTO 380' DISPLAY
1000 GOTO 600' DWAIT
1010 GOTO 600' ECHO
1020 GOTO 600' ESCAPE
1030 GOTO 600' FLOW
1040 GOTO 600' FRACK
1050 GOTO 600' FULLDUP
1060 GOTO 600' HEADERLN
1070 GOTO 380' ID
1080 GOTO 600' LCALLS
1090 GOTO 600' LCOK
1100 GOTO 600' LFADD
1110 GOTO 600' MALL
1120 GOTO 600' MAXFRAME
1130 GOTO 600' MCOM
1140 GOTO 600' MCON
1150 GOTO 600' MFILTER
1160 ATTRIB$(51)="":GOTO 380' MHCLEAR
1170 IF CO2$<>"*" THEN 1525 ELSE PRINT CO$+" "+ATTRIB$(51)
:GOTO 380' MHEARD
1180 GOTO 600' MONITOR
1190 GOTO 600' MRPT
1200 GOTO 600' MSTAMP
1210 IF CO2$="*" THEN 600 ELSE CALL1$=CO2$:
GOTO 600' MYCALL
1220 GOTO 600' NEWMODE
1230 GOTO 600' NUCR
1240 GOTO 600' NULF
1250 GOTO 600' NULLS
1260 IF CO2$="*" THEN 600 ELSE ATTRIB$(60)=CO2$:
GOTO 600' PALEN
1270 GOTO 600' PACTIME
1280 GOTO 600' PARITY
1290 GOTO 600' PASS
1300 GOTO 600' PASSALL
1310 GOTO 600' REDISPLA
1320 RESTORE 300:GOTO 300' RESET
1330 GOTO 600' RESPTIME
1340 IF CO2$="*" THEN 600 ELSE ATTRIB$(68)=CO2$:
GOTO 600' RETRY
1350 IF CO2$="*" THEN 600 ELSE WIDTH VAL(CO2$):
ATTRIB$(A%)=CO2$:GOTO 600' SCREENLN
1360 SENDPAC=VAL(CO2$):GOTO 600' SENDPAC
1370 GOTO 600' START
1380 GOTO 600' STOP
1390 GOTO 600' TRACE
1400 A$=INPUT$(1):IF ASC(A$)<>VAL(ATTRIB$(18)) THEN 1400
ELSE A$=INPUT$(1):IF ASC(A$)<>VAL(ATTRIB$(18)) THEN 1400
ELSE A$=INPUT$(1):IF ASC(A$)<>VAL(ATTRIB$(18)) THEN 1400
ELSE 380' TRANS
1410 GOTO 600' TRFLOW

```

```

1420 GOTO 600' TXDELAY
1430 GOTO 600' TXFLOW
1440 GOTO 600' UNPROTO
1450 GOTO 600' XFLOW
1460 GOTO 600' XMITOK
1470 GOTO 600' XOFF
1480 GOTO 600' XON
1481 GOTO 600' LCSTREAM
1482 GOTO 600' NOMODE
1483 GOTO 600' TRIES
1484 GOTO 600' USERS
1485 GOTO 600' STREAMDBL
1486 GOTO 600' STREAMSW
1487 GOTO 600' STREAMCA
1488 GOTO 820' RECONNECT
1489 PRINT:GOTO 340' RESTART
1490 GOTO 915' CSTATUS
1491 GOTO 600' MYALIAS
1492 GOTO 600' CLKADJ
1493 GOTO 600' CONPERM
1500 PRINT"?call":GOTO 380
1510 PRINT"?clock not set":RETURN
1520 PRINT"?EH":GOTO 380
1525 PRINT"?bad":GOTO 380
1530 PRINT"?not enough":GOTO 380
1540 PRINT"?range":GOTO 380
1550 PRINT"?too long":GOTO 380
1560 PRINT"?too many":GOTO 380
1570 PRINT"?VIA":GOTO 380
1575 PRINT"?Not while connected":GOTO 380
1576 PRINT"?Not while disconnected":GOTO 380
1577 PRINT"?already connected to that station":RETURN
1580 PRINT"Link state is: CONNECTED to ";CALL2$:GOTO 380
1590 CALL2$="":PRINT"Link state is: DISCONNECTED":GOTO 380
1600 PRINT"Link state is: CONNECT in progress":GOTO 380
1610 PRINT"Link state is DISCONNECT in progress":GOTO 380
1620 PRINT"Link state is. FRMR in progress":GOTO 380
1630 PRINT CHR$(13);"*** CONNECTED to ";CALL2$:RETURN
1640 PRINT"*** connect request: ";GOTO 380
1645 PRINT"*** LINK OUT OF ORDER, possible data loss
[opt. daytime stamp":GOTO 380
1650 PRINT"*** retry count exceeded"
1660 FLAG2=0:FLAG3=0:GOSUB 1950:PRINT"*** DISCONNECTED":
GOTO 380
1670 PRINT"*** ";CO2$;" busy"
1680 PRINT"*** DISCONNECTED":GOTO 380
1690 PRINT"frmr frame just sent:":
PRINT"FRMR sent: hexadecimal value":GOTO 380
1700 PRINT"FRMR rcvd:","hexadecimal value":GOTO 380
1710 '
1720 PRINT CU$;CHR$(31+12);CHR$(31+0);
1730 PRINT E$b";:IF FLAG2=1 THEN GOSUB 1830
1740 PRINT SETCU$
1750 W$="
1760 Z$=INPUT$(1):IF ASC(Z$)=VAL(ATTRIB$(18)) THEN W$="":
GOTO 380
1770 IF ASC(Z$)=VAL(ATTRIB$(14)) THEN
FOR A=1 TO LEN(W$):PRINT CHR$(8);:NEXT:GOTO 1750
1780 IF ASC(Z$)=VAL(ATTRIB$(70)) THEN PRINT SAVECU$::
GOTO 1800 ELSE PRINT Z$;:W$=W$+Z$
1790 IF LEN(W$)<>VAL(ATTRIB$(60)) THEN 1760
1800 PRINT CU$;CHR$(43);CHR$(31);E$;CHR$(108);CHR$(13),
CHR$(8);CHR$(8);
1810 IF CO$="CONVERS" AND ATTRIB$(52)="ON" AND FLAG2=0 THEN
PRINT ATTRIB$(55);">";ATTRIB$(78);",",
1820 PRINT W$;SETCU$:PRINT E$I":GOTO 1750
1830 SWAP CALL1$,CALL2$:GOSUB 1630:SWAP CALL1$,CALL2$:RETURN
1840 IF FLAG1=0 THEN PRINT CO$+" "+ATTRIB$(A%):RETURN
1850 IF ATTRIB$(A%)<>CO2$ THEN PRINT CO$;" was "+ATTRIB$(A%):
ATTRIB$(A%)=CO2$:RETURN ELSE PRINT CO1$+" "+ATTRIB$(A%):
RETURN
1860 DATA 8BITCONV,AUTOLF,AWLEN,AX25L2V2,AXDELAY,AXHANG,
BEACON,BKONDEL,BTEXT,BUDLIST,CALIBRA,CALSET,CANLINE,
CANPAC,CHECK,CMDTIME,CMG$
1870 DATA COMMAND,CONMODE,CONNECT,CONOK,CONSTAMP,CONVERS,
CPACTIME,CR,CTEXT,DAYTIME,DAYUSA,DELETE,DIGIPEAT,
DISCONNE,DISPLAY,DWAIT
1880 DATA ECHO,ESCAPE,FLOW,FRACK,FULLDUP,HEADERLN,HID,ID,

```

```

LCALLS,LCOK,LFADD,MALL,MAXFRAME,MCOM,MCON,MFILTER,
MHCLEAR,MHEARD,MONITOR,MRPT
1890 DATA MSTAMP,MYCALL,NEWMODE,NUCR,NULF,NULLS,PACLEN
1900 DATA PACTIME,PARITY,PASS,PASSALL,REDISPLA,RESET,
RESPTIME,RETRY,SCREENLN,SENDPAC,START,STOP,TRACE,TRANS,
TRFLOW,TXDELAY,TXFLOW,UNPROTO,XFLOW,XMITOK,XOFF,XON
1910 DATA LCSTREAM,NOMODE,TRIES,USERS,STREAMDBL,STREAMSW,
STREAMCA,RECONNECT,RESTART,CSTATUS,MYALIAS,CLKADJ,CONPERM
1920 DATA OFF,ON,7,OFF,0,0,EVERY 0,ON,"",OFF,"*",2060,24,
25,30,1,OFF,4,CONVERS,"",ON,OFF,"",OFF,ON,"",",",
ON,OFF,ON,"",",",16,ON
1930 DATA OFF,ON,3,OFF,OFF,OFF,"",",",ON,OFF,ON,4,OFF,ON,
,"",",",ON,ON,OFF,"" NOCALL",OFF,OFF,OFF,0,128,
AFTER 10,3,22,OFF
1940 DATA 18,"",12,10,00,13,17,19,OFF,"",OFF,30,OFF,CQ,
ON,ON,19,17
1945 DATA ON,OFF,0,1,OFF,124,OFF,"",",",",",",0,OFF
1950 ATRIB$(51)=ATRIB$(51)+CALL2$+" "·CALL2$="":RETURN
1960 RANDOMIZE PEEK(12):IF ATRIB$(52)<>"ON" THEN RETURN
1970 IF RND(1)<.0001 THEN PRINT "W1AW>WB9FLW:
Hi Pete, TNC-2 is super!!!"
1975 CA$="":IF RND(1)>.0005 THEN 1980
1977 CA$="KA"+MID$(STR$(FIX(RND(1)*10)),2)
1978 FOR A=1 TO 3:CA$=CA$+CHR$(65+(FIX(RND(1)*10))):NEXT:
PRINT CA$;">CQ:This a test packet"
1980 IF RND(1)>.9999 THEN PRINT"0A4KO>CQ:
Hello there!! This is Packet!!"
1990 RETURN
2000 END

```

Equipment Available

TNC-2, Tucson Amateur Packet Radio, PO Box 22888, Tucson AZ 85734.
 HD-4040, Heath Company.
 Packet Communicator, Kantronics.
 PK-1 Packet Radio Controller, GLB Electronics, Inc.
 PK-80 Packet Controller, Advanced Electronic Application.
 PK-64 Packet System for Commodore 64, Advanced Electronic Application.
 Packet for TRS-80, software approach, Richcraft Engineering Ltd., 22 North Lake Drive, Chatauqua Lake, NY 14722.
 PAC/NET System, Bill Ashby and Son, Box 332 Pluckemin NJ 07978.

Frequencies

145.832 MHz (satellite OSCAR 10),
 145.01 MHz,
 14.103 MHz,
 10.149 MHz,
 7.097 MHz,
 3.642 MHz,
 3.630 MHz.

*

Publications

PSR Quarterly, published by TAPR, Edited by Gwyn Reedy
 AMRAD Newsletter, Amateur Radio Research and Development Corp., PO Drawer 6148, McLean, VA 22106-6148.
 FADCA Beacon, (newsletter), The Florida Amateur Digital Communications Association, 812 Childers Loop, Brandon, FL 33511.
 Gateway, The Packet Radio Newsletter, from The American Radio Relay League.
 Gateway, The Packet Radio Newsletter (Spanish), Radio Club de Chile.
 COMUNICA. Newsletter in Spanish, Apartado 66994, Caracas 1061-A, Venezuela
 AX.25 Amateur Packet Radio Link Layer Protocol, available from ARRL.
 First, Second, Third and Fourth Computer Networking Conference Proceedings. Available from ARRL.
 Amateur Radio Software, by John Morris, GM4ANB, from ARRL.

Articles

H. Price, "A Closer Look At Packet Radio.", QST August 1985.
 H. Price, "What's All This Racket About Packet", QST July 1985.
 H. Price, "Join The Packet-Radio Revolution", Part III, 73 Magazine, January 1984.
 H. Price, "Join The Packet-Radio Revolution", Part II, 73 Magazine, October 1983.
 H. Price, "Join The Packet-Radio Revolution", Part I, 73 Magazine, September 1983.
 M. Morrison and D. Morrison, "Amateur Packet Radio: Part 2", Ham Radio, August 1983.
 M. Morrison and D. Morrison, "Amateur Packet Radio: Part 1", Ham Radio, July 1983.
 D. Borden and P. Rinaldo, "The making of an Amateur Packet Radio Network", QST, October 1981.
 I. Hodgson, "An Introduction to Packet Radio", Ham Radio, June 1979.

HOME FINANCE SYSTEM VERSION 2

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

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

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

—System consists of 130 page users manual with 5 program disks (5-1/4") and a sample data disk.

Hardware: H8/HZ89 (64K) or HZ100 with 2 disk drives. Any Heath*, Zenith® or other printer.

Software: CP/M or CP/M-85/86 (Ver. 2.2) and MBASIC 5.21 for CP/M.

Order: Complete System \$89† (specify hard or soft sector 5¼", HZ89 or HZ100). Manual alone \$21.†

MasterCard/Visa accepted, please include your phone number.

†Prices include shipping.



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



Jim Buszkiewicz
HUG Software Developer

Paranoia Reigns Supreme Or Is Copy Protection Really Necessary?

The opinions expressed in this article are not necessarily those of Heath or Zenith, but should be!

Although the subject of copy protection has been addressed many times in other publications, I feel that there is a need to discuss it at least once with the Heath/Zenith community.

Back in the good-ole' days, when CP/M (remember CP/M?) was king, copy protection was almost unheard of. Software vendors marketed their products on standard 8" disks, and users were able to make back-ups of these programs using standard operating system utilities such as PIP. For you MSDOS types, PIP is to CP/M as COPY is to MSDOS. A user was also able to exactly duplicate his disk using a manufacturer supplied utility, sometimes called DUP or COPYDISK. Realizing that anyone with a CP/M based computer could copy their software, software manufacturers charged tremendously inflated prices for their wares. This was done under assumption that, for every piece of software sold, one or more would get copied (stolen), or just plain given away. Unfortunately, this assumption was probably correct. The question that should be asked here is, was there piracy because of high software prices, or were the software prices high because of piracy? In this case, I feel the egg came before the chicken, and piracy resulted from ridiculous software prices.

Time marches on, and all good things give way to better. IBM and Microsoft have set the hardware and software standards with the 'PC' and MSDOS, respectively. Since the hardware is now specifically defined (if you want to manufacture a PC clone, you must follow the hardware standard, exactly), software manufacturers now have TOTAL control over your machine with their product! If you think that's a scary thought, read on, 'cause you're right!

I guess it all began when some unknown software writer (probably at MicroSoft) discovered that he could actually read and write on the 41st track of a 40 track drive. This, of course, was accomplished by going directly to the disk controller with the

proper software commands. It didn't take long to realize that the 41st track wasn't recognized by MSDOS, and normal methods of disk copying wouldn't work on data written to that track. One thing lead to another, and before too long, programs were using track 41 to hide parts of themselves on, and thus copy protected software was born.

Even though most PC clones were hardware and software compatible, not all drives were capable of going out to this 41st track. Because of this, some software that used this method of copy protection wouldn't work on all PC look-a-likes unless 'compatible' disk drives were used! For a short time, Heath/Zenith machines had this problem.

Today, there are probably as many ways of copy protecting software, as there are software vendors. As an example, one of the more popular copy protection schemes involves writing a special magnetic pattern on the disk. These patterns might be a track made up of a peculiar number of sectors, or a series of odd sized sectors. Another method is to give a sector a strange ID, or to physically overlay a series of sectors. One method reflects the mentality of its' creator. This technique uses 'weak' bits. This magnetic pattern is actually made up of unstable data that has a different value each time it's read! If the program gets different values each time these 'weak' bits are read, it then knows that this is an original disk.

Another most popular method of copy protection is to purposely write errors in certain locations on the disk. The program then looks for these 'errors' and if it finds them, executes properly. This method is used by Softguard Systems for its SUPERLoK protection scheme. Softguard now allows copy protected software to be placed on a hard disk by hiding files in certain locations on the disk. Although usable, these 'hidden' files interfere with normal hard disk backups, and cannot themselves be backed up. Older methods of using copy protected software with hard disks involved a 'key' disk always being installed in drive A:. If this key wasn't available, the application program on

the hard disk wouldn't run. All these types of copy protection are considered 'software' methods.

The second type of copy protection is hardware, or physical in nature. The best known, of course was PROLOK by Vault Corporation. PROLOK works by burning one or more tiny holes in the protected disk with a laser! A little more than a year ago, it was rumored that Vault had designed a 'killer' version of PROLOK. This version, if it couldn't find the laser holes on a bogus copy, would wipe out data. If the disk was write protected, it would wait and smash the next unsuspecting disk! Vault supposedly never released this product. Some companies require that a 'key' module be plugged into one of the RS-232 serial ports before their software will run. This module contains some sort of ROM, which the application program reads before working properly.

If you're not convinced that paranoia exists yet, read on. Version 2 of WORD from MicroSoft has a real doozy built into it. Normally, not only is the software copy protected, but it also traps out the break point interrupt vector so you can't debug the program. If you replace this vector while the program is running, the following message gets displayed:

```
***INTERNAL SECURITY VIOLATION***  
the tree of evil bears bitter fruit  
    crime does not pay  
    THE SHADOW KNOWS
```

Trashing program disk

The software then proceeds to turn on the default drive and move the heads back a forth as if wiping out data. So far it hasn't really done any damage, but is this the way of the professional software writer?

My research for this article actually began when HUG purchased a copy of dBASE III PLUS for internal use. The package came with seven or eight disks, two of which were copy protected system disks. Each of those two disks could be installed on a hard disk only once. Once they were installed, the system diskettes were then useless unless the system was de-installed again from the hard disk. Once installed on a hard disk, you were warned not to backup or restore that drive for you would lose that copy of dBase forever! Such Nonsense! At this point my biggest concern was, what happens if an error occurs during the installation, or de-installation process? Would that copy of dBASE be destroyed? For those of you who aren't aware, dBASE III PLUS retails for around \$700. Not the hunk of change you'd want to lose because one of your memory cells decided to "have a headache"! I was furious at being at the mercy of some paranoid software vendor. I have the RIGHT to be able to backup my investment. Would you drive around in your brand new Cadillac without full coverage insurance on it! Further searching revealed that I wasn't the only user to feel this way. In fact, ever since the dawn of copy protected software, three major software vendors have been selling programs to defeat these copy protection schemes.

The first company I would like to talk about is TranSec Systems, Inc. They sell a product called UNlock. Three versions, or albums, of UNlock are available, and each version is capable of TOTALLY removing the copy protection from 7 different copy protected programs! A fourth version, or album, is also available, and this version works with the six most popular programs from the first three albums. The following are the more popular programs presently supported: dBASE III & dBASE III PLUS, FRAMEWORK I & II, LOTUS 1-2-3, REALIA COBOL, SYMPHONY,

CLIPPER, DOUBLEDOS, MICROSOFT WORD, MULTILINK ADVANCED. TranSec is quite confident in their product. "Unlock products will remove the copy protection from the programs specified in its advertisements." or "TranSec will either provide a repaired copy of UNlock which will work on your copy of the program, or refund your money". Within minutes of using UNlock on my version of dBASE III PLUS, I was able to copy it to any disk I chose using the standard MSDOS COPY command! Here's to you Mr. Ashton and Mr. Tate!

Next in line is Central Point Software of Portland, Oregon. They market a product called COPY II PC. This product is more general purpose in nature and quite easy to use. Copies can be made using a one or two drive system and the program operates much in the same manner MSDOS's DISKCOPY works. Although a bit slower than DISKCOPY, COPY II PC not only formats each track as the copy is made, but also does a verification on the destination. Because drive rotation speed is critical in making accurate copies, COPY II PC has a built-in drive speed test in case you're encountering problems. This product comes with a list of approximately 200 software products which are copyable by COPY II PC. Some products may require special attention when making the copy, and these are indicated on that list. This list, by the way, is by no means conclusive. Also included with COPY II PC, are two programs called NOKEY and NOGUARD. NOKEY is a utility which allows certain versions of protected software to run on a hard disk without the need for a 'key' disk in drive A:. NOGUARD is a utility which allows you to make functional backups of certain programs and to run those programs from a hard disk without fussing with their install/uninstall (dBASE III PLUS) or copyon/copyoff procedures. For most applications (including games), COPY II PC is the best suited copying program available.

It might be interesting to note here that in a most recent issue of PC Magazine, Michael Brown, the author of COPY II PC, was refused his "gold" certification for that product by the Software Publishers Association. This certification is awarded when sales for a software product reaches 50,000 and 100,000 copies. The 'association' felt that "...the product did not promote the growth and vitality of the industry. COPY II PC is used largely to enable people to illegally duplicate software". Obviously, this 'association' was made up of 'copy-protected' software vendors!

The last company I'd like to talk about is Quaid Software Limited of Toronto, Ontario. Quaid produces a product called COPYWRITE. Although totally menu driven, COPYWRITE works much in the same manner COPY II PC does. The authors of COPYWRITE have addressed the backing up of utility and business type software only. COPYWRITE has the ability to copy disks protected by Vault Corporation's PROLOK laser hole protection scheme, as well as most of the other 'software' methods. Because of this unique ability, Vault Corporation has filed suit against Quaid Software, with the objective of getting this copying ability eliminated. COPYWRITE also comes with two additional programs called RAMKEY, and UNGUARD. These programs function in the same way Central Point Software's NOKEY and NOGUARD work.

There are two (and probably a lot more) success stories that immediately come to mind when I think of software vendors that don't appear to be caught up in this war of bits and bytes. First, there is Software Toolworks of Sherman Oaks, California. This company has been producing high quality, low cost software for over five years without the need for copy protection. They have over 50 products that sell for less than \$60, and each comes with a

money back guarantee! The second company I'm thinking about is Borland International. I'm sure everyone is familiar with, or has heard of their Turbo Pascal. The question that needs to be asked here is, how can these companies succeed without copy protecting their software while others can't?

Fortunately, software manufacturers are starting to take the hint, and eliminating copy protection from their products, or making them available without copy protection for a higher price. For the time being, however, the need exists to be able to backup your copy protected software investment, and the three products reviewed (none of which, by the way, are copy protected), will do the job for you, most efficiently.

Sources

UNlock	
Albums A, B, & C	\$49.95 ea.
Album D	\$74.95
TranSec Systems Inc.	
1802-200 North University Drive	
Mercede Executive Park	
Plantation, FL 33322	
(305) 474-7548	
COPY II PC	\$39.95
Central Point Software Inc.	
9700 SW Capitol Hwy. #100	
Portland, OR 97219	
(503) 244-5782	

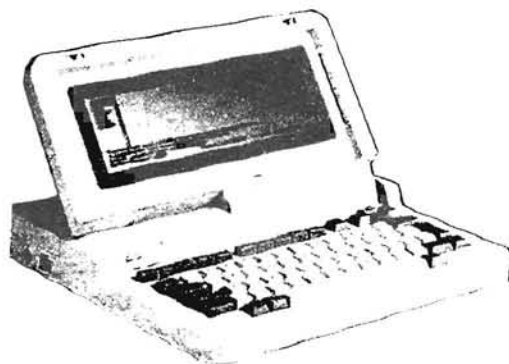
COPYWRITE \$50.00
 Quaid Software Limited
 45 Charles Street East Third Floor
 Toronto, Ontario
 CANADA M4Y 1S2
 (416) 961-8243



MOVING?



Please let us know 8 weeks in advance so you won't miss a single issue of REMark!



SAVE ON LAPTOP RAM 32K RAM for the ZP-150

American Cryptronics, in our third year as a supplier of after market low-power memory for portable computers, now has available 32K RAM modules designed specifically for the Zenith ZP-150 laptop. These user installed modules plug into existing sockets in the computer. The basic 32K machine can accept from one to twelve modules providing a maximum of 416K of on-line, non-volatile memory.

Featuring low-power CMOS static RAM, these modules are backed by a one year warranty. Since your satisfaction is our goal, we have a 30 day full refund return policy.

Manufactured to the highest standards and 100% factory tested, these modules represent significant savings. **Priced at: \$99 for 1 or 2**

\$89 for 3 to 9
\$85 for 10 or more

Illustrated step-by-step instructions are included.

TO ORDER
 CALL 714-540-1174
 or WRITE

M/C, VISA, CHECK or MONEY ORDER
 CALIF. RESIDENTS ADD 6% Sales Tax

SHIPPING-ADD
 UPS Ground \$1.50
 2nd Day Air \$4.00
 Next Day Air \$12.00

Zenith is a registered trademark of Zenith Data Systems Corporation.



(Formerly Cryptronics, Inc.)
 1580 Corporate Drive, Suite 123
 Costa Mesa, California 92626
 (714) 540-1174

More Megabytes for the Buck!

Move up to Winchester Performance for Only \$895

Here's the ultimate solution to your H/Z89 or H8 data storage problems, a Winchester Hard Disk. No other single upgrade for your system can deliver such a performance leap.

- Super Fast Disk Drive Access Speeds
- Have All of Your Data On-Line Simultaneously
- Can Eliminate Hundreds of Floppy Disks
- Allows Better Organization of Your Data
- Better Data Integrity than Floppy Disk Storage



System #1 - 10 Megabytes with Internal Mounting \$895

- The Very Latest 3.5 Inch Low Power Technology
- Mounts in the Front of Your H/Z89 Cabinet
- Ideal for H/Z89's that are Frequently Transported
- Additional Drives can be Attached with an External Cabinet
- This Mounting cannot be Combined with our Interactive Graphics Controller Product

System #2 - 10 Megabytes with External Cabinet \$995

- Super Fast 5.25 Inch Half Height Drive
- Elegantly Mounted in an External Color-Matched Cabinet
- Cabinet Features Room for a Second Drive
- Compatible with the Interactive Graphics Controller

System #3 - 20 Megabytes with External Cabinet \$1095

- Super Fast 5.25 Inch Half Height Drive
- Elegantly Mounted in an External Color-Matched Cabinet
- Cabinet Features Room for a Second Drive
- Compatible with the Interactive Graphics Controller



All Three Systems Feature:

- Up to 16 Data Partitions for HDOS, CP/M, or Both
- Cold Boots HDOS and Heath CP/M (Requires MTR-90 ROM)
- Expandable to 3 Hard Disk Drives and 4 Floppy Drives
- Full ECC Error Correction for Very High Data Integrity
- 2 Centronics Parallel Printer Ports
- Independent Power Supply
- Installs Professionally and Easily
- Uses Only One Bus Slot (H/Z89 Left Side)
- Six Month Limited Warranty on All Components

These systems include a Z37 compatible floppy disk drive controller and software for no extra charge. Just plug in your existing drives, or order a drive to mount in the extra space available in our external cabinet and you'll have double density performance for your floppy drives!

System Expansion Options

- Half Height 40 Track Floppy Drive \$195
- Half Height 80 Track Floppy Drive \$225
- Additional 10 Megabyte Hard Disk Drive..... \$445
- Additional 20 Megabyte Hard Disk Drive..... \$495
- Additional External Cabinet with Power Supply . \$195



"Support is the most important feature."

17000 Dallas Parkway, Suite #207, Dallas, TX 75248, (214) 380-6187

Our engineers would love to answer all of your questions about our products. Please give us a call to check current prices. Please include \$10 for UPS shipping and specify H/Z 89 or H8. Texas residents please add 6% sales tax.



REMark: May 1986, N. B. Day

Dear HUG:

After reading the article describing the solution to the Z-100 printer hang-up problem by N. B. Day (REMark May 1986), I immediately tried it. I wrote the program in C and it worked fine with a half dozen printer on/off tries.

But when trying it immediately after booting my Z-100, and before turning the printer on, it would not work. I, therefore, recoded my program to AND in both low order bits (BUSY and NOT ACKNLG) and printed out the result. After some investigation, it seems that all printer/computer combinations do not work the same. In fact, my OKIDATA 92 would usually give me a 00 for the two low order bits input from port 0xE2 upon initial computer power up. After the printer had been turned on and data fed to it, it would then show a 01, 11, or occasionally 00 when the printer was turned off again. But it always showed a 10 when the printer was on and ready. Therefore, I changed my program so that the input word was ANDed with 2 (10 binary). It worked every time.

Although Mr. Day gave an excellent explanation of the workings of the PIA and the pull-up resistors, it seems that some printers may not play by the rules. Apparently, my printer randomly holds these signals down when powered down. In fact, in some cases when my printer was powered down, it would show 00 for 1 to 20 seconds and then change to 01. It had no rhyme or reason as to how it would act next.

The moral being, if your printer does not work with Mr. Day's program as is, try ANDing the input with 3 (11 binary) and printing out the results. Your printer may have another combination of conditions.

Sincerely,

Ivan D. Harnage
President, C & H Consultants
204 Alexander Drive
Warner Robins, GA 31093

ACCELERATE 8/16 And The NEC V20

Dear HUG:

I recently purchased this package, hoping to run some of my old H-89 MBASIC programs on the H-160.

The good news is that this is a well documented program that does everything claimed, as far as I can tell. The V20 chip works fine in the H-160.

The bad news is that MBASIC contains Z-80 opcodes! This prevents it, and programs written using it, from running. My MBASIC is version 2.3.02. The offending opcodes seem to be the relative jump instructions. Perhaps other versions will run.

I suggest potential users check their H-89 programs using DDT before ordering this package. If your Assembler listing shows

“??=nn”, this means the opcode nn is not a valid 8080 opcode, and you cannot use ACCELERATE 8/16 or the NEC V20 chip to run it on a PC-compatible.

John Bohannon
1304 Pecan
Ponca City, OK 74604

April Issue Of REMark

Dear HUG:

Congratulations on the April issue of REMark. This was an outstandingly useful issue! I am a Z-100 owner and I have been following the ZPC series. ZHS is the greatest. I had it built and running in a few hours, and was running an unprotected copy of 1-2-3 within a week! The articles on V-20 were really helpful, too.

A few comments:

1. The 'street price' of 8 MHz V-20s in Silicon Valley is \$18.00, quantity 1. I ran out and got one, and I find a 10% improvement on real 1-2-3 applications. Some things, like the memory test in the monitor, run as much as 25% faster.
2. The source listing of KEYINT.ASM given in the ZHS article will not run through EXE2BIN as written, the Stack declaration must be removed. Once this is done, everything works. The reason is that EXE2BIN only works within a single segment and the stack is assigned automatically. Also, the output BIN file must be renamed KEYINT.COM.
3. I found a few compatibility problems running 1-2-3:
 - The cursor stays in the upper left of the control panel when in EDIT and doesn't follow the cursor position, which works otherwise.
 - When loading very large WKS files, garbage appears on the screen, this goes away, except around the borders when loading is completed. Everything works until a CALC is done, at which point it blows up altogether.
 - Screen graphics don't work with the standard Lotus graphics adapter driver.

Let's hear more about ZHS and ZPC2, keep up the good work.

Sincerely,

Seth Neumann
2712 Katrina Way
Mountain View, CA 94040

Looking For A Music Editor

Dear HUG:

Having tried for months to find the right software, the time has come to present my problem to the ever-ingenuous REMark readers.

I am a music teacher and wish to locate a music editing program. The only commercial programs available either do too little (elementary level music education) or too much (interface with electronic keyboards, act as synthesizers, etc.).

The ideal program for me would work right from the computer keyboard. I would be able to enter a musical staff, clef signs,

meter and key signatures, note heads for pitch and stems/flags to show duration. There should be the capability to enter the standard musical words and symbols which show loudness, mood, inflection, etc. And transportation from one key to another would be terrific. Displaying four parts at once is enough for my purposes.

What I do not need to do is create any musical sounds with the program or have it edit direct from the piano keyboard.

My equipment is a ZF-151-52 computer, a ZVM-135 hi-resolution color monitor, and a Panasonic KX-P1091 dot matrix printer.

Do any of you enthusiasts out there know of such a program? I'd love to hear from you?

Robert Shamo
18W 665 Thirteenth Street
Lombard, IL 60148

Installing ZRAM-768K

Dear Mr. Rudy:

We received a copy of the letter you sent to HUG concerning your problems with installing our ZRAM-768K modification kit and UCI's 8 MHz upgrade in your Z-100. We regret the difficulty you experienced getting all the modifications to work. We would, however, like to respond with a clarification.

Since Wideman first distributed his public domain instructions for the 768K RAM upgrade, Tex-Matics has sold hundreds of parts kits worldwide. First introduced at \$219.95, they sell for \$129.95 with a money-back guarantee at this writing. In addition, UCI includes instructions to contact them if there were any problems using their speed upgrade. Tex-Matics supports UCI's money-back guarantee if a customer's machine simply won't run using the upgrade. UCI also cautions that additional chip replacements might be necessary. UCI provides a consultation line, the one you called to get your problem resolved. That aid must be effective, since we've not had a single UCI customer request a refund.

As for Wideman's 768K upgrade, and its compatibility to the UCI upgrade, I believe you are now using both successfully. I regret that we could not resolve the problem over the phone. I believe we did offer to check out the problem if you would send your motherboard to us. Let me add that the only returns of the 768K kit have been by people who ordered it not knowing they could not use it. The few problem calls we've had (disregarding installation errors) generally concerned bad or damaged parts (which we replaced immediately), or were related to misunderstandings as to what programs and operating systems were excluded by Wideman's change, namely the Z-100 ROM monitor RAM test routine, and Watzman's CPM-Plus. So far, it has been compatible with all other Z-100 hardware, including DEL's Gemini board, and we expect no problems with UCI's emulator.

No one likes to have his computer down. No one like problems that don't have cook-book fixes. And no one likes to make long distance calls for assistance and not get the answers he needs. I regret the problems you had, and wish that I could have given you the resolution. I hope our referral to UCI won't be construed as "passing the buck", as our intention is to provide you with prompt assistance from the right expert.

We appreciate each and every customer, and promise to provide the best support we can with each product. We make it a point to

deal with suppliers that agree with that philosophy. Please let me know if we can be of any further assistance.

Sincerely,

Kenneth A. Patrick
Tex-Matics Micro Systems
3059 W. 15th Street, Suite 100
Plano, TX 75075

Use Your HUG ID To Eat At The "Gaslight Club"

Dear HUG:

I thought you might be interested in an incident that happened to me the other day. It doesn't have anything to do with computers, but I think you'll find it amusing.

Two weekends past, I had to go to Chicago's O'Hare Airport to pick up a relative returning from Spain. I asked along a friend to give me a hand, and promised a supper in return for his aid. When we arrived at the "World's Busiest Airport", I noted a long delay before the flight was returning and decided to have supper there. Because the airport itself only has "coffee shop" like restaurants, my friend and I crossed over to the Hilton O'Hare and up to the lobby.

There, I noted the "Gaslight Club", and asked the receptionist outside if there were any reservations open. She answered yes and I requested a table for two.

"Do you have a key card?" she asked, taking me by surprise.

"Ah, no, I don't," I answered.

"Are you a guest at the hotel?" She tried again.

"Ah, no, I'm not," I replied again.

"Well, are you *visiting* someone at the hotel?" she inquired.

"Ah, no, sorry," I grinned back.

"Do you have anything?" she kidded in mock desperation.

I remembered my mail earlier that afternoon. "I'm a member of the Heath Users' Group!" I announced triumphantly.

"Good enough!" she laughed back and we were in! It may have cost quite a bit, but we had a damn fine supper, thanks to Heath. Just two questions. How come I didn't get my 20% discount? And will my HUG membership work at the local Playboy club as easily?

Now for serious stuff. I have both a Zenith 90 computer and an Apple Macintosh. For some time, I was using two printers, a Centronics for the Zenith and the ImageWriter (I or II). I decided to sell the Centronics and use the ImageWriter as my system printer. The question was the cable hookup. The ImageWriter I used a standard DB-25 connector while the ImageWriter II used the Apple circular ("DIN" style) eight pin connector. With my Zenith configured for 9600 baud, 8-bits, no parity with the controlling pin as pin 20, set to HIGH for a go to transmit, I decided to hook it up to the ImageWriter II with an "Apple-to-ImageWriter II" cable (Apple part number A9C0313). I figured I could always check the manual for the wiring pin-outs and cut off the DB-25 and, using the almost impossible to get 8-pin end, adding my own DB-25 which I would wire myself.

To my shock and surprise, the ImageWriter II, as it comes from the factory, is already adjusted for the Macintosh, also just hap-

Continued on Page 82

For Experts Only

A Review Of The Seidl Make Utility

Pat Swayne

HUC Software Engineer

If you are a software author for MS-DOS systems who writes complex programs involving several modules that are assembled and/or compiled separately and then linked together to produce the final programs, your work could be made easier if you had the Seidl Make Utility. The Seidl Make Utility (abbreviated SMK by its author) can direct the assembling and/or compiling and linking of your modules for you, and make sure that it is done in the right way and in the correct order. As you develop your software, making changes here and there and recompiling, SMK can save time by ensuring that only those modules affected by your changes are recompiled.

Basically, SMK uses the time and date stamp on your various module files to determine which ones have been changed. There are other utilities for compiling programs that do this, such as the MAKE utility included with the MS-DOS version 3 Programmer's Utility Pack, but SMK is much more elaborate. SMK can make sure that modules that have not changed themselves, but are dependent on modules that have changed, are properly recompiled. It is directed by a description file that you produce using what the author calls the SMK dependency definition language, or DDL. DDL is almost a programming language in itself, and there is quite a bit you can do with it.

The main element of DDL is the MAKE statement. Each MAKE statement describes a file that is to be made, the files (modules) required to make it, and the commands required to construct the file. A file made in a MAKE statement can be used in another MAKE statement. The file made in the second MAKE statement is said to be dependent on anything made in previous MAKE statements. As SMK processes MAKE statements, it alters the date/time stamp on each file processed to the current date/time. If SMK is rerun on the same DDL file, and no dates and times have been altered since the last run, nothing is recompiled. However, if a file has been altered, it is recompiled, and any files dependent on it are also recompiled.

Other statements in DDL support the MAKE statement. For example, a FOR statement allows you to compile several files with one MAKE statement. There are 8 commands in DDL altogether: MAKE, FOR, PROLOG, EPILOG, COMMENT, CONST, MACRO, and INCLUDE. The names of the commands are descriptive of what they do.

There are two special utilities included with SMK that can be included as commands in MAKE statements to add to its usefulness. One is SMKFMt, which allows you to insert information, such as the date or a serial number, into a source file each time SMK is run. The other utility, SMKDATE, is used to change the date/time on a file. Normally, SMK changes the date on each file that is remade to the current date/time, but SMKDATE can be used to change the date/time on files that are not remade. This is important when a file that others are dependent on is changed in a trivial way (a comment added, etc.), and you don't want everything recompiled the next time SMK is used because of the trivial

change. If SMK is run using a special switch, it uses SMKDATE to change the date/time on all files without compiling anything.

What SMK actually does when it runs, in addition to checking the date/time on specified files and updating the date/time as required, is to create a batch file which in turn is used to command the compilation of your programs. A master batch file, called SMK.BAT, runs the whole show, by first running the actual SMK program (SMK1.EXE), and then running the batch file that SMK creates. A single command line is all that you have to type to accomplish any compilation, no matter how complex, once you have set up your DDL file.

SMK can be used for other jobs besides compiling programs. Two other jobs that are explained in the manual are maintaining libraries and automated file backup (as when using a RAM disk for editing and compiling, to get things back to a real disk after the work is done). It will ensure that only those files that have been changed are copied to the back-up disk.

SMK is a generic MS-DOS system that will run on any MS-DOS or PC-DOS computer, as long as you have DOS version 2 or above. It runs fast and can process a fairly large DDL file in a matter of seconds. It comes on one 5.25" disk and is packaged with a 48 page full size (not IBM size) manual in a loose leaf binder. The manual is professionally typeset and does not appear to have any typographical errors. It is, however, the reason why I called this article "For Experts Only". It is definitely not for beginners. But if you have gotten to the point where you can use a program like SMK, then you must know how to read (it's amazing how many people can sound the words but don't really know how to read!), and if you can read, you can use SMK by following the manual. One possible area of minor confusion is where the manual states that SMK1.EXE produces a "do file". It actually produces a batch file called SMK2.BAT, and has nothing to do with the DO program included with the MS-DOS Version 2 Programmer's Utility Pack.

The Seidl Make Utility disk comes in a sealed plastic pouch attached to a card, the back of which is covered with the kind of gobble-de-goop one usually finds in the software license agreements of larger software houses. The agreement says that you may make only one back-up copy, but the program is not copy protected, and I'm sure that you could get away with making two copies, or putting it on your hard disk in addition to making a back-up copy.

The agreement says that the one copy you make must be for back-up purposes only. I guess that means that if you copy it to your hard disk (which would be for USE, not back-up purposes), you would be breaking the agreement. That just shows you what nonsense most software license agreements really are. Just ignore them, and maybe they will go away.

The Seidl Make Utility is available for \$99.95 from Seidl Computer Engineering, 1163 Ogden Avenue, Suite 705-171, Naperville, IL 60540, phone (312) 983-5477. ✱

More RAM And A RAMDISK With MegaRAM-150 For The H/Z-100 PCs

Richard L. Mueller, Ph.D.
11890-65th Avenue N.
Maple Grove, MN 55369

Do you have a need for more than 320K of RAM (Random Access Memory) memory on your H/Z-100 PC series microcomputer in order to run large programs or applications? Or do you need more memory just to run the "average size" programs after installing one or more of the "memory resident" packages such as Sidekick, Superkey, Genie, etc.? Or do you have a need for a RAMDISK capability for faster file operations (which implies faster program/application execution), such as loading of very frequently called/used programs/applications, sorting of data, manipulating data files, etc?

My answer was "yes". If your answer is "yes" to one or more of the questions above, then "stay tuned" to this article for information on a product that I installed in my Z-160 and that I highly recommend: MegaRAM-150 from FBE Research Company, Inc. This product has been "advertised" in both BUSS and H-SCOOP. Before I get into some of the details on the MegaRAM-150 product, let me cover some background information first on memory limits, memory chip sizes, RAMDISKS in general, etc.

Overview Of Additional Memory

In the H/Z-100 PC series, 64K RAM memory chips were/are used on the main memory board. There are sockets for five (5) banks of these memory chips, giving a total of 320K of memory when all the banks are filled. Each "bank" consists of nine (9) RAM chips, 8 for the 8 bits in a byte and one for parity. So for 320K, you will need forty-five (45) 64K memory chips (this is before the availability of the 256K chips).

Again, before the availability of the 256K memory chips, to go beyond the 320K of RAM memory, one had to install a second memory board in the microcomputer. This additional board could also contain up to 320K, bringing the total to 640K. This is the magic number that you see in all the literature as the maximum one can have in his/her micro. In fact, there are references in some of the literature (manuals) that imply that this figure is the maximum supported by MS-DOS (PC-DOS). However, this is not entirely correct; MS-DOS (PC-DOS) can support more than 640K and I will explain this in detail shortly. MegaRAM-150 lets you access more than 640K; 704K to be exact.

Memory Size	Bank 0	Bank 1	Bank 2	Bank 3	Bank 4
256K	256K	----	----	----	----
320K	256K	64K	----	----	----
384K	256K	64K	64K	----	----
448K	256K	64K	64K	64K	----
512K	256K	256K	----	----	----
512K	256K	64K	64K	64K	64K
576K	256K	256K	64K	----	----
640K	256K	256K	64K	64K	----
640K	256K	256K	----	----	256K *

* Only 128K utilized

Table 1
From Software Wizardry and RAM PAL-150

256K Memory Chips

Today there is a way to upgrade your memory without adding an optional memory board and taking up a valuable expansion slot in your microcomputer. This involves changing the Programmable Array Logic (PAL) chip at U455 on your main memory board with a new PAL chip that allows the system to recognize/access/utilize 256K memory chips along with the 64K memory chips. That's right; you can mix the memory chips. One bank can have 256K chips while another can utilize 64K chips. See Tables 1 and 2 for sample configurations depending on the PAL chip used. This implies that some of the 64K chips have to be removed and "thrown away" (really put aside for spares). These new PAL chips are available from several sources including, but not limited to, FBE Research with ZP640 and ZP704 (actually these PAL chips are available through vendors, such as Quickdata and RAM Technology), FBE Research with MegaRAM-150 (which is a special PAL chip as we will see shortly), Software Wizardry with RAM PAL-150, and KEA Systems Ltd. with ZPAL.

Memory Size	Bank 0	Bank 1	Bank 2	Bank 3	Bank 4
384K	64K	64K	256K	----	----
448K	64K	64K	256K	64K	----
640K	64K	64K	256K	256K	----
704K	64K	64K	256K	256K	64K

Table 2
FBE Research with ZP640 and ZP704

Now let's look at this "640K" limit in more detail. The INTEL 8086/8088 microprocessor chip (used in the IBM PC and Zenith H/Z-100 PCs) has 20 bits of addressing, which allows addressing of 1 megabyte of memory. However, part of the 1 megabyte is used by the Monitor ROM, by buffers for the video modes (monochrome and color), etc. The following is a memory map taken from the Zenith Programmer's Reference Manual for the H/Z-100 PCs:

Type Of Memory	Hexadecimal Address	Decimal Address	
RAM Memory	00000-9FFFF	0000000-0655359	640K
Reserved	A0000-AFFFF	0655360-0720895	64K
Monochrome Video	B0000-B3FFF	0720896-0737279	16K
Reserved	B4000-B7FFF	0737280-0753663	16K
Color Graphics	B8000-BBFFF	0753664-0770047	16K
Reserved	BC000-BFFFF	0770048-0786431	16K
Bit-Mapped Graphics	C0000-EFFFF	0786432-0983039	192K*
MFM-150 Monitor RAM	F0000-F3FFF	0983040-0999423	16K
Winchester Buffer	F4000-F7FFF	0999424-1015807	16K*
MFM-150 ROM	F8000-FFFFF	1015808-1048575	32K

This adds up to 1024K or 1 Megabyte (1MB)

* Only if the Z-319 Bit-Mapped Graphics Board is installed. If Z-319 is NOT installed, by default, the address range starting with C0000 is used by the Winchester Controller ROM. When Z-319 is installed, the Winchester Controller ROM is changed to start at F4000.

Table 3
Memory Map

As you can see there are some "wholes"; that is, there are some "chunks" of memory that are not used, but listed as "reserved" for future expansion. Please note that the 64K block immediately following the 640K (00000-9FFFF) area is a 64K block (A0000-AFFFF) that is not used except when the system has an Enhanced Graphics Adaptor (EGA) board installed.

If you do not have or use an EGA board, then this 64K block can be used to extend your 640K limit to 704K. MS-DOS (PC-DOS) will recognize this additional RAM and use it if the proper PAL is installed and the memory switches on the CPU board are set correctly. Also the "JUMPER" at J401 on the memory board has to be changed when adding 256K chips and going beyond 320K. See the memory switch settings in Table 5 which is in the section on the MegaRAM-150 product.

RAM Disks

What is a RAMDISK? It simply means that a chunk of your RAM is used to emulate a diskette drive. However, unlike a true floppy disk drive or harddisk drive, the RAMDISK is extremely FAST. The reason is simply that you are transferring information from one section of RAM to another without going through some mechanical device which adds considerable delay. Unfortunately, RAM is not generally backed up with battery power, so when power is turned off to the microcomputer, the contents of the RAMDISK are lost.

RAMDISKs are also referred to as "Electronic Disks". I prefer to just call them RAMDISKs. RAMDISKs are made to look just like a floppy disk drive which means they have File Allocation Tables (FAT) and Directories. They can even have a pseudo "boot" record to allow adding the "system" (IO.SYS, MSDOS.SYS, and COMMAND.COM), if you have a reason to have that on your RAMDISK.

RAMDISKs came into being when MS-DOS V2 was released with the capability of adding your own "device drivers". I don't want

to get into a discussion of "device drivers" here because that's an article in itself. However, just briefly, "device drivers" are nothing more than code that takes care of initializing and reading/writing whatever new device you are adding to your system and that includes RAMDISKs. RAMDISKs are assigned a drive letter just as your floppy disk drives or harddisks. You can read and write to RAMDISKs just as you would to any floppy disk or harddisk.

As I said earlier, RAMDISKs up to this time have taken a chunk out of your valuable RAM space. However, with MegaRAM-150 that is not so.

Main Memory

Memory Size	Bank 0	Bank 1	Bank 2
256K	256K	----	----
320K	256K	64K	----
512K	256K	256K	----
576K	256K	256K	64K
704K	256K	256K	256K *

* The upper 64K block is not used as part of main memory RAM.

Overlay (or Paged) Memory

Memory Size	Bank 3	Bank 4
256K	256K	----
320K	256K	64K
512K	256K	256K

Table 4
From MegaRAM-150 Users' Guide

MegaRAM-150

Now let's take a closer look at MegaRAM-150, the main purpose of this article. MegaRAM-150 not only allows you to go to 704K of RAM (main memory), but also allows you to add up to 512K of RAMDISK which is over and above your 704K main memory RAM. I installed MegaRAM-150 and the full amount for main memory (704K) and the full amount of memory for the RAMDISK (512K) and I'm quite pleased with all this memory.

MegaRAM-150 itself just consists of the MR-150 PAL chip and a small Jumper wire. The product does not include the 256K memory chips. This is generally true of all PAL products. The memory chips have to be purchased separately. A number of vendors have been advertising their 256K chip prices in BUSS, H-SCOOP, as well as other microcomputer magazines.

When the other PAL products are installed, the Jumper at J401 has to be set in position 2 to allow more than 320K of memory. However, the MegaRAM-150 product actually does not use this Jumper; the instructions state to either remove the Jumper Cap or to put it on the leftmost pin of J401 only to get it out of the way. The small Jumper wire (E-Z-Hook) included with MegaRAM-150 has "hooks" on both ends. One end is connected to the rightmost pin of J401 and pin 10 of chip U458. I have connected the Jumper this way, but Henry Fale pointed out in a recent issue of H-SCOOP that it might be better to solder the Jumper for a more permanent connection. The choice is left to you.

As I said before, the main memory can go up to 704K by installing 256K memory chips in banks 0, 1, and 2. The upper 64K block of bank 2 is not used since this memory B0000-BFFFF is used partly by the monochrome video board. So we can only go to AFFFF

which is 704K. In addition, you can go to 512K of RAMDISK by installing 256K memory chips in banks 4 and 5. See Table 4. Also see Table 5 for memory switch settings on the CPU board.

Memory Switch Positions					
Memory Size	0	1	2	3	4
256K	right	left	left	right	right
320K	right	right	right	left	right
512K	right	left	left	left	right
576K	right	right	right	right	left
704K	right	right	left	right	left

Table 5
From MegaRAM-150 Users' Guide

Once the hardware is installed, the next thing that needs installation is the software. Just as with other RAMDISKS, MegaRAM-150 requires a "device driver". The device driver MR150.DVD comes with the MegaRAM-150 product and should be copied to your bootable system disk. In addition, file CONFIG.SYS should also contain the following entry:

```
DEVICE = MR150.DVD x      where x is as follows:
```

The x parameter represents the size of RAMDISK memory; A=256K, B=320K, and C=512K. The distribution disk contains file CONFIG.SYS with the above entry in it set for 512K (C). If you do not currently have a CONFIG.SYS file on your system bootable disk, then you can just copy this file to it. Otherwise, if you do have a CONFIG.SYS file, just add the above entry. Reboot your system and you should get a message that states that the MegaRAM-150 RAMDISK is installed:

```
xxx KByte MegaRAM-150 Memory Disk Installed as Drive y:
```

where xxx is either 256K, 320K, or 512K (as specified on your CONFIG.SYS entry: A, B, or C, respectively) and y is the next available disk device letter (e.g., as in my case where I only have two floppy disks, the letter is C).

You can perform operations on your RAMDISK just like your other disk drives. Do a "DIR C:". You should get the message "File Not Found" which means no files currently exist on your RAMDISK. Please note that you have a Label listed for this RAMDISK which is generally not done with other RAMDISKS: "MR-150 DISK." Try copying any file to your new RAMDISK and do a DIR. Same as any other disk, right?

After installing MegaRAM-150, I did some quick testing of loading programs from an actual floppy disk versus my RAMDISK and the MegaRAM-150 is extremely fast. A frequently used program (125K) that I have, loaded from a floppy disk in 18 seconds. From the MegaRAM-150 RAMDISK it took just 2 seconds.

Now, let me spend some time on the internal workings of MegaRAM-150. First, some preliminary details. MegaRAM-150 uses 512 bytes per "sector" just as the 5-1/4" inch disk drives. However, MegaRAM-150 only has 1 sector per allocation unit rather than the 2 used by the regular disk drives. This, I feel, better utilizes the available memory and less is "wasted". Only 1 File Allocation Table (FAT) is used rather than 2 which is common for the regular disk drives. The MegaRAM-150 RAMDISK is defined to be an 8-sector per track device. All this is to make a RAMDISK look and function like any other disk drive.

For the next part of this discussion I will be referring to Figure 1 to try to explain how MegaRAM-150 utilizes the memory in banks 3 and 4. Take a look at Figure 1 to get a feeling of what I will be talking about next. There has been much talk recently about the

Lotus/Intel/Microsoft Expanded Memory Specification (EMS) board. However, MegaRAM-150 handles the extra memory in a different manner. The EMS board allows users to access extended memory via one of the "Reserved" blocks of memory. Just how the EMS board works in detail, is a topic for another article.

MegaRAM-150, on the other hand, does not use any of the Reserved blocks of memory as the EMS board does to access the extended memory. However, MegaRAM-150 uses a paging technique (or overlaying technique as FBE refers to it) to "switch in" (as I call it) the RAMDISK memory into the first 704K area as shown in my Figure to be accessed by the MegaRAM-150's Device Driver rather than by the user directly. The MegaRAM-150 product is implemented as a RAMDISK while EMS is implemented as an extension of user RAM where a user's program must ask for specific blocks to be "paged in" to be accessed. The MegaRAM-150 implementation frees the user from this task by treating the extra memory as a RAMDISK.

Now to the details. Each 256K bank is divided into four (4) 64K chunks which I will call "blocks". The banks will be labeled 0 through 4 and the blocks within a bank a-d. So, for the first block in bank 0, I will refer to that as block 0-a, and so forth. The main memory is contained in banks 0, 1, and 2; blocks 0-a through 2-c. Remember, the last block in bank 2 (2-d) is not used as main memory since it is the monochrome video buffer.

Since we cannot use block 2-d, this causes some peculiarities in the "switching in" of the extended memory in banks 3 and 4. What MegaRAM-150 does is to switch in (page in) banks 3 and 4 and map them into the upper portion of the lower three banks (blocks 0-d through 2-c or addresses 30000H through AFFFFH). From here the MegaRAM-150 Device Driver can access the contents of this extended memory or store information into this extended memory by referencing the addresses above.

When this switching takes place, a block "a" maps into a block "a", "b" into "b", etc. Since we want to map the two extended memory banks into the upper-most part of the lower 3 banks, the "d" blocks are mapped out of sequence. Block 3-d maps into 0-d instead of 1-d, and block 4-d maps into block 1-d instead of 2-d which we cannot use. The rest of the blocks in banks 3 and 4 map into their corresponding blocks in banks 1 and 2. See Figure 1 for this mapping correspondence.

This "out-of-sequence" mapping creates no problem since this mapping always takes place each time you read or write this extended memory. This memory "switching" is controlled via bit 3 of the memory parity I/O port: 100H. Setting bit 3 to 1 will switch in the extended memory and this memory can be referenced using addresses 30000H to AFFFFH. Setting bit 3 to 0 will switch the extended memory "back out".

Up to this point, I failed to mention one very important fact: When the extended memory (banks 3 and 4) is switched in, the original contents of banks 0 through 2 (blocks 0-d through 2-c) are left intact. No memory contents are lost in this switching or swapping operation.

What the MegaRAM-150 Device Driver does when a request is made to read or write this extended memory, is to first send out a function byte on I/O port 100H with bit 3 set to switch in the extended memory. On a READ operation, one sector at a time is read into a buffer in the Device Driver's memory space, the extended memory is switched out, and the contents of the buffer are transferred to the requesting programs memory space. If

more than one sector was to be read in, then the above process is repeated starting with switching in the extended memory.

The process is similar for WRITING. First, a sector of information is read from the programs memory space into the Device Driver's buffer, the extended memory is switched in, the data from the Driver's buffer is written to the extended memory, then the extended memory is switched out. The process is repeated if more than 1 sector needs to be written.

Because of this double moving of data in memory, this RAMDISK is somewhat slower than a RAMDISK that would occupy some of the Main Memory space. However, it still is extremely fast.

That's basically how MegaRAM-150 works. Some additional information is provided in the User's Guide for the MegaRAM-150 product. If you need additional Main Memory and/or need a RAMDISK for better operations, I highly recommend MegaRAM-150. MegaRAM-150 will allow you to install the maximum Main Memory possible and a maximum of 512K RAMDISK without adding any addition memory board.

Good luck and happy computing . . .

```
*****
* MegaRAM-150 *
*****
```

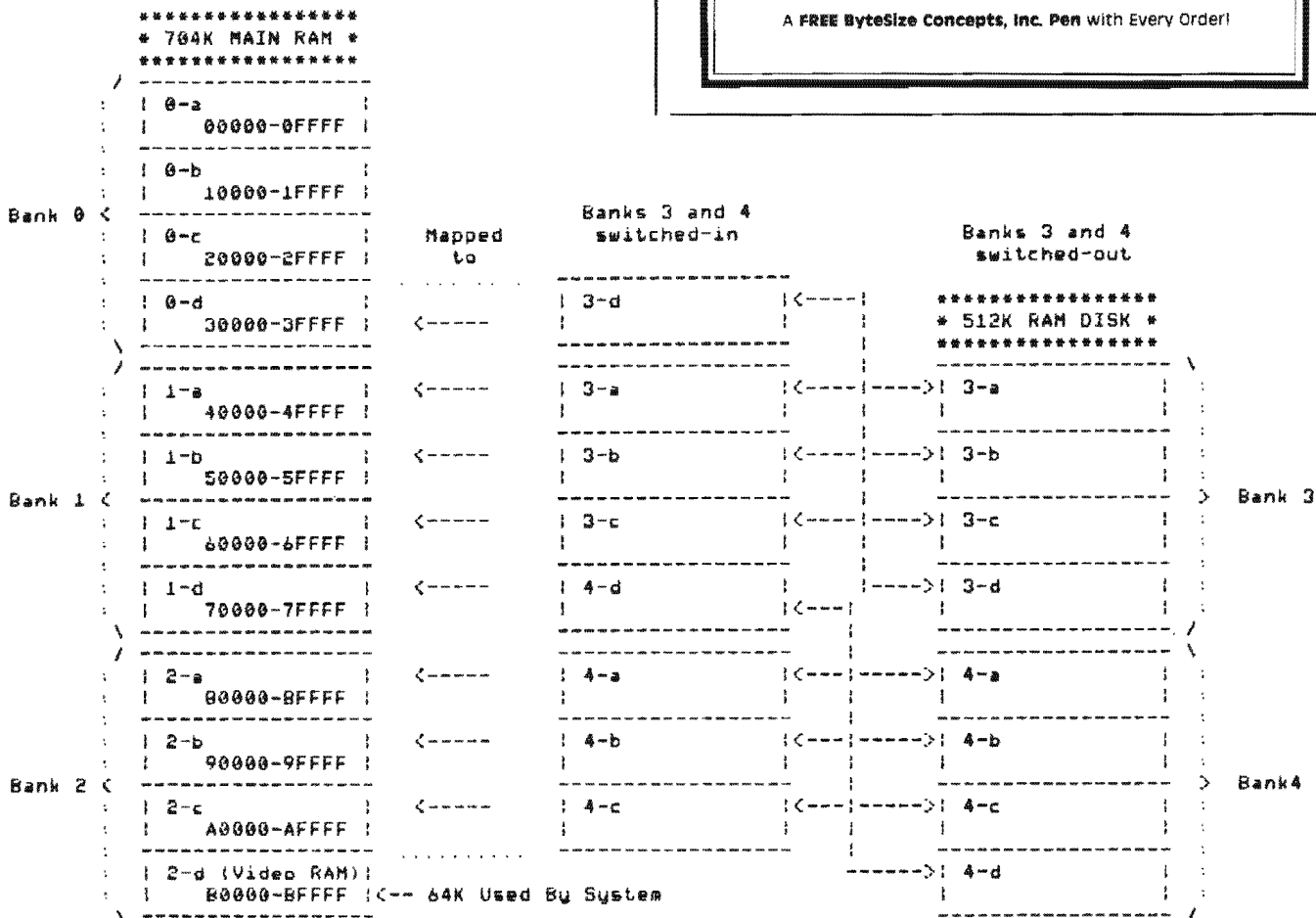


Figure 1



THE HOME SERIES
Now permanently reduced
each just **\$39⁹⁵**

- **Home Budget Manager**
complete checkbook management
 - **Home Data Manager**
easy to use database for the home
 - **Home Mechanic**
auto expense log and how-to-fix-it guide on disk!
- Available for H-8, H/Z-89, H/Z-100,
CP/M or ZDOS (MS-DOS)

Also...
ELECTRONIC TYPEWRITER
still only **\$29⁹⁵** now also
available for H/Z-150 series machines
(and Zenith-compatible IBM's!)

Orders should include: \$2.75 shipping & handling
Ohio orders - 6.5% sales tax • State disk format/DOS

ByteSize Concepts, Inc.
18101 Euclid Ave. • Cleveland, OH 44112
(216) 486-1114

A FREE ByteSize Concepts, Inc. Pen with Every Order!

Keep Your H/Z-100 Blinking For MS-DOS 2.0 Or Greater

M. D. Holmberg II
5146C Gunn Avenue
Scott AFB, IL 62225

Soon after I purchased my Z-100, I was convinced that it was the best computer on the market. But, the hardware, MS-DOS, or CP/M, doesn't seem to automatically handle blinking screen displays as do other inferior 'Z-100 look alikes', not to mention any names. I thought about this project for some time and was sure I could implement a blinking display, without a great cost in speed, and without needing complicated programming every time a blinking character, word, phrase, or line is needed. Solving this problem required several general steps:

1. Devise a resident program to 'wake up' whenever a blinking data element is needed.
2. Handle blinking display writing without slowing down the computer.
3. Figure a way to pass information back and forth between the background program and the foreground program.

The Resident Program (Listing 1)

To devise a program to handle blinking, I had to write a resident program in assembly language to make data items blink on the display whenever needed. To do this, I had to cause my program to activate on the timer interrupt, which occurs every 1/100 of a second. Using the bios interrupt function handled this very nicely. The documentation is available in the 'MS-DOS Programmers Utility Pack' and the files 'DEFBIOS.ASM' and 'DEFIPAGE.ASM'. To use this function, you simply call it at runtime and place your interrupt handler in a far procedure; MS-DOS then takes care of 'fixing' the code segment and instruction pointer.

The next problem was deciding how to efficiently write blinking characters to the display screen. My first thought was to use the escape codes to save the cursor position (ESC j), plot a new cursor position (ESC Yxx), write the blinking data with the bios conout function, and then restore the saved cursor position (ESC k). But, I found all this screen writing to be too time consuming, and other programs running in the foreground didn't seem to like all

these escape codes being fed in unexpectedly. My answer was to write directly to video memory. This made for a much more complicated algorithm, but the result was well worth it. I have tested the program with 20+ blinking data items ranging from 1 to 30 characters and the cost in speed isn't noticeable.

Video Memory

Addressing bytes in video memory is a pretty complicated process. It is well documented in the Z-100 technical manual and many, many articles have been written on it in 'REMark'. However, I decided instead of calculating pixel addresses on the display screen to store the offset for each line on the display screen. This only took up about 500 bytes, and the speed advantage was worth it.

Then I had to handle writing the correct characters to the video planes. The best way to do this is to use the system font table. The font table is located in the bios segment and can be accessed through the jump tables. Bios_ctaddr (DEFBIOS.ASM) is a pointer to a table of pointers. In this table, the offset of the font table is the 8th and 9th bytes. The font table is organized in the same way as the ASCII chart, starting with the space (20h) at 9 bytes per character.

So, now we have everything we need to write characters to video memory. When we have a character to write, we find the location of the font table, we find the location in video memory to place the character and write the nine bytes for each character to the video planes.

When And Where To Blink

Now we need some sort of procedure for telling the resident program when and where to blink. For this, I had to set up a data table with information on the data item we wish to flash on the display screen.

The layout of the table is as follows:

1st byte — This is a flag byte. The resident program checks this byte to determine whether or not to activate; in essence it is an on/off switch.

The following items are repeated 100 times, allowing up to 100 blinking data items.

1. (byte) number of characters to blink.
2. (byte) character row on the display screen.
3. (byte) character column on the display screen.
4. (word) segment of data item.
5. (word) offset of data item.

Now, we are able to write data to video memory through pixel addressing and the system font table, and we are able to access a table of data to determine where to find the data to blink, and where to blink it on the video display.

But, we want to access this table when running other programs such as Pascal, BASIC, or Assembler. To do this, I used the bios vector table. Now remember, this table is part of our resident program. In the bios segment are the interrupt vectors for system calls, and according to the 'MS-DOS Programmers Utility Pack', vectors including and above 80h are user definable. So, at vector 81h I installed the address of our table. Each entry in the vector table requires four bytes, so, in another program if we read the four bytes following location 0:81h * 4, we will have the offset and segment of our data table and writing the correct information to that table will cause data to blink on the display screen.

Listing 1

```

;
; title -- Z100 blinker --
; page ,132
;
dosf_outstr equ 09h
dosi_func equ 21h
dosi_termr equ 27h
bif_setint equ 0
vcr equ 0d8h ; video control register
newvid equ 78h ; allow write to video mem
;
ipage segment at 0h
org 14ch
int_utma equ 51h
ipage ends
;
bios segment at 40h
org 23 * 3
bios_intfunc label far
org 32 * 3 + 1
bios_ctaddr label word
bios ends
;
;
;
code segment
assume cs:code, ds:code, es:code, ss:code
org 100h
main. jmp install
;
; system data
;
mystack label word
oldss dw ?
oldsp dw ?
oldds dw ?
oides dw ?
;

```

```

; sign on data
;
msg db 'Z100 Blinker installed ..$'
;
; interrupt data
;
addrtbl dw 0h, 80h, 100h, 180h, 200h, 280h, 300h, 380h, 400h
dw 800h, 880h, 900h, 980h, 0a00h, 0a80h, 0b00h, 0b80h
dw 0c00h, 1000h, 1080h, 1100h, 1180h, 1200h, 1280h
dw 1300h, 1380h, 1400h, 1800h, 1880h, 1900h, 1980h
dw 1a00h, 1a80h, 1b00h, 1b80h, 1c00h, 2000h, 2080h
dw 2100h, 2180h, 2200h, 2280h, 2300h, 2380h, 2400h
dw 2800h, 2880h, 2900h, 2980h, 2a00h, 2a80h, 2b00h
dw 2b80h, 2c00h, 3000h, 3080h, 3100h, 3180h, 3200h
dw 3280h, 3300h, 3380h, 3400h, 3480h, 3880h, 3900h
dw 3980h, 3a00h, 3a80h, 3b00h, 3b80h, 3c00h, 4000h
dw 4080h, 4100h, 4180h, 4200h, 4280h, 4300h, 4380h
dw 4400h, 4800h, 4880h, 4900h, 4980h, 4a00h, 4a80h
dw 4b00h, 4b80h, 4c00h, 5000h, 5080h, 5100h, 5180h
dw 5200h, 5280h, 5300h, 5380h, 5400h, 5800h, 5880h
dw 5900h, 5980h, 5a00h, 5a80h, 5b00h, 5b80h, 5c00h
dw 6000h, 6080h, 6100h, 6180h, 6200h, 6280h, 6300h
dw 6380h, 6400h, 6800h, 6880h, 6900h, 6980h, 6a00h
dw 6a80h, 6b00h, 6b80h, 6c00h, 7000h, 7080h, 7100h
dw 7180h, 7200h, 7280h, 7300h, 7380h, 7400h, 7800h
dw 7880h, 7900h, 7980h, 7a00h, 7a80h, 7b00h, 7b80h
dw 7c00h, 8000h, 8080h, 8100h, 8180h, 8200h, 8280h
dw 8300h, 8380h, 8400h, 8800h, 8880h, 8900h, 8980h
dw 8a00h, 8a80h, 8b00h, 8b80h, 8c00h, 9000h, 9080h
dw 9100h, 9180h, 9200h, 9280h, 9300h, 9380h, 9400h
dw 9800h, 9880h, 9900h, 9980h, 9a00h, 9a80h, 9b00h
dw 9b80h, 9c00h, 0a000h, 0a080h, 0a100h, 0a180h
dw 0a200h, 0a280h, 0a300h, 0a380h, 0a400h, 0a800h
dw 0a880h, 0a900h, 0a980h, 0aa00h, 0aa80h, 0ab00h
dw 0ab80h, 0ac00h, 0b000h, 0b080h, 0b100h, 0b180h
dw 0b200h, 0b280h, 0b300h, 0b380h, 0b400h, 0b800h
dw 0b880h, 0b900h, 0b980h, 0ba00h, 0ba80h, 0bb00h
dw 0bb80h, 0bc00h, 0c000h, 0c080h, 0c100h, 0c180h
dw 0c200h, 0c280h, 0c300h, 0c380h, 0c400h
;
counter db 40 ; timer counter
flag db 0 ; blink on/off flag
font dw ? ; font table location
;
oldvid db ?
buffer db 80 dup (?)
row db ?
col db ?
nchar db ?
fontbuf db 9 dup (?)
sbuf db ' '
;
install
cli
mov ax,cs
mov es,ax
mov bx,offset start ; ES:BX := handler
mov al,int_utma ; timer interrupt
mov ah,bif_setint
call bios_intfunc
sti
;
mov ah,dosf_outstr ; print sign-on
; message
lea dx,msg
int dosi_func
;
; install interrupt 81h
;
cli ; disable intrs
push es ; save ES
mov ax,0
mov es,ax ; bios segment
mov bx,81h * 4 ; pointer location
mov ax,offset blink ; get intr offset
mov es:[bx],ax ; install it
mov ax,cs ; get code seg

```

```

mov     es:[bx + 2],ax ; install it
pop     es             ; restore ES
sti

;
push    es             ; find font table
mov     ax,40h
mov     es,ax
mov     ax,0
mov     bx,offset bios_ctaddr
mov     si,es:[bx]
mov     bx,es:8[si]
mov     ax,es:[bx]
mov     font,ax
pop     es

;
;
mov     dx,offset stop ; terminate resident
int     dosi,termr

start   proc far

; interrupt handler
;
; preserve segments and registers
;
cli
push    ax             ; save al, it
; nas char
mov     cs:oidds,ds    ; save DS
mov     cs:oldes,es    ; save ES
mov     cs:oldes,es    ; save SS
mov     cs:oldsp,sp    ; save SP
mov     ax,cs
mov     ds,ax          ; set DS
mov     es,ax          ; set ES
mov     ss,ax          ; set SS
mov     ax,offset mystack
mov     sp,ax          ; set SP
push    bx
push    cx
push    dx
push    si
push    di
push    bp
pushf

; program area
;
cli
dec     counter
cmp     counter,0      ; char 0?
jc      doit
jmp     restore

doit:
mov     counter,40     ; reinit timer
; counter
cmp     onoff,1        ; are we active?
je      active         ; yes, do it
jmp     restore        ; no, forget it

active:
in      al,vcr         ; get video
; control reg
mov     oldvid,al      ; save it
mov     al,newvid
out     vcr,al         ; put in ours

;
cmp     flag,0         ; turn on or off
je      blinkon
jmp     blinkoff

; blinkon
;
mov     cx,0
lea     di,btable      ; point to data
table

```

```

nxtable:
mov     al,[di]        ; get 1st table
; byte
cmp     al,0ffh        ; := ffh?
jne     continue      ; yes, continue
jmp     notable        ; no, done

continue:
push    di
mov     al,1[di]
mov     row,al         ; get row
mov     al,2[di]
mov     col,al         ; get column
mov     ax,3[di]
push    es
mov     es,ax          ; segment of data
mov     ax,5[di]
mov     si,ax          ; offset of data
mov     cl,[di]        ; number of chars
mov     nchar,cl      ; save number chars
lea     di,buffer

inbuf
mov     al,es:[si]
mov     [di],al
inc     di
inc     si
loop   inbuf          ; word now in out
; buffer

;
pop     es
lea     bx,buffer      ; point to buffer
mov     cx,0
mov     cl,nchar        ; # char loop

next:
push    cx
lea     di,addrtbl     ; calculate pixel
; row
mov     ax,0
mov     al,row
mov     cx,9

mult0:
add     di,ax
add     di,ax
loop   mult0

;
mov     cx,0
mov     cl,[bx]
sub     cl,32
mov     si,font        ; find char in
; font table

cmp     cx,0
je      outloop

mult1:
add     si,9
loop   mult1

outloop:
;
mov     ax,0
mov     al,col         ; get char column
; print it
call    print
inc     bx
inc     col
pop     cx
loop   next           ; do next char

;
pop     di             ; restore table ptr
add     di,7           ; point to next
; element
jmp     nxtable        ; do this table

; notable
mov     flag,1         ; set blink off
; flag
jmp     done

; blinkoff
;
mov     cx,0
lea     di,btable      ; point to data
table

```

```

nxtable1:
    mov     al,[di]
    cmp     al,0ffh
    jne     continuel
    jmp     notable1

continuel:
    push    di
    mov     al,1[di]
    mov     row,al
    mov     al,2[di]
    mov     col,al
    mov     cl,[di]
    mov     nchar,cl

    lea     bx,sbuf
    mov     cx,0
    mov     cl,nchar

next1:
    push    cx
    lea     di,addrtbl

mult00:
    add     di,ax
    add     di,ax
    loop   mult00

    mov     si,font

    mov     ax,0
    mov     al,col
    call    print
    inc     col
    pop     cx
    loop   next1

    pop     di
    add     di,7
    jmp     nxtable1

notable1:
    mov     flag,0

done:
    mov     al,oldvid
    out     vcr,al

restore:
    popf
    pop     bp
    pop     di
    pop     si
    pop     dx
    pop     cx
    pop     bx
    mov     ds,cs:oldds
    mov     es,cs:oldes
    mov     ss,cs:oldss
    mov     sp,cs:oldsp
    pop     ax
    sti
    ret

start     endp

blink     proc     near
; interrupt 81h a data table for the timer interrupt to

```

```

; cause a sting of characters at a specified row, col
; to blink
;
; interrupt data
;
onoff     db     0
btable    db     7 * 100 dup (0ffh)

; dat table layout
; -----
;
; 1st byte ( 'onoff' ) this is a flag
; byte the entire routine will be by-
; passed if this byte = 1
;
; it is the users responsibility to cor-
; rectly fill the table with data the
; table starts in byte 2
;
; data table:
; -----
;
; 1 # of characters of string to blink
; (byte) if byte = ffh then inactive
;
; 2 character row of string to blink
; (byte)
;
; 3 character col of string to blink
; (byte)
;
; 4 segment in memory of strings location
; (word)
;
; 5 offset in memory of strings location
; (word)
;
start_int:
;
blink     endp

print     proc     near
; this procedure will write to video memory one character
; entry
;
; SI = pointer to char in font table
; DI := pointer to pixel row address
; AL := column
;
    push    cx
    push    si
    push    di
    push    ax
    push    bx
    push    es

nbyte
    mov     cx,9

    mov     bx,[di]
    add     bx,ax
    push    ax
    mov     ax,40h
    mov     es,ax
    mov     ah,es:[si]
    mov     dx,0e000h
    mov     es,dx
    mov     es:[bx],ah
    mov     dx,0c000h
    mov     es,dx
    mov     es:[bx],ah
    inc     di
    inc     di

```

```

        inc     si
        pop     ax
        loop    nbyte

        pop     es
        pop     bx
        pop     ax
        pop     di
        pop     si
        pop     cx
        ret

;
print      endp

stop:      ; terminate resident label

code      ends
end        main

```

The Main Logic

The resident routine is entered every 1/100 of a second, based on the timer interrupt. We have a counter which counts down every time it is entered (I used the value 40, which causes items to blink several times a second). If the counter is not zero, we exit the program. When the counter reaches zero, we reset the counter to 40, then we check the first byte of the blinker data table to determine if the blinker is active; remember, this is the on/off switch. If this switch is off, we exit; otherwise, we will execute the blink routines.

There are two blinking routines. They are 'blink_on' and 'blink_off'. We will go to these routines based on a flag which is set by these routines. For example, when we blink on, we set the flag so that the next time we go, we will blink off, and visa versa for blink off. The blink on routines will write to video memory information based on the data table. The blink off routine will write spaces to video memory based on the data table. One important point to realize is that the blinker program will read the data table sequentially until it encounters an FFh byte where the number of characters data item should be. So, if you modify this table by another program, you should restore it to FFh, or the next time it is activated it will print whatever is in the memory locations previously set. In most cases, if other programs had been run, you will get 'garbage characters' blinking on the display screen.

Filling The Table

Filling the table with data is a relatively simple matter if you are familiar with memory writing in Pascal, BASIC, or other programming languages. You can also do it directly by using 'DEBUG'. I have included all the procedures necessary to implement blinking data items using 'TURBO PASCAL'. I used a version 3.1 compiler, but I believe that all the functions I used are available in all the earlier compilers. See Listing 2 for these procedures.

There are several important points to remember about filling the blinker data table. Before writing data to the table, be sure the blinker is not active. If it is activated and you are in the process of writing data, you may get some unpredictable results. Another point to remember is that when you are finished with the table, clean up your mess, that is, restore it to its original condition. If you don't and leave segment values and offsets, it will blink whatever happens to be in that memory location. It is safe to leave the first byte at '0' and the next 700 bytes at FFh.

Listing 2

```

{
NAME      BLINK.LIB
DATE      OCT 17, 1985
AUTHOR    M D HOLMBERG, II

This is a set of procedures for use with
BLINK.COM, which is a resident program to let
selected data blink on the Z100 screen

The procedures.

1 GET_TABLE_POINTER( <arg1>, <arg2> ),
2 BLINK_ONOFF( <arg> );
3. RESTORE_BLINK_TABLE,
4 BLINK( <arg1>, <arg2>, <arg3>, <arg4> );
}

type      string80 = string[80];
var       bdata    = array[0..99] of string[80].

procedure get_table_pointer( var table_segment integer,
                             var table_offset  integer );
{
    this procedure returns segment:offset of the
    of the blinker data table
}

var       bios_seg  integer,
          bios_ofs  integer,

          seg_byte  array[1..2] of byte,
          ofs_byte  array[1..2] of byte,

          seg_ptr   integer absolute seg_byte; { redefines }
          ofs_ptr   integer absolute ofs_byte; { redefines }

begin
    bios_seg := $0;
    bios_ofs := $81 * 4;
    ofs_byte[1] := mem[ bios_seg + bios_ofs + 0 ];
    { get long ptr offset }
    ofs_byte[2] := mem[ bios_seg + bios_ofs + 1 ];
    seg_byte[1] := mem[ bios_seg + bios_ofs + 2 ];
    { get long ptr segment }
    seg_byte[2] := mem[ bios_seg + bios_ofs + 3 ];
    {
        seg_ptr + ofs_ptr := pointer to blinker data table
    }
    table_segment := seg_ptr;
    table_offset := ofs_ptr;
end;

procedure blink_onoff( x boolean );
{
    this procedure will activate or deactivate
    the blinker resident program
    1 to activate -- fill the data table first
    then call with blink_onoff(true);

    2. to deactivate -- call with blink_off(false).
}

var       tbl_offset integer;
          tbl_segment integer,

begin
    get_table_pointer( tbl_segment, tbl_offset );
    if x then mem[ tbl_segment + tbl_offset ] := 1 else
    mem[ tbl_segment + tbl_offset ] := 0;
end;

```

```

procedure restore_blink_table;
{
  this procedure restores the blink table to its
  original condition
}
var
  tbl_segment : integer;
  tbl_offset : integer;
  counter : integer;
begin
  get_table_pointer( tbl_segment, tbl_offset );
  mem[ tbl_segment + tbl_offset ] := 0;
  tbl_offset := tbl_offset + 1;
  for counter := tbl_offset to tbl_offset + 699 do
    mem[ tbl_segment + counter ] := $ff;
  end;
end;

procedure blink( row, col : byte;
                x : integer;
                s : string80 );
{
  this procedure will fill with the item you specify
  x = offset in table ~ 1 = 1st item
                    2 = 2nd item
                    3 = 3rd item, etc.
  s := string to blink on and off
  row, col are zero based.
}
var
  tbl_seg : integer;
  tbl_ofs : integer;
  bseg : array[1..2] of byte;
  bofs : array[1..2] of byte;
  bsegment : integer absolute bseg; { redefines }
  boffset : integer absolute bofs; { redefines }
begin
  blink_onoff( false );
  get_table_pointer( tbl_seg, tbl_ofs ),
  tbl_ofs := tbl_ofs + ( x * 7 ) + 1,
  bdata[ x ] = s;
  bsegment := seg( bdata[ x ] );
  boffset := ofs( bdata[ x ] ) + 1,
  mem[ tbl_seg + tbl_ofs ] := length( bdata[ x ] );
  mem[ tbl_seg + tbl_ofs + 1 ] := row;
  mem[ tbl_seg + tbl_ofs + 2 ] := col;
  mem[ tbl_seg + tbl_ofs + 3 ] := bseg[ 1 ];
  mem[ tbl_seg + tbl_ofs + 4 ] := bseg[ 2 ];
  mem[ tbl_seg + tbl_ofs + 5 ] := bofs[ 1 ];
  mem[ tbl_seg + tbl_ofs + 6 ] := bofs[ 2 ];
  blink_onoff( true );
end;

```

The Turbo Pascal Routines

In Listing 2, I have provided all the routines necessary to implement blinking data on your display terminal. If these routines are typed into a file and are included into your programs, you can very easily dress up your programs with blinking words. (i.e. {\$i blink.lib}). This listing includes four procedures.

1. GET_TABLE_POINTER
2. BLINK_ONOFF
3. RESTORE_BLINK_TABLE
4. BLINK

The first procedure is used by all the other procedures to find the address in memory of the blinker data table. The table is pointed to by the interrupt 81H interrupt vector. When this procedure is executed, it reads the offset and segment address of the table and passes them to two variable arguments. The second procedure turns the blinker on or off. First, it gets the location of the table, then it writes to the on/off flag byte depending on its argument. This routine requires a boolean argument, that is true or false. The syntax is:

```

BLINK_ONOFF( true ),      this turns the blinker on
or
BLINK_ONOFF( false ),    this turns the blinker off

```

The third procedure restores the blinker data table to its original condition, which is a zero in the first byte, and FFh in the following 700 bytes. As I have mentioned earlier, it is recommended that you use this procedure in a final clean-up routine before your program terminates.

The fourth procedure is the heart of these routines. This procedure with its four arguments writes information to the blinker data table. The syntax is:

```

BLINK( <arg1>, <arg2>, <arg3>, <arg4> ),

```

Argument 1 and 2 are byte sized. They are row and column of the display screen and are zero based. That is, rows 0 to 24 and column 0 to 79. Argument 3 is an offset to that data table; for example, a zero is the first location in the table, 1 is the second, 2 is the third, etc. This number corresponds to the order in which you use this routine. Suppose you wanted to blink the following data:

```

'HELLO'
'Z-100 COMPUTER'
'HEATH USERS' GROUP'

```

You would use 0, 1, 2 for the offset argument. If you are going to place a large number of blinking items on the screen, it would be a good idea to use an integer variable that increments each time it is used. The final argument is a string argument 80 characters in length. You can either use a string in single quotes or pass a string variable of type 'string80', e.g.

```

VAR1 = string80;

```

When this procedure executes, it first finds the address of the data table, then it turns off the blinker. It then writes information to the blinker data table based on your arguments, and then turns the blinker back on.

Some examples:

```

BLINK(0,0,0,'I'm a member of HUG'),

```

This will blink the string in the top left hand corner of your screen.

```

BLINK(12,40,1,'I LOVE MY H/Z-100').

```

This will blink the string in about the middle of the display screen.

Here is a brief program example using these procedures.

```

program blink_test( input, output);
{$i blink.lib}      | the file created from listing #2 |

```



```

var      prompt  : char;

begin
  blink(0,35,0,'I LOVE HUG'), { this will blink 'I LOVE
                             HUG' on the middle of
                             the top line of the
                             screen }
  blink(23,35,1,'MY H/Z-100'), { this will blink 'MY
                                H/Z-100' on the 24th
                                line of the screen }
  read(kbd.prompt);           { wait for a key to be
                              pressed }
  restore_blink_table;       { restore blink table
                              after a key has been
                              pressed }
end

```

This is a very simple example of implementing these routines, but you can see how useful they can be. I have found them very useful for programs with menus and input screens to highlight certain areas of the display.

Using Debug

It is also possible to see the blinker in action using 'DEBUG.COM'. In fact, I used this to test the program before I had written the PASCAL routines. Shown below are the entries required to make the phrase 'I LOVE HUG' blink in the top left corner of your screen.

```

<CR> := return key   <SP> := space key   * := note

debug <CR>
e100 <CR>
49 <SP> 20 <SP> 49 <SP> 20 <SP>
4C <SP> 4F <SP> 56 <SP> 45 <SP>
20 <SP> 4B <SP> 55 <SP> 47 <CR>
d100 <CR>

```

Now you will see the hex values and the ASCII equivalents at offset 100h. You need to write down somewhere the segment and offset that this data is located. The offset is 100h, but the segment value will be different for your computer. In our example, we will use 913eh.

The next step is to find the location of our data table. To do this, we must look at the bios section of memory and get the pointer to interrupt vector 81h, it is located at 0:204.

```
a0:204 <CR>
```

The first 4 bytes you see is the pointer to our data table. They were put there by our blinker program. If the first 4 bytes you see is '64 00 40 00', then you haven't run the blink program yet. If that is so, you need to run BLINK.COM and start over. These 4 bytes will locate the data table. On my computer, they are: 'D2 05 8C 8D'. The first 2 bytes is the offset and the next 2 bytes is the segment. If you are not familiar with memory organization, you should know that words in memory are reversed. For example, the above 4 bytes are really segment 8D8C, and offset 05D2.

```
a8d8c:05d2 <CR>
```

If you used the pointer that was on your computer and not mine, you should see our data table. The first byte is '0' and the following bytes are 'FF'. Now, all we have to do is fill in those bytes and we will have a blinking 'I LOVE HUG'.

```
e8d8c:05d2 <CR>
<SP>
```

Pressing space here skips the first byte. Remember, the first byte is the on/off switch and we want to leave it alone until we are finished.

```
A <SP> 0 <SP> 0 <SP>
```

Now we have entered the number of characters in 'I LOVE HUG', and the row and column. We are now ready to put in the segment and offset for the data we entered earlier. In my case, it was 913E:0100. Here also, we need to reverse these bytes so I would enter 3E 91 for the segment and 00 01 for the offset.

```
3E <SP> 91 <SP> 00 <SP> 01 <CR>
```

At this point, we have the table for 1 data element loaded. You could continue to do this until we have entered 100 items, but we will do only one.

```
e8d8c:05d2 <CR>
```

Be sure you used your address for the segment:offset for the data table.

```
1 <CR>
```

If you did everything right, 'I LOVE HUG' should be blinking in the upper left hand corner of the screen.

```
e8d8c:05d2 <CR>
0 <CR>
```

This will turn the blinker off.

```
f8d8c:05d3 05d9 ff <CR>
```

This will restore the data table to its original condition.

For more information on using 'DEBUG', refer to the MS-DOS operating system documentation.

The Assembler

Listing 1 is the source code for the blinker resident program, and must be entered into the computer using your favorite editor. Be especially careful with the data section at the 'ADDRTBL'. This section contains the addresses of video memory, and if you have a typo here, you won't write to the screen correctly. Looking through the listing, you will find numerous comments. This is documentary information only and is not necessary for assembly. After you have entered in the program and saved it in a file called 'BLINK.ASM', you will need the following files on disk to complete the assembly process:

```
BLINK.ASM
MASM.EXE
LINK.COM
EXE2BIN.COM
```

Once you have all these files on the default disk, type in these lines:

```
MASM blink;
```

This assembles your source file and creates an object file.

```
LINK blink;
```

This links your object file and creates an 'EXE' file. You will get one error message for the stack segment, this is normal so don't worry about it.

```
EXE2BIN blink.exe blink.com
```

This converts you 'EXE' file to a 'COM' file.

After you have finished these steps, you can delete 'BLINK.EXE' and 'BLINK.OBJ'. The file named 'BLINK.COM' is your final executable file. Now type BLINK. After you have done this, the blinker is installed. You should see a message such as 'Z100 Blinker installed...'. Do this step only once. This sets up your computer to blink when called upon, and you may now use that Turbo Pascal procedure, DEBUG, or other languages to implement blinking displays on your H/Z-100.

Possible Enhancements

For all those talented programmers, I can think of several enhancements that could be made to the blinker. Such as:

1. Including the graphics characters from the font table.
2. Implementing blinking in color.
3. Including a customized font table to give that extra touch.

I enjoyed submitting this article to HUG and would be very interested in any comments by mail or through Buggin' HUG that you may have. *



GRAPH

Not just another plotting program.

GRAPH will analyze and plot both data and functions making it the ultimate TOOL for analysis and signal processing for Engineers, Scientists, Students, and other professionals.

Here are just some of the many features GRAPH offers: High resolution printer and HP plotter output, data file input, multiple function plotting, automatic scaling, smoothing, linear and log axes, parametric equations, extensive math functions including numerical integration, weighting functions (e.g. Taylor, Hamming), complex modeling, FFT analysis, convolution, interactive TEXT mode including Greek characters....and so much more.

GRAPH is database driven and will run your favorite word-processor directly (min 448K RAM required for this feature).

Requires H/Z100, H/Z150, IBM PC/XT/AT or true compatibles, ZDOS or MSDOS, minimum 256K RAM, standard color/graphics card. 8087 numeric coprocessor recommended.

Only \$129.95 check or money order, plus \$4.00 shipping and handling. Michigan residents add 4% sales tax.

To order (specify computer type, memory size, 8087 option) or for more information write to...

Wolfpac Technology
P. O. Box 378
Plymouth, Michigan 48170

FOR THE Z-100

Conforms to
IEEE -696 (S-100)

**A low-cost high-performance
memory board from
Software Wizardry, Inc.**

- Up to 1 megabyte on a single card with parity error checking
- On-board clock/calendar with software
- The only memory card not manufactured by ZDS that is specifically designed for the Zenith Z-100
- Emulates the Z-205 registers for complete present and future Z-100 compatibility
- Runs at 8 MHz with no wait states [when loaded with 150 nanosecond chips]
- Ramdrive software included
- Supports 24 bit addressing

List price: \$395 [unpopulated]
\$520 [with 1 megabyte]



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

ZPC Update #7



Pat Swayne
HUG Software Engineer

*ZPC - 1-13-87
Patcher 12-02-86
SET ZPC 5-19-86*

This is the seventh in a series of articles in support of ZPC, a program that allows you to run IBM software on Z-100 (dual processor) computers. By now, I hope all of you that had the original ZPC have upgraded to the new version 2.

In this edition of ZPC Update, I will discuss QuickBASIC, MultiMate version 3.31, the WATFOR FORTRAN compiler, and (I suppose it had to happen) bug fixes for ZPC Version 2. The bugs are in PATCHER and SETZPC. In conjunction with WATFOR, I will discuss two things of possible interest to all ZPC users: How to run programs that use interrupt FF (hex), and how to handle an unusual I/O method.

Some of you have asked me how you can test the ZPC Hardware Support Circuit that was described in the April 1986 REMark. Following software discussions in this article, I will present some simple tests for ZHS.

PATCHER Bug Fix

The PATCHER program (used to patch files) contains a bug that never showed up until I tried to work out the patch for MultiMate version 3.31. The bug only shows up if the patch description (addresses and data) in the PATCHER.DAT file takes up a certain number of bytes. To fix PATCHER, create a file called FIXPCH.DAT that contains these lines:

```
E1F9
90 90 90
W
Q
```

Copy this file and DEBUG.COM to a disk containing PATCHER.COM, log on to the disk, and enter

```
DEBUG PATCHER.COM <FIXPCH.DAT
```

at the system prompt, and hit RETURN. PATCHER will be fixed for you. If you would rather use PATCHER to fix itself, use an editor to add these lines at the end of your PATCHER.DAT file:

```
PATCHER Patch
Insert the disk containing PATCHER.COM
PATCHER.COM
F9.90.90.90
z
```

Run PATCHER as instructed in your ZPC Version 2 manual and select "PATCHER Patch" from the menu to apply the patch to PATCHER. Note: If PATCHER.COM is dated later than 3-25-86, the patch has already been made. *120386*

SETZPC Bug Fix

A user has reported a bug in the SETZPC program that is included with ZPC Version 2. If you change the monochrome normal intensity color to yellow, and try to permanently record the change by answering Y to the "Write changes to disk" prompt, the change will not be permanently recorded, and the next time you boot and load ZPC, the color will be green again. To fix SETZPC using DEBUG, make a file called FIXSET.DAT that contains these lines:

```
E7FC
12
W
Q
```

Now, copy DEBUG.COM to your ZPC system disk or partition (if it is not already there), and enter

```
DEBUG SETZPC.COM <FIXSET.DAT
```

at the system prompt, and hit RETURN. SETZPC will be fixed, so that monochrome color changes are permanently recorded when you want them to be. If you would rather use PATCHER to fix SETZPC, add these lines at the end of your PATCHER.DAT file, using an editor:

```
SETZPC Patch
Insert the disk containing SETZPC.COM
SETZPC.COM
6FC,12
z
```

Run PATCHER as instructed in your ZPC Version 2 manual and select "SETZPC Patch" from the menu to apply the patch to SETZPC. **Note:** If SETZPC.COM is dated later than 3-05-86, the patch has already been made.

Compiled QuickBASIC Programs

Provided with ZPC Version 2 is a program that patches QuickBASIC programs that have been compiled with the /O switch so that they will run under ZPC. As it turns out, the program can also be used to patch the Runtime utility provided with QuickBASIC so that programs that have been compiled WITHOUT the /O switch can be run. Just enter the name of the Runtime utility (BRUN10.EXE or whatever your version is) as the argument to the FIXQB program to fix it. ZPC version 1 owners can use the QuickBASIC patches in the March 1986 REMark to fix the Runtime utility. Whatever method you use, just make sure that you do not patch anything on the original QuickBASIC disk. Normally, you would place a copy of the Runtime utility on the disk containing your compiled program(s), and it is this copy that you would patch.

Multimate version 3.31

If you have used the patches that were released for MultiMate version 3.3 as a guide in trying to patch MultiMate 3.31, you probably did not have any luck. As it turns out, they rewrote the video section of MultiMate for version 3.31, and the patches are quite different. If you have the standard IBM PC version of MultiMate 3.31, you can patch it by adding these lines to your PATCHER.DAT file (be sure you have fixed PATCHER first):

```
MULTIMATE version 3.31 (IBM)
Insert the System disk containing WP.EXE
WP EXE
D53B,90,90
D597,EB
D63D,90,90
D721,90,90
D7B9,EB
D849,90
D860,90
z
```

Note: When you add lines to your PATCHER.DAT file, you can add them to the beginning or end of the file, or anywhere in the middle, as long as the new lines immediately follow any lower case letter z that is on a line by itself. Use an editor such as EDLIN or a word processor that can produce normal ASCII text (such as WordStar in the Non-document mode) to add the lines. Follow the instructions for PATCHER in your ZPC manual to patch MultiMate 3.31. If you do not have PATCHER, you can patch MultiMate 3.31 by creating a file called MM31PCH.BAT that contains these lines:

```
REN WP.EXE WP.BIN
DEBUG WP.BIN <MM31PCH.DAT
REN WP.BIN WP.EXE
```

Now create a file called MM31PCH.DAT that contains these lines:

```
ED63B
90 90
ED697
EB
ED73D
90 90
ED821
90 90
ED8B9
EB
ED949
90
ED960
90
W
Q
```

Copy MM31PCH.BAT, MM31PCH.DAT, and DEBUG.COM to your MultiMate system disk, log on to the disk, and enter

```
MM31PCH
```

at the system prompt, and hit RETURN. **Note:** This is probably the last time I will include patches for PC programs in both PATCHER and DEBUG formats. Patches will be presented in PATCHER format only. If you study the two formats above, you should be able to see the relationship between them which will allow you to develop DEBUG patches from the PATCHER format. By upgrading to ZPC Version 2, you would then have PATCHER.

To upgrade to ZPC Version 2, send your original ZPC disk (885-3030-37) and \$20.00 (check or money order only! Made payable to Heath Users' Group) to Heath/Zenith Users' Group, Attn: Nancy Strunk, Hilltop Road, St. Joseph, MI 49085. If you have both the old ZPC and the ZPC Support Disk (885-3034-37), send both disks and \$15 in to HUG to get the new ZPC. The new ZPC replaces both disks. If you only have the old ZPC Version 1, send that disk in plus \$20.

WATFOR FORTRAN

WATFOR FORTRAN (WATFOR77.EXE) will not run under ZPC for two interesting reasons. The first is that it uses interrupt FF (hex), which is not available on a Z-100 whether you are using ZPC or not. The other is that it opens the CON device as a file for console I/O during a running program, rather than using the predefined standard I/O handles.

The use of interrupt FF is a problem because the MTR-100 monitor ROM in the Z-100 uses the vector for that interrupt to store a pointer to the ROM's data segment. If a program overwrites that vector with something else, the screen goes crazy, and the computer locks up. Fortunately, though, the first thing in the ROM data segment is a jump to the reset entry point of the ROM. If you capture an attempt to set up the interrupt FF vector and place the address that the program wants to put there after that jump instruction, the program can then use the interrupt. I have written a little program called FIXIFF.COM that does just that. If you type in the following BASIC program and run it, it will generate FIXIFF.COM for you.

```
10 REM THIS PROGRAM CREATES FIXIFF.COM
20 DEFINT A-I:OPEN "0",1,"FIXIFF.COM"
30 S=0:S1 = 7663 :FOR I=1 TO 86
40 READ B:S=S+B:PRINT #1,CHR$(B);
50 NEXT I:IF S<>S1 THEN PRINT "TYPING ERROR!" :END
60 CLOSE #1:LOCATE 23,1:PRINT "DONE!" :SYSTEM
70 DATA 235,53,0,0,0,0,61,255,37,116
80 DATA 10,61,255,53,116,5,46,255,46,2
90 DATA 1,87,6,51,255,142,199,191,252,3
100 DATA 38,196,61,71,128,252,53,116,10,38
110 DATA 137,21,38,140,93,2,7,95,207,38
120 DATA 196,29,95,95,207,30,51,192,142,216
```

```

130 DATA 190,132,0,196,60,199,4,6,1,140
140 DATA 76,2,31,137,62,2,1,140,6,4
150 DATA 1,186,55,1,205,39

```

If you would rather use an assembler to make FIXIFF.COM, the source code is listed at the end of this article. To use FIXIFF.COM, just copy it to your ZPC system disk, and enter

```
FIXIFF
```

at the system prompt, and hit RETURN. FIXIFF remains resident in memory until you reboot your computer, and will capture any attempt to set up interrupt FF.

Once you have installed FIXIFF, WATFOR will actually run on a Z-100 without ZPC, but none of its support programs, such as CONFIG77 or the WATCOM editor will run. With ZPC in the PC mode, the support programs will run, but WATFOR will not run properly because of the second problem stated above: It will open the system console device, CON, as a file for console I/O.

Usually when a program does console (screen and keyboard) I/O via the newer XENIX-type functions built into MS-DOS, it will use the built-in "handles", or channels provided for this purpose. When ZPC is in the PC mode, it intercepts those standard channels so that the I/O will be done PC-style. However, a program can open its own channels for console I/O, and if it does, ZPC will not "see" what is going on, so characters going to the screen will go Z-100 style. When I ran a sample program under WATFOR after installing FIXIFF with ZPC in the PC mode, I got a combination of PC and Z-100 characters on the screen, and some characters overwrote others. It seems that WATFOR's own output goes through standard channels, while the FORTRAN program's output goes through new channels.

The solution is to capture any attempt to open new channels for the console device, and always use the standard channels. As usual, I have written a little program, FIXCON.COM, that can do the job. If you type in the following BASIC program and run it, it will generate FIXCON.COM.

```

10 REM THIS PROGRAM CREATES FIXCON.COM
20 DEFINT A-I:OPEN "0",1,"FIXCON.COM"
30 S=0:S1 = 9999 :FOR I=1 TO 103
40 READ B:S=S+B:PRINT #1,CHR$(B);
50 NEXT I:IF S<>S1 THEN PRINT "TYPING ERROR!" :END
60 CLOSE #1:LOCATE 23,1:PRINT "DONE!":SYSTEM
70 DATA 235,70,0,0,0,0,128,252,60,116
80 DATA 15,128,252,61,116,10,128,252,62,116
90 DATA 44,46,255,46,2,1,86,139,242,129
100 DATA 60,67,79,117,4,131,124,2,78,94
110 DATA 117,235,83,187,1,0,128,252,60,116
120 DATA 7,168,1,117,3,187,0,0,139,195
130 DATA 91,248,202,2,0,131,251,1,119,207
140 DATA 235,245,30,51,192,142,216,190,132,0
150 DATA 196,60,199,4,6,1,140,76,2,31
160 DATA 137,62,2,1,140,6,4,1,186,72
170 DATA 1,205,39

```

The assembly source code for FIXCON is included at the end of this article. To use FIXCON.COM, copy it to your ZPC system disk, and enter

```
FIXCON
```

at the system prompt, and hit RETURN. FIXCON will remain resident in memory until you reboot. FIXCON must be loaded after ZPC in order to work properly. With both FIXIFF and FIXCON loaded, you can run WATFOR, the WATCOM editor, and the configuration programs for both of them in the PC mode of ZPC. One unusual thing about WATFOR when you run it in the PC mode is that you cannot exit from it by typing Control-Z and

RETURN as instructed. However, Control-C will do the job. On the other hand, when you run WATFOR in the Z-100 mode, Control-Z works, and Control-C does not.

Testing The ZPC Hardware Support Circuit

To test the ZHS circuit, first load ZPC Version 2, or the original ZPC plus the KEYINT program from the April article. Turn on the PC mode of ZPC. Then run DEBUG and do several inputs from port 60. You should consistently get a value of 1C (hex). Now, do several inputs from port 3DA. On a Z-100, you will alternately get FF or F6 as you read the port. In other words, if you get FF the first time, you will get F6 the second time, FF the third time, F6 the fourth time, etc. On an ET-100, the values read may not be consistent, but bits 0 and 3 should alternately be 0 and 1. In other words, if bits 0 and 3 are both 0 the first time, they will both be 1 the next time, etc. The other bits are "floating" which is why the numbers may not be consistent.

If you get the proper values from ports 60 and 3DA, your ZHS board is working properly.

FIXIFF And FIXCON Source Listings

```

PAGE ,132
FIXIFF -- THIS PROGRAM ALLOWS PROGRAMS THAT USE
INTERRUPT 0FFH TO RUN ON A Z-100

;
; BY P SWAYNE, HUG SOFTWARE ENGINEER 14-MAY-86
;
DUMMY SEGMENT STACK ,AVOID LINK ERR MSG
DUMMY ENDS
CODE SEGMENT
ASSUME CS:CODE,DS:CODE,ES:CODE,SS:CODE
ORC 100H

START JMP SHORT SETUP

SYSADR DW 0,0 ;SYSTEM CALL ADDRESS

;
; PROCESS SYSTEM CALLS HERE. LOOK FOR INT 0FFH SETUP
;
INT21 CMP AX,25FFH ;SET UP INT 0FFH?
JZ FIXINT ;IF SO, FIX IT
CMP AX,35FFH ;GET 0FFH VECTOR?
JZ FIXINT ;IF SO, FIX IT
JMP CS:DWORD PTR SYSADR ;ELSE, GO TO SYSTEM

FIXINT PUSH DI
PUSH ES
XOR DI,DI
MOV ES,DI ;POINT TO PAGE 0
MOV DI,3FCH ;POINT TO INT FF VECTOR
LES DI,ES:DWORD PTR [DI] ;GET ADDRESS
INC DI ;SKIP JUMP
CMP AH,35H ;GET VECTOR?
JZ FIX35 ;YES
MOV ES:[DI],DX ;INSERT NEW VECTOR
MOV ES:2[DI],DS
POP ES
POP DI
IRET ;RETURN, VECTOR SET

FIX35 LES BX,ES:DWORD PTR [DI] ;GET VECTOR
POP DI ;REMOVE PUSHED ES
POP DI ;FIX DI
IRET

ENDRES: ;END OF RESIDENT CODE

;
; SET UP TO INTERCEPT SYSTEM CALLS
;
SETUP PUSH DS
XOR AX,AX
MOV DS,AX ;POINT TO INT SEGMENT
MOV SI,21H*4 ;POINT TO INT 21 VECTOR
LES DI,DWORD PTR [SI] ;GET IT
MOV WORD PTR [SI],OFFSET INT21
MOV WORD PTR [SI+2],CS ;INSERT NEW VECTOR

```

```

POP      DS          ;FIX DS
MOV      SYSADR,DI
MOV      SYSADR+2,ES ;SET UP CHAIN TO OLD CODE
MOV      DX,OFFSET ENDRES ;END OF RESIDENT CODE
INT      27H        ;EXIT, CODE RESIDENT

CODE     ENDS
END      START

PAGE     ,132
;        FIXCON --- THIS PROGRAM ALLOWS PROGRAMS THAT USE
;        "CON" AS AN I/O DEVICE SEPARATE FROM THE STANDARD
;        I/O CHANNELS TO WORK UNDER ZPC
;
;        BY P. SWAYNE, HUG SOFTWARE ENGINEER 15-MAY-86

RETFD   MACRO
DB      @CAH          ;DEFINE RET/DISP
ENDM

DUMMY   SEGMENT STACK ;AVOID LINK ERR MSG
DUMMY   ENDS
CODE    SEGMENT
ASSUME  CS:CODE,DS:CODE,ES:CODE,SS:CODE
ORG     100H

START:  JMP      SHORT SETUP

SYSADR  DW      0,0    ;SYSTEM CALL ADDRESS
;
;        PROCESS SYSTEM CALLS HERE, LOOK I/O CALLS

INT21:  CMP     AH,3CH ;CREATE?
        JZ     FIXOP  ;YES, FIX IT
        CMP   AH,3DH ;OPEN?
        JZ     FIXOP  ;YES, FIX IT
        CMP   AH,3EH ;CLOSE?
        JZ     FIXCL  ;YES, FIX IT

JMPSYS: JMP     CS:DWORD PTR SYSADR ;ELSE, GO TO DOS
FIXOP:  PUSH    SI
        MOV    SI,DX ;POINT TO PATH NAME
        CMP   WORD PTR [SI],'OC' ;CHECK FOR "CON"
        NOCON: NOCON ;NOT "CON"
        CMP   WORD PTR 2[SI],'N'
        POP    SI
        JNZ   JMPSYS ;NOT "CON", GO TO DOS
        PUSH  BX
        MOV   BX,1 ;ASSUME WRITE
        CMP  AH,3CH ;CREATE IS WRITE
        JZ   GOTHAN
        TEST AL,1 ;OPEN FOR WRITE?
        JNZ GOTHAN
        MOV  BX,0 ;NOT WRITE
GOTHAN: MOV   AX,BX ;GET HANDLE
        POP  BX
FIXEX:  CLC     ;NO ERROR
RETFD   ;RETURN, STATUS INTACT
        DW    2
FIXCL:  CMP    BX,1 ;CLOSING "CON"?
        JA   JMPSYS ;NO, GO TO DOS
        JMP  FIXEX ;ELSE, SKIP CLOSE
ENDRES: ;END OF RESIDENT CODE
;
;        SET UP TO INTERCEPT SYSTEM CALLS
SETUP:  PUSH    DS
        XOR   AX,AX
        MOV  DS,AX ;POINT TO INT SEGMENT
        MOV  SI,21H*4 ;POINT TO INT 21 VECTOR
        LES  DI,DWORD PTR [SI] ;GET IT
        MOV  WORD PTR [SI],OFFSET INT21
        MOV  WORD PTR [SI+2],CS ;INSERT NEW VECTOR
        POP  DS ;FIX DS
        MOV  SYSADR,DI
        MOV  SYSADR+2,ES ;SET UP CHAIN TO OLD CODE
        MOV  DX,OFFSET ENDRES ;END OF RESIDENT CODE
        INT  27H ;EXIT, CODE RESIDENT

CODE    ENDS
END     START

```

□ BASIC BUSINESS™

Accounting and Business Management Software

Basic Business II \$249.95
MS-DOS or PC-DOS

Basic Business I \$149.95
MS-DOS, PC-DOS or CP/M

Advanced Color On-Screen-Help Windows

with computerized indexing into the users manual

Never again wonder what to do next, or how to do it. Basic Business is the only advanced accounting system that provides a complete windowed help and index system. Answers to your questions are as close as the touch of a key.

Not another cigar box system, Basic Business II offers features equivalent to those found in systems costing hundreds more. Accounting has never been easier or more complete.

Need less help? Consider the popular Basic Business I. It has most of the features found in Basic Business II without the windows and help, and is upward compatible should your needs change.

Consider the flexibility of Basic Business. You can generate over 300 reports or use the optional dBase File Formats for ultimate information gathering and reporting. Transfer information to a database or spreadsheet program for total custom reporting ability. Source code is also available for those companies requiring customization or having specialized accounting needs.

Minimum Hardware Requirements: 80 X 24 CRT, 2-360K floppies, 132 column printer, 64K CP/M 2.x, 128K MS-DOS or PC-DOS 2.0, PC/XT/AT or compatible.

30 day money back guarantee if all claims and features listed in this ad are not true. MasterCard and VISA welcome. Add \$2.50 for shipping. CA residents add 6% sales tax.

Optional Products

Point-of-Purchase Module... \$99.95

dBASE II/III File Formats... \$29.95

some disk formats slightly higher

PARTIAL LIST OF FEATURES

General Information

- All modules are menu driven / password protected
- Unlimited data entry in all modules
- Prints hundreds of detailed management reports
- Superior documentation for ease of use
- Stock or preprinted forms available
- Supports user defined notes / comments on all forms
- All modules completely integrated

General Ledger

- Unlimited user defined Chart of Accounts
- Prints all standard financial reports
- Up to 99 user-defined departments

Accounts Receivable & Billing

- Open item or balance forward
- Verifies customer credit limit at invoice entry
- Supports service or inventory billing
- Prepares customer statements with dunning notes
- Maintains full cash flow analysis
- Prepares mailing labels and customer lists

Accounts Payable

- Cash disbursement management for all vendors
- Supports hand written checks
- Automatically posts recurring invoices monthly
- User defined aging periods
- Prints vendor mailing labels and lists

Payroll & Job Costing

- Supports labor related job costing
- Maintains federal, state and local tax tables
- Handles vacation and sick leave time
- Prints W-2 forms and 941 information

Inventory

- Up to 12 digit alphanumeric part numbering
- Supports standard / average / suggested resale costing methods
- Integrated with Sales and Purchase Order systems

Purchase Order

- Maintains PO and Inventory files
- Prints confirming POs
- Prints price variance report by part number
- Maintains vendor purchase history

Order Entry & Sales Analysis

- Supports ordering and shipment of stock and non-stock items
- Handles partial shipments / cancellations / changes of any SO
- Automatic credit checking during SO entry / invoicing / billing
- Provides extensive sales analysis reporting
- Maintains MTD billing and booking totals
- Line item/total invoice discounts during SO entry and / or invoicing

Point of Purchase (optional)

- Entry of invoices posts to all pertinent modules
- Invoices: stock / non-stock items
- Line item discount and tax application
- Verifies customer credit limits
- Supports cash / check/credit card / payment on account and returns
- Controls electronic cash drawer

CALL TODAY **National (800) 821-8778**
California (800) 521-7182

Hours: Mon - Fri 8 am - 5 pm PDT

DEALER INQUIRIES WELCOME.



advent
products inc.

3154-F E. La Palma Ave.
Anaheim, CA 92806
(714) 630-0446

Trademarks: CP/M - DRI; dBASE - Ashton-Tate;
MS-DOS - Microsoft; PC-DOS - IBM

Keyboard Codes, Menus, And TURBO Pascal

J. R. "Jeff" Jeffrey

2300 Lear Lane
Austin, TX 78745
Copyright

Introduction

Going to sleep watching ordinary, nay, plain menus on your Z-100. Would you love to make your menus sparkle and look like professional work with TURBO Pascal, but don't know how. Well then, listen up, this article is for you.

I was in the same boat or menu screen, as it were, going to sleep punching one, two, or three for my choice when I said to myself, "Self, there has gotta be a better way." Having just figured out via the kind services of this magazine's June 1984 issue and Richard Kolmar, "Here's More On The Z-100 Key Click", page 97, how to interpret escape codes for use in AUTOEXEC.BAT files, my interest was definitely piqued.

My need for an above ordinary menu was for use in programs written on my home machine concerning Operations Research Analysis. I ran into a situation where it was necessary for me to pick one of three printers on which to plot a cumulative probability distribution. Of course, they all used different codes to compress the print and change line spacing from six to eight lines per inch, so I needed a way to choose between them or to abort the routine if I had none of these printers attached or didn't want to print the graph. Thus, was born, a living color, direction arrow operated, choose with return, menu.

All That Sparkles Is Color

Now it's time for all those folks in the back row who are bored with my ramblings about whys and wherefores to wake up, because I'm going to tell you a few things not found in the books (or at least not that I could find in any form usable by a human being). Let's start with how to generate color.

If you've used TURBO for any length of time, you realize that Borlund didn't put a color routine in TURBO for us non-clone users. The routine to generate color is the first procedure in our demonstration program, TESTMENU.PAS. All you really need to do is send a four part escape code sequence to the screen via the write statement. See page 10.50, in the Z-100 Technica! Manual,

hereafter referred to as TM-100. Since you will want to use numbers as your actual parameters, this routine must also convert integers to string, i.e., the standard procedure `str(number,character)`, page 70, TURBO, Version 3, Reference Manual, hereafter referred to as TRM. All you need to do is type `Color(5,1)`, to get cyan foreground with blue background, my favorite combination, while a similar statement using another combination, say `Color(7,0)` for white on black, changes it.

Passing Menu Lines

The very first type declaration, the one for `MenuLine`, allows the passing of the actual menu line statements between two procedures which reference each other. This is poor programming practice, but I couldn't figure out another way without using forward references or embedding the actual menu lines in procedure `Menu`, neither of which I wanted to do. The `var` declaration for `MenuLineNr` allows passing that variable between procedures necessary to first set and then reset the printer. If you use the menu procedure more than once in a program, you will need to specify more than one of these variables for various menu actions. Now on to the real meat of this exercise. The `Menu` procedure itself.

Procedure Menu — Formal Declaration Statement

The formal declaration statement for this procedure contains the capability to specify the horizontal and vertical positions of the menu, `MenuHorzPos` and `MenuVertPos`, from the screen left and top, respectively. You must also pass to the procedure the number of lines of menu where the cursor will move, `NrLines`, and the array, `M`, containing the lines to be printed. The array will be set up in a procedure which uses procedure `Menu`, hence, the need to define this array as an external/global variable to both procedures. Finally the formal declaration contains the variable, `MenuLineNr`, which will be passed back to the variable of the same name used to set the printer, then reset it.

Procedure Menu — Escape Codes

The real heart of procedure `Menu` lies in the constant declaration section. It took quite a while to figure out how to set this sec-

tion up because neither Zenith's TM-100 nor the TRM tell you that the computer powers up with key expansion enabled, thus, requiring you to first check for an escape character for direction arrow depression, and then requiring you to check for 'A' or 'B', for up or down, respectively. There are also a few strange looking characters showing in the constant declaration section: # and ^[.

The # is used to indicate to TURBO that the next set of numbers, be they decimal or hexadecimal, represent a character, hence, #27 or #\$1B represent Escape. There is also another way to represent Escape, ^[. All of these codes are mentioned on page 45, of TRM. The reason for using multiple versions of Escape was to see if they all worked (they do) and because you can't declare a constant, like Esc = #27;, and then use it in a statement, like ReverseVideoOff = Esc + 'p';, in the constant declaration section. That amounts to using variables in the constant declaration section and isn't allowed. Therefore, I used RVideoOff = ^['p'; per page 45 of TRM, instead.

Procedure Menu — Logic

From this point on, the procedure is documented sufficiently to understand, with some minor exceptions, so I will cut down on my remarks and encourage you to follow the logic.

The first "for loop" positions the start of the menu on the screen and prints the menu lines. The section following the "for" statement returns the cursor to the first menu line, highlights it, and then, by setting the boolean MenuLineSelect variable to off, makes sure stray codes don't indicate a line has been selected.

The real body of the procedure logic is contained within the "(motion selection*)" routine's "repeat" statement. This routine accepts a single character from the keyboard, tests it to see if it is the code for Escape (Esc), if so, a second character is read from the keyboard buffer as if a second key depression had occurred. This second character is tested for the code for Up or Down. If the code for Up or Down is found, then the cursor is positioned at the beginning of the current line and highlighting is turned off. Following this action, the cursor is moved up or down as appropriate and highlighting is turned back on at the appropriate line. If the position from which the move is to occur is either the first or last menu line, then the cursor is moved to the last or first menu line, respectively. Lastly, if the carriage return or enter key was depressed, then the procedure is terminated and MenuLineNr is passed back to its storage variable for later use. The last statement before the line end (*procedure Menu*), i.e., RVideoOff, is required to stop reverse video from taking over the screen after you have left procedure Menu.

Procedure SetPrntr —

Dressing Up The Menu And Providing The Lines

The Menu procedure is designed to be called by another procedure, such as procedure SetPrntr which is where you will find it used in this program. Procedure SetPrntr has some escape codes defined in its constant section, only two of which have not been defined elsewhere. The codes unique to SetPrntr are used to turn the cursor off and on, positioning still works, you just don't see the cursor. The escape codes which are defined here, as well as in procedure Menu could have been defined as global codes to the whole program, but I didn't do it that way for two reasons: 1) clarity and 2) portability of the procedures, i.e., you can understand the procedure and you don't have to add extra baggage to your main program to get the procedures to work. Note also, that the constant section of SetPrntr is where you must specify the number of lines in your menu, as well as its position on the screen relative to the top and left side.

The only other esoteric declaration is the var declaration for array M. This array is used to pass the actual menu lines which will

appear on the screen to the menu procedure. It is the only type declaration in the program which must be declared in the program type section. Now on to the use of SetPrntr and Menu.

Procedure SetPrntr — Menu Information

A little housekeeping is performed first. We turn off the cursor, clear the screen and set the color to cyan on blue for the header line of the menu. The first line is printed followed by a move to the position of the explanatory portion of the menu and another color change to red on yellow. That information is printed using highlighting to make part of it stand out. For those of you unfortunate enough to own only a monochrome screen, you may rest assured that this entire set of changes for color and highlighting purposes will work on your machine by showing changes in intensity rather than color changes.

Procedure SetPrntr — Loading And Using The Menu

Next, the menu lines are loaded into array M, the color is once again changed to cyan on blue and procedure Menu is called. Following execution of procedure Menu a case statement is used to take the chosen menu line number and convert it to active codes for setting up your printer. These codes may differ for your printer, and therefore, changes may be required in this part of the program. If you plan to use this menu routine as I do, this is where you apply your choice from the menu to the outside world.

Procedure SetPrntr — Close Out

At this point, the color is changed back to white on black, the screen is cleared, the cursor is turned back on, and we are ready to move on to the last procedure of the program, ResetPrntr.

Procedure ResetPrntr

This procedure needs no comment except to say that the codes in it come from your printer manual. From here we proceed to the very short body of the program itself.

The Body

You thought we would never get here, right? Wrong. I'm just a bit slow and I made the assumption that the average reader would have as much difficulty understanding the arcane world of escape codes as I did.

The body of the program does nothing more than provide a setup to let you go around sending codes to a printer that isn't there, i.e., PrntrMenuLoc := 99;, or assuming you have one of the printers, it executes both Set and Reset Prntr followed by the program end statement.

Conclusion

This article was written because of the amount of trouble I had in figuring out how to use escape codes with TURBO Pascal. The program is a great deal more verbose than needed for operation, but it helps when you are trying to learn a subject such as this, or you are trying to modify someone else's code. Even though I find the use of escape codes a bit esoteric in TURBO, other versions of Pascal are a lot worse. I hope you get as much use out of your sparkling menus as I have out of this one.

Bibliography

TURBO Pascal, Version 3.0, Reference Manual; Borland International

Z-100 Technical Manual, TM-100; Heath/Zenith Data Systems
Epson MX Printer with Grafrax Manual; CompuSoft Publishing

Okidata 93 Printer Manual; Okidata Inc.

Toshiba 1350/1351 Printer Manual; Toshiba Inc.


```

program TestMenu(input,output,1st..
(* Copyright J R. Jeffrey, 2300 Lear Ln, Austin, TX 78745 *)
(* This program is part of a graphics procedure in a much larger cost risk*)
(* analysis program which I am writing. I converted it to show the use of*)
(* menus, and escape and control characters in TURBO Pascal *)

type
  MenuLine = array[1..10] of string[30]; (* declaration required *)
  (* external to Menu routines *)
var
  I,PrntrMenuLoc,MenuLineNr integer;
  (* ***** *)
  procedure Color(Fore,Back:integer);
  (* Sets foreground and background colors *)
  const
    Esc = #27;
  var
    Foreground,Background : string[1];
  begin
    str(Fore,Foreground);
    str(Back,Background);
    write(Esc + 'm' + Foreground + Background);
  end;
  (* ***** *)
  procedure Menu(MenuHorzPos,MenuVertPos,NrLines:integer);
  (* Provides for a Menu within a program - prints menu - moves highlighted *)
  (* cursor - allows selection of cursor line number for passing to later proc*)
  const
    Esc = #27, (* See pg 45, Version 3, TURBO ref man for use of # and ^[ *)
    RVideoOn = ^['p', (* See pg 10.47, TM 100 *)
    RVideoOff = ^['q', (* See pg 10.47, TM 100 *)
    First = 1,
    Up = 'A', (*See pg 10.60, TM100 Z100 powers up w/key expansion enabled*)
    Dn = 'B', (* The manual doesn't tell you this tho *)
    CR = #13;
  var
    InKey : char;
    MenuLineSelect,Quit : boolean;
  begin (* procedure Menu *)
    (* Print Menu *)
    for I := First to NrLines do
      begin
        gotoXY(MenuHorzPos,MenuVertPos+I*2); (* I*2 gives a space between lines *)
        write(M[I]);
      end;
    (* Set line number and highlight first entry *)
    MenuLineNr := 1;
    gotoXY(MenuHorzPos,MenuVertPos+MenuLineNr*2);
    write(RVideoOn,M[MenuLineNr],RVideoOff);
    MenuLineSelect := false; (* Set line selection to off *)
    (* Motion selection routine *)
    (* On InKey check for carriage return, uparrow, and downarrow. On carriage
    <return> set MenuLineSelect to true. On UpArrow turn off highlighting, move
    to entry above, highlight that entry, if at top go to bottom. On downarrow
    go down instead of up *)

```

```

repea;
  read(kbd,InKey); (* read one char from the keyboard *)
  if InKey = Esc then (* assumes key expansion enabled *)
  begin
    read(kbd,InKey); (* read second character *)
    if (InKey = Up) or (InKey = Dn) then
      begin
        gotoXY(MenuHorzPos,MenuVertPos+2*MenuLineNr); (* begin current line *)
        write(RVideoOff,M[MenuLineNr]); (* turn reverse video off *)
      end;
    case InKey of
      (* Checks each keyboard entry for arrows *)
      Up (* Up Arrow *) : if MenuLineNr = First
        then MenuLineNr := NrLines
        else MenuLineNr := MenuLineNr - 1;
      Dn (* Down Arrow *) : if MenuLineNr = NrLines
        then MenuLineNr := First
        else MenuLineNr := MenuLineNr + 1;
    end; (* case InKey *)
  end;
  gotoXY(MenuHorzPos,MenuVertPos+2*MenuLineNr); (* begin current line *)
  write(RVideoOn,M[MenuLineNr]); (* turn reverse video on *)
end; (* if InKey = Up or Dn *)
end (* if InKey = Esc *)
else if InKey = CR then MenuLineSelect := true;
until MenuLineSelect,
write(RVideoOff);
end; (* procedure Menu *)
(* ***** *)
procedure SetPrntr(MenuLineNr:integer);
const
  Esc = #27,
  CursorOff = ^['x5', (* See pg 10.48, TM100 *)
  CursorOn = ^['y5', (* See pg 10.48, TM100 *)
  RVideoOn = ^['p', (* See pg 10.47, TM100 *)
  RVideoOff = ^['q', (* See pg 10.47, TM100 *)
  NrLines = 4; (* Number of lines in the menu *)
  MenuVertPos = 6; (* Position of Menu relative to top of screen *)
  MenuHorzPos = 27; (* Position of Menu relative to left side of screen *)
var
  Quit boolean;
  M : MenuLine;
begin
  (* Menu to screen *)
  write(CursorOff);
  clrscr;
  Color(5,1);
  gotoXY(27,1);
  write('Printer Selection Menu');
  gotoXY(19,3);
  Color(2,6);
  write('Choose the type printer you have. ');
  gotoXY(19,4);
  write('Arrow keys move you in the Menu. ');
  gotoXY(19,5);
  write('<RETURN> selects the ',RVideoOn,'highlighted',RVideoOff,' item. ');

```

```

M[1] := 'Epson MX80';
M[2] := 'Toshiba 1350/1351';
M[3] := 'Okidata ML93';
M[4] := 'Quit (abort selection process)';

Color(5,i);

Menu(MenuHorzPos,MenuVertPos,NrLines,M,MenuLineNr i);

case MenuLineNr of
  1 . begin (* MX80 *)
    write(1st,chr(15)), (* Set compressed print mode *)
    write(1st,Esc + 'g'); (* Set vert spc to 8 lin/inch *)
  end;
  2 . begin (* Toshiba 1350/1351 *)
    write(1st,Esc + chr(91)), (* Set compressed print mode *)
    write(1st,Esc + chr(30) + chr(56));(*Set vert spc to 8 lin/inch*)
  end;
  3 . begin (* Okidata ML 93 *)
    write(1st,chr(29)); (* Set compressed print mode *)
    write(1st,Esc + '8'), (* Set vert spc to 8 lin/inch *)
  end;
  4 . Quit := true;
end, (* Case MenuLineNr *)
Color(7,0),
clrscr,
write(Cursor0n);
end, (* procedure SetPrntr *)

(* ***** *)
procedure ResetPrntr(MenuLineNr:integer);
const
  ResetEpson = ^['e',
  ResetToshiba = ^[#26#73;
  ResetOkidata = ^[#24;
begin
  case MenuLineNr of
    1 : write(ResetEpson); (* Reset Epson Printer *)
    2 : write(ResetToshiba); (* Reset Epson Printer *)
    3 : write(ResetOkidata); (* Reset Epson Printer *)
  end, (* case *)
end, (* procedure ResetPrntr *)

(* ***** *)
begin (* program TestMenu *)
PrntrMenuLoc := 99;
SetPrntr(PrntrMenuLoc);
writeln('***** BODY OF PROCEDURE OR PROGRAM GOES HERE *****');
if PrntrMenuLoc <> 99
then ResetPrntr(PrntrMenuLoc);
end

```

*



for the Z-151/161

WILDFIRE

from Software Wizardry

Speed The ultimate speed-up solution for the Z-151 and Z-161.*
With WILDFIRE your Z-151/161 will go faster than the 8 MHz Z-158.

Compatibility WILDFIRE surpasses processor chip additions for compatibility with the IBM PC and costs up to \$600 less.

Featuring A daughter board, higher speed replacement chips for the processor board, a high/low speed switch, an LED indicator to mount on the front panel, and hardware reset switch.

Price Software Wizardry brings you WILDFIRE, with attractive front panel, complete installation instructions, and reliable daughter-board design for only \$249.

Contact your dealer, or order direct from Software Wizardry by calling (314) 724-1738.



Software Wizardry, Inc.
1106 First Capitol Drive
St. Charles, MO 63301
(314) 724-1738.



IBM PC is a registered trademark of International Business Machines.
*WILDFIRE only works on Zenith Z-'51 and Z-161.

Patch Page

Pat Swayne

HUG Software Engineer

This article contains patches for H/Z-100 KEYMAP (885-3010-37), PeachText (H/Z-100 configuration), and the HUG File Manager (version 2).

KEYMAP Patch

The patch for KEYMAP that was presented in the April 1986 REMark contained a typographical error. This patch is to allow the MS-DOS 2 version to run on MS-DOS 3. To make the patch, create a file called KEYFIX.DAT that contains these lines:

```
AA3E
JNB A49
```

```
W
Q
```

Copy this file and DEBUG.COM to your KEYMAP disk, log on to the disk, and enter

```
DEBUG KEYMAP2.COM <KEYFIX.DAT
```

at the system prompt, and hit RETURN. Do it three more times, but with KEYWS2.COM, KEYBAS2.COM, and KEYSYS2.COM in the command line instead of KEYMAP2.COM. After those files are patched, they can be used with either MS-DOS version 2 or 3. The UNMAP program must also be patched. To patch it, make a file called UNFIX.DAT that contains these lines:

```
A106
JNB 111
```

```
W
Q
```

Copy this file and DEBUG.COM to your KEYMAP disk, log on to the disk, and enter

```
DEBUG UNMAP2.COM <UNFIX.DAT
```

at the system prompt, and hit RETURN. UNMAP2 will be patched for use with either MS-DOS version 2 or 3.

PeachText Patch

When PeachText is set up for use on an H/Z-100 (not PC), it will not work properly with the Hug Background Print Spooler, because it does not print via the DOS. To patch it so that it will work with the spooler, create a file called PTPCH.BAT that contains these lines:

```
REN PT.EXE PT.BIN
DEBUG PT.BIN <PTPCH.DAT
REN PT.BIN PT.EXE
```

Create a file called PTPCH.DAT that contains these lines:

```
A2E0C
MOV AH,5
INT 21
```

```
OR AL,AL
NOP
NOP
NOP
```

```
W
Q
```

Copy PTPCH.BAT, PTPCH.DAT, and DEBUG.COM to any PeachText disk containing PT.EXE, log on to the disk, and enter PTPCH

at the system prompt, and hit RETURN. Repeat the procedure for all other disks containing PT.EXE. **Note:** PeachText must have been fully installed for use on an H/Z-100 before you make the patches.

HUG File Manager Patch

The HUG File Manager (version 2), released as HFM2.COM on disk on 885-3025-37, and as HFM on the updated release of 885-3020-37, has an error in that the attribute designation is not correctly set up when the program accesses a disk directory. The result is that files may not be displayed correctly after certain operations, such as the Rename command. The patch presented here is for HFM2.COM or HFM.COM dated 6-07-85. To make the patch, create a file called HFMPCH.DAT containing these lines:

```
A32EC
MOV CX,0
NOP
```

```
W
Q
```

Copy this file and DEBUG.COM to your disk containing HFM.COM or HFM2.COM, log on to the disk, and enter

```
DEBUG HFM.COM <HFMPCH.DAT
```

or

```
DEBUG HFM2.COM <HFMPCH.DAT
```

at the system prompt, and hit RETURN. The patch will be installed. If you would like to patch the source code, load HFM.ASM OR HFM2.ASM into your editor and locate the label GET__DIR1:. Add a line after this label as shown below:

```
GET__DIR1
MOV CX,0
```

Reassemble the program, and it will work properly. If you would like to modify the HUG File Manager so that it will show hidden files in the directory display (that are not normally shown), use MOV CX,6 instead of MOV CX,0 in either HFMPCH.DAT or the source code. *

The H89 *SPEED* Center

4MHz mod

\$45

An easy to install plug-in module. No trace cutting or soldering. Speed may be toggled with software. Includes a replacement Z80A (4MHz). Includes CP/M software support for Heath, CDR Systems and Magnolia. Call or write for info on HDOS support. Specify disk format.

6MHz mod

\$59

Similar to our 4MHz modification, but increases the CPU speed to 6MHz. Requires some soldering on the CPU board. Includes a replacement Z80B (6MHz). May require replacing additional parts. Some technical knowledge is recommended for installation. CP/M support only. Specify disk format.

CDR Super RAM 89

Make your H89 a speed whiz with this board and the included RAM drive software. With our 6MHz mod and the RAM drive, the Heath BIOS assemblies (using MAKEBIOS) in less than 1 1/2 minutes! Note our discount prices!

Main board w/o RAM - specify disk format \$175
 Expansion board w/o RAM \$69
 Main board w/512K RAM \$239
 System w/MEGABYTE RAM \$369
 Clock option \$45
 SCSI option \$85

CDR Controllers

Double density disk controllers. Includes board, instruction manuals, ROMs and software. Allows control of 5 and 8 inch drives. Includes BIOS source code and some useful utilities. Please include your CP/M s/n when ordering.

For the H89 FDC-880H \$349

REP3 - Automatic Key Repeat

Stop wasting time - make full use of the repeat function just like the Z100. Simple plug-in installation on your keyboard assembly. Hold any key down for half a second and the key begins repeating. Combine this with our 4MHz mod and make WordStar fly! Provision for a defeat switch.

Kit \$35
 Assembled \$45

TIM2 - Real Time Clock

This circuit installs in the left hand expansion slots of the H89 or H89A. Can provide application programs with time and date information. Includes battery backup for continuous timekeeping when the computer is turned off. Program access to the clock is through an I/O port. The port address is user selectable by means of a jumper. Requires soldering 4 wires to the CPU board. A prepared ribbon cable is included.

Kit \$65
 Assembled \$75
 Software on disk - specify disk format \$10

WSPATCH

\$20

Adds H19/H89 function key patches to versions 3.0 or 3.3 of WordStar. Key functions similar to the PIE editor. Includes provision for redefining the keys by the user. Also includes a printer driver for the Epson MX80/FX80 printers.

DATESTAMPER

Product of Plu*Perfect Systems. Provides automatic time and date stamping for CP/M 2.2 files. Works with many real time clocks, including our own TIM2 product.

CP/M - specify disk format \$45

EMULATE

Allows the H89 to read/write to the following disk formats.

Actrix	Eagle II	Morrow MD	Superbrain Jr
AMPRO	Epson QX-10	NCR DecMate 5	Televideo
Beehive Tpr	Fujitsu CP/M86	NEC PC-8001A	TRS80-1 CP/M
CDR	IBM CP/M86	Osborne 1	TRS80-3 CP/M
Cromemco	IMS 5000	Otrona	TRS80-4 CP/M
DEC VT180	Kaypro II	PMC MicroMate	Visual 1050
DEC Rainbow	Magnolia	Royal/Adler	Xerox 820
		Sanyo 1100	Zorba

Now includes 44 formats! Uses a modified BIOS which is included with the program. Allows reading of 40-track disks in an 80-track drive.

For H37 with Heath CP/M \$59
 For CDR BIOS 2.91 \$49
 Check for Magnolia version.

NEWWORD - the better word processor from NewStar Software

Faster than WordStar, same commands as WordStar, MailMerge feature is built-in, supports more printers, has an undelete function and excellent documentation. Includes WORD Plus spelling checker by Oasis - normally a \$150 item by itself! We offer this package at discount and also include a function/cursor key patch for the H19/H89.

CP/M - specify format \$115
 MSDOS - specify computer \$195

DISKETTES

High quality private label diskettes at an economy price.

5" SSDD 10 hard sector \$16/box of ten
 5" DSDD \$18/box of ten
 8" SSDD \$26/box of ten
 8" DSDD \$29/box of ten

ANALYTICAL PRODUCTS

209/564-3687

20663 Ave. 352 Woodlake, CA 93286

Specify Disk Format on Software

CALL OR WRITE FOR CATALOG PRICES SUBJECT TO CHANGE

Terms. Check or Money Order - VISA/MC - C.O.D.

Add \$3 per order for shipping and handling

California residents add 6% tax

On The Leading Edge

ZPC And Other PC Emulators For The Z-100

William M. Adney

P.O. Box 531655
Grand Prairie, TX 75053

No review of PC emulators for the Z-100 series would be complete without talking about Pat Swayne's (HUG's) dynamite ZPC emulator. ZPC was developed last year and has matured into a rather spectacular (not to mention best selling) piece of HUG software. ZPC, and its hardware companion ZHS that appeared in the April issue of REMark, provide an inexpensive way to make the Z-100 quite compatible with a lot of PC software.

In addition to looking at ZPC, I have also included a summary of ZPC, Easy PC, and the Gemini Emulator Board. The intent of this summary is to help you answer the question: "Which emulator is best for me?" Plus I have another alternative that may surprise you.

For those of you who have the Z-150, no doubt you are finding articles about the PC emulation on the Z-100 to be somewhat boring. The good news is that this is the last of the planned articles on PC emulation (at least for a month). And as you will see, there are some new developments that promise to provide you with some additional information.

Introduction To ZPC

ZPC is certainly one of the most innovative software packages available on the market today. It provides a VERY clever approach to the subject of PC emulation on the Z-100, and we can thank HUG's intrepid Pat Swayne for that. As I have said before, there is always a price to pay when you try to "modify" the Z-100 so that it will run PC software, and ZPC is no exception to that. However, the most spectacular part of Pat's program is that he was able to provide most of the PC hardware features through the use of software — a remarkable approach that requires an in-depth understanding of both PC and Z-100 architecture, not to mention some of the sophisticated programming required to accomplish the end result.

ZPC is a memory resident utility that emulates the PC keyboard, printer I/O, disk I/O, and both the monochrome and color/graphics adapters of the IBM PC. The ZPC disk actually contains two different ZPC programs: ZPC.COM (for 768K memory) and

ZPCSM.COM (for systems with less than 768K memory). We will look at those in a minute.

The latest ZPC disk has a total of 38 files including 13 COM files. Twelve ASM files are included, 9 ACM files (assembler source for support), 2 PIC files for the DEMO program, a PATCHER.DAT file, and the always present README.DOC. The ZPC.ASM file is conditionally assembled to provide either ZPC.COM or ZPCSM.COM, so that is why there are only 12 ASM files to support 13 COM files.

The Hardware Configuration

Your hardware configuration is definitely the critical factor in determining what IBM PC software that you can run with ZPC. That is the reason that two ZPC programs (ZPC and ZPCSM) are provided on the disk. My H-100 has been upgraded to the new motherboard so that I have 768K present in three banks of 256K RAM chips. In order to realize the maximum PC compatibility available with ZPC, you must have exactly (and only) 768K of RAM in your system. That is one of the "prices" of PC compatibility with ZPC, although you still can have some PC emulation capabilities with less memory.

PC Memory Addressing

While that may seem to be an unduly harsh restriction for PC compatibility, you can thank the designers of the IBM PC for that. Most people know that IBM PCs have an "inherent" memory limit of 640K, but the reason for that is somewhat obscure. The reason for that limit is due to the fact that PCs use the memory above 640K as reserved memory. Part of that restriction is due to an inherent limitation in the 8088 CPU which includes a maximum of one megabyte of memory addressability and 64K of I/O addressability.

A quick look at the Z-150 Programmer's Reference Manual (page 6.10) shows a system memory map that tells us that memory above A0000H (655,360 decimal) is reserved for one thing or another. For example, monochrome video memory begins at B0000H (720,896 to 737,279 decimal), and color/graphics begins

at B8000H (737,280 to 753,663 decimal). Since 768K of RAM amounts to 786,432 decimal bytes, Pat has used the top 64K of Z-100 memory to emulate the PC monochrome and color/graphics capability of the IBM.

What does all of that technical discussion have to do with ZPC? The point is that an overwhelming percentage of the PC programs do NOT use BIOS or DOS routines for writing graphics and text displays. These programs write DIRECTLY to the video memory addresses and totally ignore the BIOS and DOS. For example, I found that the Z-150 WordStar program does this. Although the program seems to work, it is rather disconcerting to see a blank screen when you are expecting to see the main menu. Unless you have a photographic memory, you will find it impossible to make any file changes with the Z-150 WordStar on a blank screen.

The end result of all this is that unless you have exactly and only 768K RAM in your '100, you will only be able to run a small percentage of PC programs. So much for the memory restriction.

The ZPC Disk Programs

As previously mentioned, two ZPC programs are included on the disk for the different memory sizes. One of the nicest features of ZPC is that you can switch back and forth from the Z-100 mode and the PC mode. Once ZPC is loaded into memory, all you need to do is use PC.COM to activate the PC mode. Z100.COM is used to return to the Z-100 mode.

PATCHER.COM is used with PATCHER.DAT to patch programs that write to video memory on the IBM PC. Most of the popular programs access ports that do not exist on the Z-100. This program and data file are provided so that non-technical users can patch some programs without having any technical knowledge of assembly language. More on that later.

SETZPC is a configuration program that allows you to set ZPC defaults for video mode, font style, intense white and color emulation, blinking, normal monochrome color, graphic mode cursor on or off, Num Lock default, and emulated system memory size.

Since some PC programs require that the ANSI.SYS device driver be installed, the ANSISYS program loads that function. Once loaded, it can be activated or deactivated by the SETANSI program. My experience is that you won't need this feature very often, but it is a thoughtful addition.

Four more programs provide program fixes of one kind or another to allow PC programs to run under ZPC. You can patch a compiled GW-BASIC program to run under the small memory version of ZPC with FIXCB.COM. QuickBASIC programs can be patched to run under ZPC with FIXQB.COM. The Z-100 PSC (Print Screen) utilities will run under ZPC if they are patched with FIXPSC.COM. And FIXFWII.COM specifically patches Framework II so that it will run under ZPC.

Last, but certainly not least, is a dynamite demonstration program (DEMO.COM) that displays virtually all of the capabilities of ZPC. Although it works fine in monochrome, it is really outstanding in color. If you attended last year's HUG Convention in Chicago, you probably saw this demo at the HUG booth.

Running ZPC

ZPC is easy to set up and run. In order to run the DEMO program, here is the sequence of commands:

```
ZPC (Load memory resident ZPC program)
PC (Enter the PC mode)
```

```
ANSISYS (Load ANSI.SYS emulation)
DEMO (Run the DEMO program)
Z100 (Return to Z-100 mode)
```

That's all there is to it! All of this is contained in the first three pages of the documentation, so it is pretty easy to get ZPC up and running.

Documentation And ZPC Supported Programs

Documentation supplied with ZPC includes 27 pages of information plus the README.DOC file on disk. The documentation is excellent and describes just about everything you would want to know about the programs. In addition, a copy of PAT's April article on the ZHS (ZPC Hardware Support) board is also included. The ZHS provides the capability to run a number of programs without having to patch them first.

Since it is not altogether obvious what programs ZPC can support, I have shown the information provided in Appendix A of the ZPC documentation as Listing 1. Additional information will be published in REMark when it is available.

The following is a list of programs that have been tested under ZPC Version 2 (with 768K of memory, as of 3-18-86).

Program:	See Notes:
BENCHMARK Word Processor vers. 4.4	1
CORNERSTONE database	1
Compiled PC GW BASIC Programs	1
Compiled QUICKBASIC PROGRAMS	5
DAC EASY ACCOUNTING	5
DBASE III version 1.1	2, 3, 4
DBASE III + version 1.0	2, 3, 4
EDIX version 2.05	2, 3
EINSTEIN WRITER version 7.2	2, 3
ENABLE version 1.1	2, 3
FRAMEWORK version 1.1	2, 3, 4
FRAMEWORK II version 1.0	2, 3, 4, 5
GW BASIC (Zenith PC versions)	2, 3
LOTUS 1-2-3 release 1A	2, 3
LOTUS SYMPHONY	1, 4
MICROSOFT WORD vers. 1.1 (Zenith PC)	2, 3
MICROSOFT WORD version 2.0	1
MULTIMATE version 3.3	2, 3
MULTIPLAN version 1.2 (Zenith PC)	2, 3
NORTON UTILITIES	1
PC FILE	1
PC PALETTE version 1.0	3
PC WRITE version 2.4 or 2.55	3
PRINT MASTER	1
RUN/C	1
SIDEWAYS version 2.02	1
SUPERCALC3 version 2.02	3
TURBO PASCAL	1
VOLKSWRITER DELUXE version 2.0	3
WORD FINDER	1
WORD PERFECT version 4.1	2, 3

Notes:

1. Runs without any patches or hardware support.
2. Runs in the monochrome mode without any patches or hardware support.
3. Runs without patches if the ZPC Hardware Support circuitry is installed. Otherwise, you must use the patches supplied with ZPC.

- Copy protection must be removed before you can run this program.
- Requires a special patcher, supplied with ZPC.

Listing 1
ZPC Supported Programs (from ZPC documentation)

The ZPC Keyboard

Like all PC emulators, ZPC has keyboard emulation so that IBM PC key functions can be accessed. PC key functions and their Z-100 counterparts are shown in Listing 2.

PC Key	Z-100 Key(s)
Alt	HELP
Num Lock	F11
Scroll Lock	F12
Home	Keypad 7 or HOME
End	Keypad 1 or SHIFT-HOME
PgUp	Keypad 9 or D CHR
PgDn	Keypad 3 or I CHR
Ins	Keypad 0 or INS LINE
Del	Keypad . (period) or DEL LINE
Cursor Keys	Keypad 2/4/6/8 or Z-100 cursor keys
Keypad Plus (+)	SHIFT-Keypad minus (-)
PrtSc (unshifted)	SHIFT-F11
SHIFT-Tab (back tab)	SHIFT-HELP
CTRL-RETURN	LINE FEED
CTRL-SHIFT-RETURN	SHIFT F0 then LINE FEED
CTRL-Num Lock	F0 then F11 (See note 1)
CTRL-Break	F0 then F12 (See note 2)
Alt-Break	BREAK (See note 3)
SHIFT-PrtSc	SHIFT-F12 (See note 4)
CTRL-[F0 then [(See note 5)

Notes:

- This combination suspends program operation temporarily. It can be used like CTRL-S to suspend listing on the screen, except that it always works, where CTRL-S might not. After suspending operation,, type any key to resume.
- This combination is used to hold the execution of a program and return to the system. You can usually use CTRL-C instead.
- This combination empties the type-ahead buffer. The BREAK key by itself does this on a Z-100.
- This combination dumps the text on the screen to a printer. While ZPC is in the PC mode, this works even if you do not have a screen print utility loaded, but it is only a textual (not graphic) dump. If you want to dump graphics, load a Z-100 (not Z-150) graphic screen dump utility AFTER loading ZPC. Before you can use the Z-100 screen dump utility with ZPC, you must patch it. Just copy the utility and FIXPSC.COM (from the ZPC disk) to the system disk, log on to it, and enter: FIXPSC prog; where: prog is the name of the screen dump utility.
- CTRL-[is normally the same as an ESCape, but CTRL-[produces a different scan code from Esc, so you should use F0 and [when a program specifies CTRL-[.

Listing 2
ZPC Keyboard for PC Emulation

Z-100 Port Configuration

Because of the fact that you are essentially using the configured Z-100 BIOS, there are very few, if any, port changes that will need to be made to run most software. The first unit of each PC I/O device driver (COM1 serial and LPT1 parallel) is mapped to the PRN device on the Z-100 J-1. The second unit of each (COM2 serial and LPT2 parallel) is mapped to the AUX device on the Z-100 J-2.

The best news is that a PC program that is configured to use a parallel printer (normally LPT1) will actually use the device that was specified in the MS-DOS CONFIGUR command. I have MS-DOS and my trusty H-25 printer connected to J-1. Word Perfect 4.1 for the IBM PC seemed to have no problem with that since printing seemed to work normally after I changed the print parameters within Word Perfect for the H-25.

Running PC Software

I ran Word Perfect 4.1 in the monochrome mode, and there seemed to be no problems. I also happened to have a copy of Norton Utilities 3.1, and that also seemed to run with no problems as advertised. I did not attempt to do any "fixes" to the disk, but it was interesting to see Norton run under the standard Z-100 MS-DOS 3.1.

It is unfortunate that I do not have more PC software to test, but those two programs seemed to run just fine.

My Opinion Of ZPC

Performance of ZPC seemed to be on a par with what I normally expect in the standard Z-100 mode for either CP/M or MS-DOS. I did not do any specific benchmarks, but I didn't notice any performance degradation. Although I suspect that ZPC may slow down the Z-100 just a little, I certainly could not see it.

ZPC has a couple of special advantages that I haven't seen in the other emulators. The biggest advantage is that you don't have to buy another DOS. ZPC works under the standard Z-100 MS-DOS. I specifically tested programs using Z-100 MS-DOS Version 3.1 with no problems.

The other advantage is that you can switch between the PC mode and the Z-100 mode without rebooting your computer. Being able to easily move back and forth between both modes is a very significant advantage in my opinion. As I said before, Pat has developed a very sophisticated and clever program.

If you haven't built the ZHS board, perhaps the biggest disadvantage of ZPC is that PC software usually must be patched in order to run. In some cases, ZPC requires that existing copy protection be removed in order to run the program. This is specifically true for software that uses the Softguard protection scheme. Information about some utilities (i.e. UNLOCK and UNGUARD) that can remove the copy protection is listed in the ZPC documentation.

Speaking of the documentation, it is excellent. For those programs that are not specifically listed in Listing 1, the technical section describes in detail how to find and patch code for ZPC. There is a wealth of other information included in the pages. The information for both Listings and much of the technical information about IBM PCs was taken from the ZPC documentation.

If ZPC supports the program(s) that you want to run or you want to experiment, I can certainly recommend ZPC as a VERY inexpensive way to run popular PC software. If you purchased ZPC Version 1, Pat Swayne tells me that you can get the update to Ver-

sion 2 by sending in the original disk and \$20.00 to HUG as listed at the end of the article. If you have both ZPC Version 1 and the ZPC Support Disk, send in both original disks and \$15.00 for the upgrade.

PC Emulators — A Summary

At this point, there are apparently three choices if you want PC emulation of the Z-100. The Gemini Emulator Board was reviewed in the January issue. The Easy PC was reviewed last month. And we just finished talking about ZPC. What's the best?

As usual, that depends on exactly what YOU are looking for. If you are planning to do a lot of heavy duty work with PC software, there is one clear choice: the HS-148-41 kit. Whoa! That was not even in the running . . . was it? Of course it was! I specifically noted the requirement for heavy duty work with PC software.

The point is that, if you really want to get into the PC compatible software, you really should have a PC compatible. In addition, you can get that performance for \$899.00 (less HUG discount). In addition to the kit, you also receive the MS-DOS, and the Spring/Summer 1986 Heathkit catalog has a special so that you also get a FREE HVM-122A monochrome monitor. By the time you spend the money for the Easy PC plus the Z-150 MS-DOS operating system, you are within about \$20.00 or so of a COMPLETE PC compatible system. And of course, the HS-141 features two clock speeds: 4.77 Mhz (standard) and 8 Mhz for dynamite performance. Based on cost versus performance, the HS-148 kit is a clear winner . . . and the best alternative.

Perhaps you don't have the physical room or the inclination for another computer system, but you still have a fairly heavy requirement for PC software. In that case, I recommend the Gemini Emulator Board. As I said last time, the Easy PC (and I) destroyed the 8088 socket on my Z-100 motherboard twice in addition to wiping out 23 megabytes of data on my Winchester due to an Easy PC ROM problem. Even now, I am still not convinced that the Easy PC is ready for the field because I continue to hear about various installation and hardware problems. Although I am quite confident that UCI will correct all of the problems (they did send me a new system board), I remain unconvinced.

On the other hand, I had (and have) no problems with the Gemini on my H-100. I have the new board that runs at both 5 and 8 Mhz. I have been running it at 8 Mhz for about a month now with no problems. And the Gemini Sound Board provides PC audio emulation, so I have PC sound, too. Although the Gemini cannot quite do all of the video emulation, it certainly provides a remarkable degree of compatibility that should be satisfactory for almost everyone.

I also admit to a certain amount of technical prejudice in favor of the Gemini. I agree with Barry Watzman that the Gemini has a technical elegance that the Easy PC does not have. Furthermore, the Gemini requires much less hardware, and that usually has a higher degree of reliability and less maintenance costs because of the reduced amount of hardware.

ZPC is certainly one of the finest pieces of software that I have ever seen. And if you are so inclined, you can significantly improve performance by adding the ZHS board. For those of you interested in ZHS, Bob Eilerton tells me that Software Wizardry is considering making the ZHS available in the near future. Watch for the announcement in REMark about that.

From a technical perspective, I really like Pat Swayne's approach with ZPC since it is also a technically elegant and sophisticated approach to solving the PC compatibility problem on the Z-100. And from a price versus performance perspective, it is impossible to beat.

What Else Is New?

This summer promises to be extremely busy, and I have all kinds of new goodies coming up. From a personal perspective, I have finally decided to get the HS-241 kit with a 20 megabyte Winchester hard disk. I am NOT selling my H-100, however; I learned my lesson the last time and have been sorry ever since that I sold my trusty H-89. In any case, I will let you know about the '241, but I don't expect to get it for a couple of weeks yet.

I also have several other things that will be coming up this summer. I will be talking about at least one new product from Zenith, and I think that you will find those announcements quite interesting. In addition, I also have some announcements of a more personal nature that will be forthcoming this summer.

My consulting business (Adney & Associates) has been keeping me quite busy selling and writing about Heath/Zenith computers. One of the hazards of being a freelance consultant and writer is that there never seems to be enough time to do everything. I am now working eight days a week in an attempt to get things done, but it is clear that I will need at least a ten day week to even come close.

And by the way, if you have either Z-100 or PC hardware or software that you feel would be interesting to Heath and Zenith users, please let me know. I have found a number of good products that I regularly recommend to my consulting clients as a result of the vendor support for my articles over the last two and a half years. Hilgraeve's HyperAccess is probably the most outstanding example of a product that I have talked about in this column. You may also have noticed that HyperAccess has recently been added to the Heathkit catalog.

Help On The PC Cursor

One of the things that I have truly disliked about the IBM PC and compatibles is the blankety-blank blinking underline cursor. I have always thought that the underline cursor was difficult to see, and I really can't stand the blinking.

The first assembler program I ever wrote for my H-89 was to set the block, nonblinking cursor. I moved that over to my H-100 with no problem since the escape sequences are the same. Trying to do that on a PC is a slightly different story.

One of the nicest features of the IBM 3278 series display terminals is that you can change the cursor type by pressing a couple of keys. It's easy to change from underline to block and blinking to nonblinking. Not quite as trivial on the PC.

Getting the block cursor is no problem. All you have to do is use the Video I/O Interrupt 10H, set the AH register to one, and set the CH and CL registers for the starting and ending scan line, respectively. But I have yet to figure out how to stop the cursor from blinking. There must be some incredibly easy way to do this that I haven't been able to figure out.

In short, HEELPPPPP! Based on the letters that I get from you, there must be at least one person who knows how to do this. I've been fooling around with this using C and assembler, but all I've been able to do is get the blinking block cursor. I've asked a number of my local friends, but they don't know how to do it either.

So I decided to ask my national friends. And by the way, I will publish the program in this column giving credit to the first person who sends me the information or a working program.

Laptop Computers

Zenith has certainly had its share of good luck lately with the government contracts. And after reading about some of IBM's new products, it becomes clearer why Zenith has been chosen.

I am thinking about the IRS selection of the Z-171 laptop. If you haven't seen the IBM laptop, it is a disaster in my opinion. I doubt that anyone, including IBM, can force a new disk drive standard for the new 3-1/2" drives that come with their laptop. They also announced that 3-1/2" drive support would be available for the XT, but I'm not sure when that is supposed to be available.

Although a few vendors have announced 3-1/2" support, it is difficult to understand how that will help the six million IBM PCs and compatibles that are currently in the field. Since a large percentage of those are in the business environment, can you imagine the headaches for the microcomputer manager in a large company. I have recently had a number of calls from clients interested in the Z-171 for the simple reason that it has 5-1/4" disk drives. Of course, the publicity surrounding the IRS announcement helped quite a bit, too.

Aside from the 3-1/2" drive issue, the PC laptop is apparently something less than PC compatible. Given the idea that a laptop is for "on the road" use, it is difficult to understand why a laptop would be designed that is not totally PC compatible. That is also a reason that I have received lots of calls about the Z-171.

The Head In The Sand Trick

Bill Gates, Chairman of Microsoft, still has his head in the sand. You may remember that he made a comment a couple of years ago to the effect that he thought that 640K was the maximum amount of memory that anyone would EVER need. A number of people have remembered that and have criticized him ever since for that comment.

And he has done it again. A front page story in the April 28, 1986 issue of InfoWorld attributed the following to Bill Gates: "If someone doesn't think that 16 megabytes of memory is enough, I wish they'd tell us why".

Well son (Bill is younger than I am), let me tell you why. Let's take a look back through a short history of data processing, in general. I can remember when four megabytes of main memory in a mainframe computer was the ultimate. I can also remember when just one megabyte was considered the ultimate. Although some of the smaller mainframe computers today have a four megabyte main memory as standard equipment, big mainframes usually have some multiple of 16 megabytes of memory. Given enough money, you can buy a mainframe (e.g. IBM 3090) with 64 megabytes of main memory. This memory increase was a direct response to the need for faster and more powerful multiuser operating systems featuring a real-time multitasking capability.

And so it is with microcomputers. You thought that 640K was the ultimate four or five years ago. Today, you think that 16 megabytes is "enough". It won't be. Yesterday, memory was designed for 4K, then 16K, then 64K, and now 256K memory chips. Tomorrow, memory will be designed for one megabyte chips and beyond. If the history of both mainframes and micros doesn't convince you, there is at least one other factor to be considered.

Current reports indicate that the 80386 will probably be the CPU of the future. The 80386 has a four gigabyte direct address space — 4,000 megabytes. Why in the world would anyone want to impose a measly artificial limit of 16 megabytes on a CPU with those limits? Why not design your new DOS 286/386 to handle the maximum limit of the CPU?

Computers have been around for years. Fortunately or unfortunately, I have already seen most of these mistakes made in the mainframe environment. Copy protected software is another one of those "experiments" that fizzled out in the mainframe environment. Users would simply not stand for it because of backup requirements among other things. Although copy protection is not dead in the microcomputer arena, many enlightened vendors are removing copy protection from their products and advertising the fact.

Although published reports indicate that Lotus FINALLY agreed to remove copy protection from Lotus 1-2-3 for sale to the government only, it was probably too little too late. Based on reports about Zenith's sale of Z-200's to the military, it appears that Lotus 1-2-3 was not in the contract. In case you did not know, the government does NOT buy copy protected software.

I wish that software vendors would check with some of us who have had experience in mainframe computers. Many of the "problems", and not a few of the solutions, have been encountered in mainframes. It somehow seems that they could take a lesson from past history, but I suppose that is too much to expect.

IBM's "New" Keyboard

One of the other interesting announcements of late was IBM's "enhanced" keyboard. It has, among other things, twelve function keys! IBM has announced that this keyboard will be standard on all future XTs and ATs, and it's being shipped with new XTs and ATs that were introduced in April.

Many people are criticizing IBM for the twelve function keys. They have obviously never used an IBM 3278 series display terminal which has always had at least 12 function keys (some have 24). Instead of being on the left side of the keyboard, the function keys are now located across the top like the '100. For my part, I've always been a little puzzled as to why IBM did not have 12 function keys on the original PC keyboard since that was a well established standard when the PC was announced. Even the '100s have F1 through F12 plus a number of others.

Aside from the function keys, the keyboard itself has undergone considerable rearrangement. The CTRL key is no longer in the "standard" position next to the "A" key — it's now on the bottom. The CAPS key has been moved to the CTRL key location so that the keyboard is similar to a typewriter in that respect. The standard position of the ESCape key has also been changed. It is not clear why IBM felt compelled to change the keyboard at this late stage of the game.

The keypad has undergone significant change. All of the "function" keys like Home, PgUp, etc. (and the cursor keys) have been moved to an area between the main keyboard and the new keypad. The Num Lock and Scroll Lock keys are located at the top of the keyboard. The keypad now boasts an ENTER key plus an asterisk (for multiplication) and a slash (for division).

Keyboard changes are enough to strike terror into any touch typists heart. Even though this keyboard change is clearly in line

with William Lowe's stated objective of "making it more difficult for clone makers to keep up with IBM", the change to the "standard" keyboard was not well advised. Software vendors have already taken public exception to the additional function keys for several technical reasons. Even though some of the changes were really improvements, as far as touch typists are concerned, the change was simply too radical.

Since Zenith will clearly have to add the additional function keys to their PC compatible series at some point, I trust that they will be able to come up with a better keyboard than IBM. It appears that IBM forgot the fundamental concept of keyboard design — the keyboard IS the user interface to the computer. Clumsy keyboard design with non-standard key locations is not in the best user interest.

Lest you become too concerned about the keyboard change with respect to Heath and Zenith computers, I wouldn't worry about it. There are about six million IBM PCs and compatibles in the field right now. Current software supports the existing function keys. Software vendors will be reluctant to ignore this existing market and add the new function keys to their software so that the new keyboard is a required item. In short, I doubt that you will see much software that utilizes these new function keys in the next few years.

In The Future . . .

I will be speaking at the HUG Convention again this year. The topic is on "MS-DOS Version 3 and Windows". That includes the DOS changes and additions for both the Z-100, as well as the Z-150 (and Z-200) PC series. We will look at the new commands for Version 3, as well as changes to existing Version 2 commands. An introduction to MS-DOS Windows will also be included in the article and presentation. I hope to see you in Chicago.

Products Discussed

HUG Software	
ZPC Version II (885-3037-37)	\$ 60.00
Heath/Zenith Computer Centers	
Heath/Zenith Users' Group	
Attn: Nancy Strunk	
Hilltop Road	
St. Joseph, MI 49085	
(616) 982-3571 (HUG Software only)	
HyperACCESS (all Operating Systems)	\$149.95
ACCESS	
H-8, H/Z-89 (CP/M-80)	49.95
Z-100 (MS-DOS, CP/M-85, CP/M-86)	59.95
PC only (MS-DOS, CP/M-86)	69.95
Heath/Zenith Computer Centers	
Hilgraeve, Inc.	
P.O. Box 941	
Monroe, MI 48161	
(313) 243-0576	
WordPerfect 4.1	\$495.00
SSI Software	
228 West Center Street	
Orem, UT 84057	
(801) 227-4000	
Software	
HyperACCESS (PC only) (PM-160)	\$149.95
MS-DOS Version 3	

Z-100 only (OS-63-30)	\$150.00
PC only (OS-63-31)	150.00
Programmer's Utility Pack (CB-3163-30)	150.00
Microsoft Windows	
PC only (MS-5063-30)	\$ 99.00
WordStar Professional	
Z-100 only (MP-463-17)	\$399.00
PC only (MP-5063-13)	399.00

Hardware

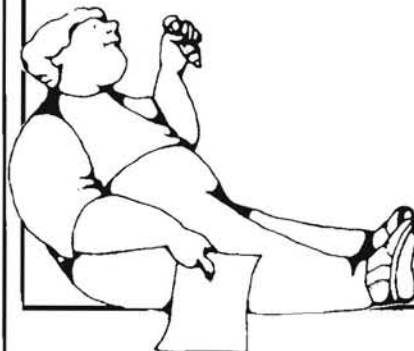
Advanced Personal Computer (HS-241)	\$2899.00
Video Card (Z-409)	239.00
20MB Winchester (ZD-200)	1499.00
Compact Personal Computer (HS-148-41)	899.00
H-100 Desktop Computer (HS-1108-41)	1599.00
Advanced Portable Computer (ZF-171-42)	2399.00
Easy PC Emulator (PC-250)	699.00
Z-150 MS-DOS (OS-63-31)	90.00
	(with PC-250 only)
Gemini Emulator Board (PC-251)	599.00
Z-150 MS-DOS (OS-63-31)	90.00
	(with PC-251 only)
Monochrome Monitor (HVM-122A)	79.95

Heath/Zenith Computer Centers
 Heath Company Parts Department
 Hilltop Road
 St. Joseph, MI 49085
 (800) 253-7057 (Heath Catalog orders only)



Have You Ever Asked?

1. How can I retrieve programs from the HUG Bulletin Board?
2. How can I tell what programs to choose from the hundreds available?
3. How can I get a quick reference to the commands available on the HUG Bulletin Board?
4. How can I communicate with other HUG members that happen to be on-line on the HUG Bulletin Board with me?



Get The HUG
 Bulletin Board
 Handbook
 (PN 885-4700)

It's \$5.00 from the
 Heath Parts Dept.

HUG Price List

The following HUG Price List contains a list of all products in the HUG Software Catalog. For a detailed abstract of these products, refer to the issue of REMark specified.

Part Number	Description of Product	Selling Price	Vol. Issue	Part Number	Description of Product	Selling Price	Vol. Issue	Part Number	Description of Product	Selling Price	Vol. Issue
HDOS HARDCOPY SOFTWARE				885-1089-[37]	Disk XVIII Misc H8/H89	20.00	20	885-3029-37§§	ZDOS/MSDOS HUG Bg. Print Spool	20.00	66
885-1008	Volume I Documentation	9.00		885-1090-[37]	Disk XIX Utilities H8/H89	20.00	22	885-3032-37§	MSDOS Halley's Comet Locator	20.00	70
885-1013	Volume II Documentation	12.00		885-1092-[37]	Relocating Debug Tool H8/H89	30.00	14	885-3035-37§§	MSDOS SPELL5 & SPELL5F	20.00	72
885-1015	Volume III Documentation	9.00		885-1098	H8 Color Graphics ASM	20.00	19	885-3038-37§	ZDOS/MSDOS DEBUG Support Util	20.00	77
885-1037	Volume IV Documentation	12.00	8	885-1099	H8 Color Graphics Tiny PASCAL	20.00	19	885-8039-37§§	MSDOS DPATH	20.00	74
885-1058	Volume V Documentation	12.00		885-1105	HDOS Device Drivers H8/H89	20.00	24	885-8040-37§§	MSDOS HELP Programs	20.00	74
MISCELLANEOUS HDOS COLLECTIONS				885-1116	HDOS Z80 Debugging Tool	20.00	27	§ All program files run on both §§ Program files run partially on both			
885-1032	Disk V H8/H89	18.00	8	885-1119-[37]	BHBASIC Support	20.00	29	PC/IBM COMPATIBLE			
885-1044-[37]	Disk VI H8/H89	18.00		885-1120-[37]	HDOS 'WHEW' Utilities	20.00	33	885-6001-37	MSDOS Keymapper	20.00	59
885-1064-[37]	Disk IX H8/H89 Disk	18.00		885-1121	HDOS Hard Sec Sup Pkg 2 Disks	30.00	37	885-6002-37	CP/EMulator II & ZEMulator	20.00	59
885-1066-[37]	Disk X H8/H89	18.00	10	885-1123	XMET Robot Cross Assembler	20.00	40	885-6003-37	MSDOS EZPLOT	20.00	65
885-1069	Disk XIII Misc H8/H89	18.00		885-1126	HDOS Utilities by PS:	20.00	42	885-6004-37	MSDOS CheapCalc	20.00	67
885-1135-[37]	HDOS Variety Pkg.	20.00	76	885-1127-[37]	HDOS Soft Sector Support Pkg	30.00	45	885-6005-37	MSDOS Screenviews	20.00	67
GAMES				885-1128-[37]	HDOS DISKVIEW	16.00	46	885-6006-37	MSDOS Cardcat	20.00	69
HDOS				885-1129-[37]	HDOS CVT Color Video Terminal	20.00	46	885-6007-37	MSDOS OND (Dung. & Dragons)	20.00	70
885-1010	Adventure Disk H8/H89	10.00	4	885-8001	SE (Screen Editor)	25.00	28	885-6009-37	MSDOS Screen Saver Plus	20.00	76
885-1029-[37]	Disk II Games 1 H8/H89	18.00	8	885-8003	BHTOMB	25.00	28	885-8033-37	MSDOS Fast Edit	20.00	62
885-1030-[37]	Disk III Games 2 H8/H89	18.00	8	885-8004	UDUMP	35.00	28	885-8037-37	MSDOS Grade	20.00	70
885-1031	Disk IV MUSIC H8 Only	20.00	25	885-8006	HDOS SUBMIT	20.00	31	PROGRAMMING LANGUAGES			
885-1067-[37]	Disk XI H8/H19/H89 Games	18.00	12	885-8007	EZITRANS.	30.00	30	HDOS			
885-1068	Disk XII MBASIC Graphic Games	18.00	10	885-8015	HDOS TEXTSET Formatter	30.00	42	885-1038-[37]	Wise on Disk H8/H89	18.00	
885-1088-[37]	Disk XVII MBASIC Graph. Games	20.00	14	885-8017	HDOS Programmers Helper	16.00	42	885-1042-[37]	PILOT on Disk H8/H89	19.00	
885-1093-[37]	D&D H8/H89 Disk	20.00	16	885-8024	HDOS BHBASIC Utilities Disk	16.00	46	885-1059	FOCAL-8 H8/H89 Disk	25.00	13
885-1096-[37]	MBASIC Action Games H8/H89	20.00	18	CP/M				885-1078-[37]	HDOS Z80 Assembler	25.00	21
885-1103	Sea Battle HDOS H19/H8/H89	20.00	20	885-1210-[37]	CP/M ED (same as 885-1022)	20.00	20	885-1085	PILOT Documentation	9.00	
885-1111-[37]	HDOS MBASIC Games H8/H89	20.00	23	885-1212-[37]	CP/M Utilities H8/H89	20.00	21	885-1086-[37]	Tiny HDOS PASCAL H8/H89	20.00	13
885-1112-[37]	HDOS Graphic Games H8/H89	20.00	23	885-1213-[37]	CP/M Disk Utilities H8/H89	20.00	22	885-1094	HDOS Fig-Forth H8/H89	40.00	18
885-1113-[37]	HDOS Action Games H8/H89	20.00	23	885-1217-[37]	HUG Disk Duplication Utilities	20.00	26	885-1132-[37]	HDOS Tiny BASIC Compiler	25.00	59
885-1114	H8 Color Raiders & Goop	20.00	23	885-1223-[37]	HRUN HDOS Emulator 3 Disks	40.00	37	885-1134	HDOS SMALL-C Compiler	30.00	63
885-1124	HUGMAN & Movie Animation Pkg	20.00	41	885-1225-[37]	CP/M Disk Dump & Edit Utility	30.00	40	CP/M			
885-1125	MAZEMADNESS	20.00	41	885-1226-[37]	CP/M Utilities by PS:	20.00	40	885-1208-[37]	CP/M Fig-Forth H8/H89 2 Disks	40.00	18
885-1130	Star Battle	20.00	45	885-1229-[37]	XMET Robot Cross Assembler	20.00	40	885-1215-[37]	CP/M BASIC-E	20.00	26
885-1133-[37]	HDOS Games Collection I	20.00	59	885-1230-[37]	CP/M Function Key Mapper	20.00	42	BUSINESS, FINANCE AND EDUCATION			
885-8009-[37]	HDOS & CP/M Galactic Warrior	20.00	32	885-1231-[37]	Cross Ref Utilities for MBASIC	20.00	43	HDOS			
885-8022	HDOS SHAPEES	16.00	45	885-1232-[37]	CP/M Color Video Terminal	20.00	46	885-1047	Stocks H8/H89 Disk	18.00	
885-8026	HDOS Space Drop	16.00	49	885-1235-[37]	CP/M COPYDOS	20.00	54	885-1048	Personal Account H8/H89 Disk	18.00	
885-8032-[37]	HDOS Castle	20.00	59	885-1237-[37]	CP/M Utilities	20.00	55	885-1049	Income Tax Records H8/H89 Disk	18.00	
CP/M				885-1245-37	CP/M-85 KEYMAP	20.00	63	885-1055-[37]	MBASIC Inventory Disk H8/H89	30.00	
885-1206-[37]	CP/M Games Disk	20.00	11	885-1246[-37]	CP/M HUG File Manager & Utilities	20.00	64	885-1056	MBASIC Mail List	30.00	
885-1209-[37]	CP/M MBASIC D&D	20.00	19	885-1247-37	CP/M-85 HUG Bkgnd Print Spooler	20.00	67	885-1070	Disk XIV Home Fin H8/H89	18.00	
885-1211-[37]	CP/M Sea Battle	20.00	20	885-5001-37	CP/M-86 KEYMAP	20.00	51	885-1071-[37]	MBASIC SmbusPk H8/H19/H89	75.00	17
885-1220-[37]	CP/M Action Games	20.00	32	885-5002-37	CP/M-86 HUG Editor	20.00	52	885-1091-[37]	Grade/Score Keeping H8/H89	30.00	14
885-1222-[37]	CP/M Adventure	10.00	35	885-5003-37	CP/M-86 Utilities by PS:	20.00	54	885-1097-[37]	MBASIC Quiz Disk H8/H89	20.00	18
885-1227-[37]	CP/M Casino Games	20.00	38	885-5008-37	CP/M 8080 To 8088 Trans. & HFM	20.00	64	885-1118-[37]	MBASIC Payroll	60.00	30
885-1228-[37]	CP/M Fast Action Games	20.00	39	885-5009-37	CP/M-86 HUG Bkgnd Print Spool	20.00	66	885-1131-[37]	HDOS CheapCalc	20.00	47
885-1236-[37]	CP/M Fun Disk I	20.00	55	885-8018-[37]	CP/M Fast Eddy & Big Eddy	20.00	43	885-8010	HDOS Checkoff	25.00	32
885-1248-[37]	CP/M Fun Disk II	35.00	69	885-8019-[37]	DOCUMAT and DOCULIST	20.00	43	885-8021	HDOS Student's Statistics Pkg	20.00	44
ZDOS				885-8025-37	CP/M-85/86 Fast Eddy	20.00	49	885-8027	HDOS SciCalc	20.00	50
885-3004-37	ZDOS ZBASIC Graphic Games	20.00	37	ZDOS/MSDOS				CP/M			
885-3009-37	ZDOS ZBASIC D&D	20.00	50	885-3005-37	ZDOS Etchdump	20.00	39	885-1218-[37]	CP/M MBASIC Payroll	60.00	31
885-3011-37	ZDOS ZBASIC Games Disk	20.00	52	885-3007-37	ZDOS CP/EMulator	20.00	47	885-1233-[37]	CP/M CheapCalc	20.00	47
885-3017-37	ZDOS Contest Games Disk	25.00	58	885-3008-37	ZDOS Utilities	20.00	47	885-1239-[37]	Spread Sht. Contest Disk I	20.00	
885-8042-37	ZDOS/MSDOS Poker Party	20.00	77	885-3010-37	ZDOS Keymap	20.00	51	885-1240-[37]	Spread Sht. Contest Disk II	20.00	
UTILITIES				885-3022-37	ZDOS/MSDOS Useful Programs I	30.00	63	885-1241-[37]	Spread Sht. Contest Disk III	20.00	
HDOS				885-3023-37	ZDOS/MSDOS EZPLOT	20.00	63	885-1242-[37]	Spread Sht. Contest Disk IV	20.00	
885-1022-[37]	HUG Editor (ED) Disk H8/H89	20.00	20	885-3026-37	MSDOS SMALL C Compiler	25.00	65	885-1243-[37]	Spread Sht. Contest Disk V	20.00	
885-1025	Runoff Disk H8/H89	35.00		885-3030-37	ZDOS/MSDOS Z-100 PC Emulator	40.00	68	885-1244-[37]	Spread Sht. Contest Disk VI	20.00	
885-1060-[37]	Disk VII H8/H89	18.00		885-3031-37	ZDOS/MSDOS Graphics	20.00	69	885-8011-37]	CP/M Checkoff	25.00	32
885-1061	TMI Load H8 ONLY Disk	18.00		885-3034-37	ZDOS/MSDOS ZPC Support Pkg	10.00	72	885-8036-[37]	CP/M Grade	20.00	70
885-1062-[37]	Disk VIII H8/H89 (2 Disks)	25.00		885-3037-37	MSDOS ZPC II	60.00	76	ZDOS			
885-1063	Floating Point Disk H8/H89	18.00		885-8029-37	ZDOS Fast Eddy	20.00	53	885-3006-37	ZDOS CheapCalc	20.00	47
885-1065	Fix Point Package H8/H89 Disk	18.00	10	885-8035-37	MSDOS DOCUMAT and DOCULIST	20.00	70	885-3013-37	ZDOS Checkbook Manager	20.00	54
885-1075	HDOS Support Package H8/H89	60.00		885-8041-37	ZDOS/MSDOS Orbits	25.00	75	885-3018-37	ZDOS Contest Spreadsheet Disk	25.00	58
885-1077	TXTCON/BASCON H8/H89	18.00		H/Z100 ZDOS/MSDOS - H/Z150 PC MSDOS				885-8028-37	ZDOS SciCalc	20.00	50
885-1079-[37]	HDOS Page Editor	25.00	15	885-3012-37§§	ZDOS HUG Editor	20.00	52	885-8030-37	ZDOS SciCalc	20.00	50
885-1080	EDITX H8/H19/H89 Disk	20.00		885-3014-37§§	ZDOS/MSDOS Utilities II	20.00	54	885-8030-37	ZDOS Mathflash	20.00	55
885-1082	Programs for Printers H8/H89	20.00		885-3016-37§	ZDOS/MSDOS Adventure	10.00	57	Continued on Page 83			
885-1083-[37]	Disk XVI Misc H8/H89	20.00	11	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				

Performance.

Introducing EasyPC™ from UCI. The most advanced IBM PC emulator you can buy for your Z-100. First to provide the compatibility, speed and audio support to make your Z-100 run like a PC. Since the EasyPC provides complete PC circuitry, you actually run PC programs on PC hardware. Quite simply, it's like installing a PC right inside your Z-100. In the PC mode, your Z-100 can run virtually the entire library of PC programs, including LOTUS 1-2-3, dBase III, FLIGHT SIMULATOR,

and most other copy-protected software. In the Zenith mode, your Z-100 is a Z-100, with no change in performance.

EasyPC is fast. In fact, it is noticeably faster than both the Z-150 and the IBM PC. And EasyPC can run at either 5 or 8 MHz. It supports both floppy and Winchester drives, provides crisp color video graphics, supports PC-compatible sound generation, and even comes with its own self mounting speaker.

Quality.

You already know and appreciate the kind of quality and reliability that is built into your Z-100. UCI matches that commitment to quality from first step to last.

Sophisticated engineering and quality components combined with precision automatic assembly, component burn-in, and both in-circuit

and full functional testing, mean superior performance and long term reliability.

EasyPC's advanced circuitry utilizes two, high quality, S-100 multilayer boards and a piggyback board. Yet its ingenious design requires only one extra slot. So you get all the advantages of PC compatibility, with room to spare.

Availability.

EasyPC is ready when you are. If you've been waiting too long for PC compatibility, your wait is over. EasyPC is here now. Complete documentation is included for quick and easy installation,

so you'll be up and running fast. Your only problem will be which PC programs to get first.

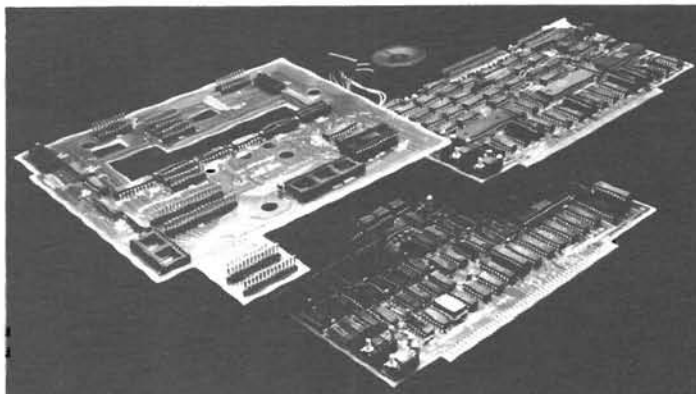
EasyPC is available through your independent Zenith dealer. Ask him for a demonstration.

Price.

At \$699, you'll pay a little more. But you'll get the best. Total compatibility. High speed performance. Unsurpassed quality and serviceability. In a word, value. You also get UCI's one year Solid

Service Guarantee and full customer support. If you have any questions or problems with any UCI product, or need the name of your nearest dealer, just call 800-UCI-COMPUTER.

EasyPC™



Full IBM PC emulation for your Z-100.

UCI CORPORATION
848 Cherry Street, Kent, Ohio 44240
(216) 673-5155 • 800-UCI-COMPUTER

EasyPC is a trademark of UCI, Inc. Zenith and Z-100 are trademarks of Zenith Electronics Corp. IBM and IBM PC are trademarks of International Business Machines. Lotus 1-2-3 is a trademark of Lotus Development Corp. dBase III is a trademark of Ashton-Tate. Flight Simulator is a trademark of Microsoft Corp.

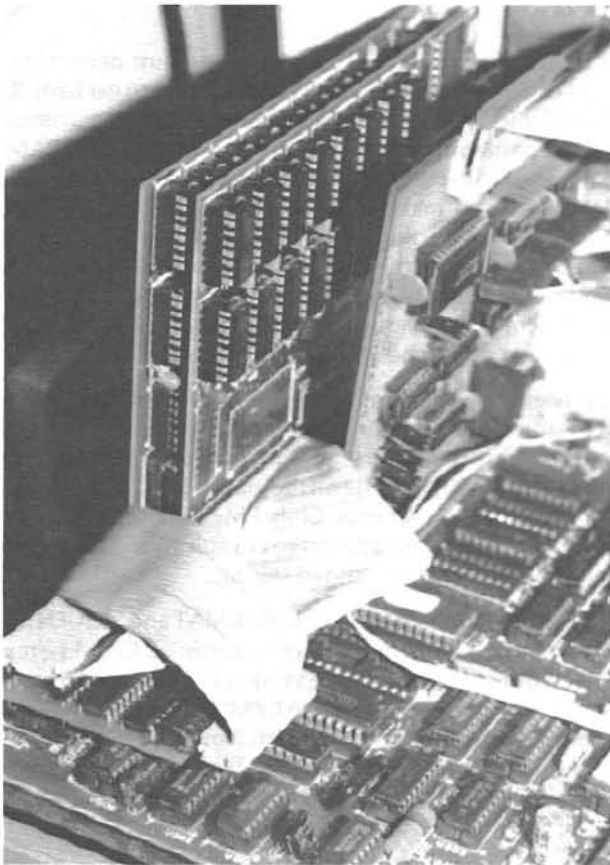


Photo 1
Super RAM 89 installed in the H/Z-89.

C.D.R.'s Super RAM 89 (II)

In the previous article, I presented an overview of Super RAM 89, a unique multi-faceted enhancement for the H/Z-89 computer system that adds (on two piggy-back boards that plug into one left-side expansion slot) 1-Megabyte of RAM, a DMA controller, a Clock, and the latest in hard disk interfacing technology — the SCSI bus.

This concluding portion will discuss the pros and cons of the software and documentation, and provide you with some background on the development of Super RAM 89.

The Creation Of Super RAM

According to Marc Brooks, this product was not on C.D.R.'s list of priorities. They were designing a new hard disk controller for the Heath/Zenith Z-100 computer series that incorporated the new SCSI bus. They had a variety of hard disk packages available for the H/Z-89 that used the Magnolia 77320 SASI host adaptor and CP/M, and the standard Xebec and Adaptec controllers with assorted Winchester drives, which they felt were adequate.

But as their work on the Z-100 controller proceeded, they began to succumb to the suggestion of L. D. Haag and John Allen (and other members of the San Diego HUG) that they do a RAMdisk board for the H/Z-89 that went beyond the 128k limit of the Magnolia and FBE Research boards. Early in 1985, as the prices of 256k RAMs began to plunge, a RAMdisk board began to appear more and more as a viable product that could be produced at a realistic price.

A Winchester For The '89

Part Six

Peter Ruber

P.O. Box 502
Oakdale, NY 11769

As the design criteria took shape, the Brooks' began to realize that the design principles they were incorporating in the Z-100 SCSI card could readily be adapted to the emerging Super RAM 89 by simply adding a DMA controller and a few extra parts.

By this time, too, the number of people involved in the project had also grown. L. D. Haag was assigned the job of developing the CP/M software. Haag, a long-time user of the H/Z-89 and related C.D.R. products, is a Master Chief and Electronic Warfare Technician for the U.S. Navy, currently the supervisor and course curriculum model manager for basic electricity and electronics schools for the U.S.N.

Haag also assisted in the initial bread-boarding of the project that was dubbed RAM 89. Eugene Skokal of Copley Computer Services and Peter Shkabara of Analytical Products (a C.D.R. dealer) contributed the Clock design and software. Skokal's firm also began to develop networking software for the SCSI implementation that would ultimately allow a group of H/Z-89s to share the SCSI bus and use high storage drive and tape media. Marc Brooks took on the task of developing the driver software for Super RAM 89 that would enable it to interface with the Xebec and Adaptec hard disk controllers.

The final design, layout and hardware implementation was done by Herm Brooks, who also supervised the entire project. Testing was assigned to John Allen, and C.D.R.'s technician, Steve Devlin. Super RAM 89 was released with CP/M in June of 1985.

Since, C.D.R. does not use HDOS for in-house purposes, they contracted with Alan L. Heigl of Mill City Records, Minneapolis, MN, to develop the necessary drivers and supporting software. Heigl is a software designer with many years of experience and a former employee of the Heath Company. He managed to imple-

ment more capabilities into the HDOS drivers than C.D.R. thought possible. The HDOS software became available in October 1985, about the same time as the SASISOFT hard disk software for the Adaptec ACB-4000 was released. The SASISOFT utilities for the Xebec S1410/a controllers were released in January 1986.

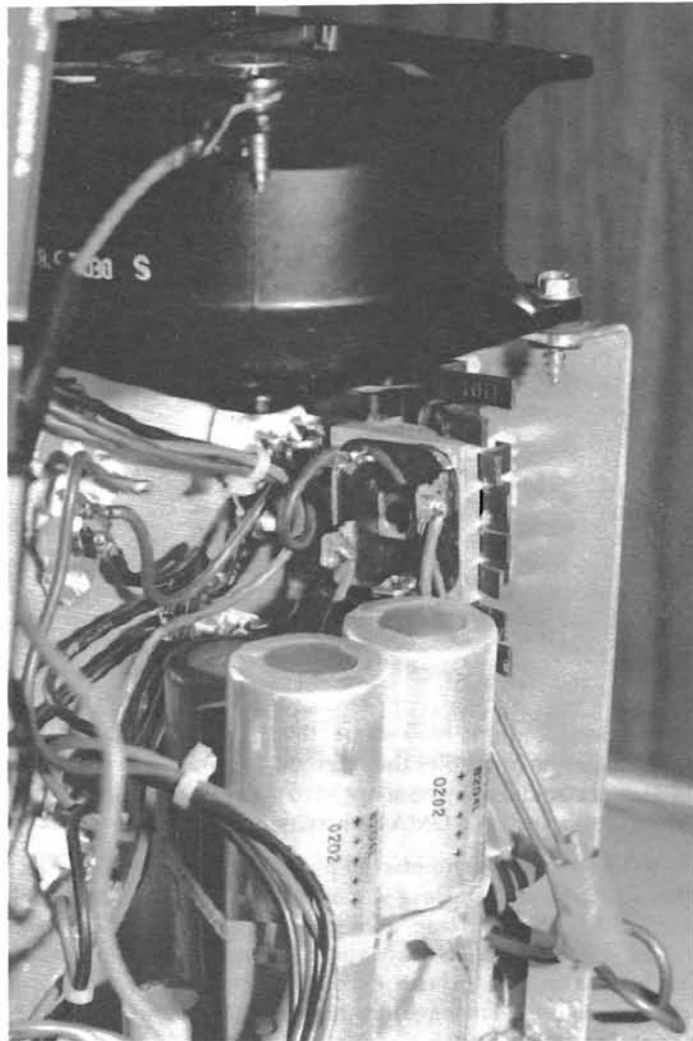


Photo 2

In the first segment of the Super RAM 89 review, I mentioned having replaced Heath's 10 Amp 25 PIV bridge rectifier with a 25 Amp 50 PIV unit and installing it on the replacement heatsink assembly available from Technical Micro Systems (P.O. Box 7227, Ann Arbor, MI 48107). Note that the bridge rectifier was moved from its heat generating position on the lower right to the back panel of the new heatsink assembly. The fan was moved from its original position on the inside of the top panel and now mounts directly on the new heatsink forcing air down on the power supply components. The result is a cool, quiet and highly stable power supply. If your H/Z-89 exhibits erratic screen behavior or generates too much heat in the power supply area when your system is over-stuffed with expansion boards, you might well consider this modification.

CP/M For Super RAM 89

All the software for SR-89 is easy to use. CP/M, which usually requires that devices be incorporated into the BIOS through the painfully slow MAKEBIOS procedure, spares us on this occasion

as the file INSRAM.COM enables us to attach the RAMdisk parameters by just answering a few questions in a menu-driven format.

Prior to installing the CP/M software, you must determine the amount of Global Memory you wish to allocate to each 64k RAM bank available on the RAMdisk. You are allowed a maximum of two RAM partitions as long as you have at least 256k RAM installed on your SR-89. I should note that bank switching is automatic in order to make each RAMdisk partition available as contiguous RAM. Partition 0 will always be one 64k bank larger than Partition 1. Since the RAM on the two SR-89 boards can be an intermingling of 64k and 256k RAM chips, the software automatically senses the amount of RAM available and will display it on the screen each time you boot the system.

Global Memory is generally called "common memory" that is used to store data common to all the banks — such as the Z-80 stack data. You establish the amount of Global Memory by setting the jumpers at JJ1 and JJ2 on the main board. Standard CP/M 2.2 requires that you set your Global Memory for 1k. If you are using CP/M 3.0, you are advised to consult your system manual to determine the recommended setting.

The next step is to take a fresh disk, FORMAT and SYSGEN it, and then set the Global Memory by constructing a CP/M system of 63k bytes, using the appropriate version of MOVCPM from your CP/M 2.2 distribution disks, or the CPMSIZE utility from C.D.R. if you are using their floppy controller. Now, you transfer the the INSRAM.COM and RAMTST.COM files to your newly SYSGENed disk, plus PIP, STAT, DIR and other frequently-used files.

INSRAM.COM allows you to establish several parameters on your new system disk:

1. RAMdisk and Logical Disk drive letter assignments
2. Number and size of RAMdisks you want
3. The ability to warm boot from RAMdisk, even if you do a SHIFT/RESET in the event a program locks

You answer the on-screen questions in a few seconds, and you're ready to go to work. You have the option to create an "auto-install" file. If you choose this, INSRAM.COM will create two files called SRAM.COM (which brings up the RAMdisk in a preset form) and ARAM.COM, which is the auto-install version. It automatically checks for the amount of RAM available, and if the RAM drives have valid files and sets a clean directory if they don't.

INSRAM.COM also allows you to set the drive designations for the RAM partitions; and, if you choose A: and B:, your logical floppy assignments move down accordingly.

If you wish, you can create an autoload routine that will automatically load the RAMdisk with your favorite application program when you boot up. The documentation provides you with a step-by-step guide using any Text Editor to construct a SUBMIT file for this purpose. If you use several application programs with any regularity, you might want to consider creating a SUBMIT file for each one on different disks, and then making an extra file copy for backup purposes. By including ARAM.COM in the first line of your SUBMIT file, followed by the names of the programs you want to load into an assigned RAMdisk partition, you actually create an "auto-boot" situation that eliminates a lot of typing and file PIPing.

Since RAMdisk allows you to create a single 1-Megabyte partition, you can really appreciate this amount of overhead if you have 96-tpi DS/DD drives. You can load an entire disk of database files, mailing lists, inventories, or whatever into RAM and have almost instantaneous manipulation of data. Certain programs require the presence of two drives — one for the program files, the other for the data files. Setting your RAMdisk for 2 logical partitions is most useful in these instances, especially if you are going to record specific data on different disks. This will allow you to transfer the files from the second partition to your disk, clear the workspace and then dump more data files in this area of RAM. When you combine RAMdisk with a Winchester drive you'll never be satisfied in using your floppy drives. The H/Z-89 RAMdisk/Winchester combination operating at 2-MHz sure beats the pants off my 4.77-MHz IBM clone and gaudy color screen.

The original version of the Super RAM 89 CP/M software set the I/O port for the RAMdisk as 38H. This conflicted with the Magnolia Microsystems double-density controller which also used this port, as did C.D.R.'s own 5"/8" floppy controller. The software was quickly updated in version 1.5 to change this to 39H. The disk currently distributed includes both versions to allow some flexibility in the choice of floppy controllers. The INSRAM0.COM and RAMTST.COM files use 38H; INSRAM.COM and RAMTST.COM use 39H.

I was a little disappointed with the original documentation supplied in the CP/M Manual. It only provided 5 pages of instructions on using the CP/M programs. I felt a better effort could have been made on the part of L. D. Haag and C.D.R. Even the installation section is meager and it does not live up to either the quality of the hardware or the software. When version 1.5 was released, the disk included more comprehensive documentation on using the software in the form of INSRAM.DOC and README.DOC text files. Seven pages are devoted to schematics, physical layouts of the two boards, and a complete list of all the parts.

HDOS For Super RAM 89

By way of comparison, the 21-page HDOS manual prepared by Alan Heigl really sparkles, and attests to his experiences at the Heath Company. But I had a small problem.

Whenever I booted up my HDOS RAMdisk system volume and tried to mount a disk in another floppy drive or attempted to invoke the SET option help menu, the computer just locked up. With the aid of Alan Heigl, this difficulty was traced to the fact that I had the Magnolia Boot ROM installed. The HDOS SR-89 software requires that you have the Heath MTR-90 Monitor ROM installed as it makes several calls into the MTR-90 ROM for various operations.

I have encountered this problem with other recent software. Writers of HDOS software automatically assume that all systems use the MTR-90 ROM. It would be nice if somewhere it would state that specific Monitor ROMs or TLB ROMs are required to run certain programs or operating system enhancements that are tied to hardware. There are so many alternate ROMs in use on the '89 that the user often neglects to suspect that a specific software problem is related to the ROM set installed in his computer. Including me.

First of all, the HDOS software includes a utility called RAMTST14.ABS so that the user can run some comprehensive diagnostics tests on the RAM banks. This program is identical to the RAMTST

utility on the CP/M disk. This is a handy program if you ever develop a hardware problem with SR-89 that you suspect might be related to a faulty RAM chip. The program first asks you to identify the type of RAM chips installed in each bank (whether 64k or 256k) and then performs a read/write to each bank.

The HDOS RAMdisk software requires that you patch the HDOS SYS file on each disk you want to use with the RAM drive capability. This is similar to SYSGENing a 63k CP/M disk. The program HDOSCDR.ABS will make automatic patches to any disk by creating a 63.5k system bank. The unique aspect of this is that it will do so not only to new disks, but to existing disks loaded with your application software, even though HDOS.SYS is a write-protected file. You can perform this function to a disk in any drive, whether or not the disk is mounted. All you do is specify the name of the drive, and answer YES, and the program mounts the drive, patches the HDOS.SYS file and then dismounts it.

The actual HDOS RAMdisk driver is called RD.DVD. It can only be loaded into memory if RAMDRIVE.SYS is resident on the same disk. If you forgot to copy RAMDRIVE.SYS, RD: will actually remove itself from the HDOS list of available drivers, and you have to start over again.

When you boot up an HDOS system volume containing the RAMDRIVE.SYS file and RD.DVD on a 1-Megabyte SR-89, RAMDRIVE.SYS automatically identifies the amount of RAM you have and breaks it up into 15-63.5 banks. The RAMdisk partitions are established by performing the SET routine: SET RD: UNITS 1 (for a single partition) or SET RD: UNITS 2 (for two partitions). If the latter is chosen, the first partition will have 8 banks of RAM available and the second only 7 banks. This allows for a 63.5k RAM bank for system overhead.

You can also do a SET SWAP or SET NOSWAP. A SET NOSWAP sets up partitions RD0: and RD1: as electrical disk devices that you access like any other device under HDOS conventions. The SET SWAP option automatically locks RD0: and RD1: into memory as SY0: and SY1: and assigns your original boot drive to SX0: and your second drive as SX1:. Your alternate system drives remain as DK0:, etc. The swapping also transfers essential HDOS system files to the electrical disk so that it is able to perform the usual I/O communications with other devices and peripherals on your system. You are also able to create a RAMDRIVE.DAT file with the utilities provided on the distribution disk that will allow you to automatically load special programs and files to the RAMdisk at the same time that the system files are transferred. It is the functional equivalent of the CP/M ARAM.COM and SUBMIT.COM programs.

The HDOS RAMdisk cannot be SYSGENed, nor can it be used to SYSGEN disks, as the conventions of the software look for the presence of the original SY0: drive that you booted from.

While the HDOS version has, until now, appeared to be similar in functions to the CP/M version of RAMdisk, it doesn't incorporate the feature and the ability to keep RAMdisk files in memory when you do a SHIFT/RESET, followed by the usual Boot routine. Therefore, you must exercise care and make certain that you save any files you have been working on to disk before exiting.

Since I use SPOOLDISK 89 from FBE Research Company as my main workhorse (it is also a silent disk with 128k RAM), I generally PIP my files there and use it as a storage depot if I'm not quite finished with my work, because it is impervious to a system SHIFT/RESET and provides me with a safety margin. If it were

possible, I would like to see some kind of inexpensive back-up battery device that would keep the SR-89 RAM alive and refreshed even during a system shutdown or power failure.

Several other programs are worth mentioning. CDRDATE.ABS is a stand-alone program that will supply the correct date to HDOS during BOOT providing the C.D.R. Clock option is installed and set. A lithium battery maintains the clock when the computer's power is off. The CP/M version will insert the date and time at the beginning of each file you save to disk and provide you with a reference period as to when the file was created or worked with last. But unlike the HDOS version, it can't insert the file's date in the directory listing.

The other programs are ASMCDCR.ABS, which is the standard Heath HDOS assembler with 5 changes; and XREFCDRC.ABS, the standard Heath HDOS cross-reference program with 2 changes. They were provided pursuant to a license courtesy of Heath Company and Zenith Data Systems. The HDOS software manual requires a careful reading as it is chock full of information and options that help to make Super RAM 89 the interesting and useful enhancement it is.

The Winchester Software

Incorporating a hard disk drive on Super RAM 89 will be an evolutionary process covering at least a year or more.

I say this because the initial SASISOFT program, released in September 1985 was specifically geared to the Adaptec ACB-4000 controller. This is a SASI hard disk controller usable on a SCSI bus that doesn't have all the features of the Adaptec ACB-5000, which I suspect C.D.R. will ultimately select for an expanded implementation of the SCSI bus in a networking arrangement with the SCSI hard disk interface they expected to release for the Zenith '100 computer series as this article was being written.

Since there is a sizable installed base of hard disk systems on '89s using the Xebec S1410/a hard disk controller, a special version of SASISOFT is also in preparation.

As you will have noted from previous articles in this series, setting up a Winchester hard disk is not overly difficult, but you must perform a variety of steps in a certain order and must read the documentation very carefully.

The User Manual for the Adaptec ACB-4000 version of SASISOFT has been thoughtfully prepared, and includes a veritable treasure of hard disk drive information that I will discuss in a few moments. It is only available for use with CP/M.

The SASISOFT distribution disk contains the following programs:

SASIPREP.COM — This is the main program file, which is used to initialize a hard disk drive. It also tests the drive and creates SASI.COM, which becomes the hard disk boot set to run with the current system configuration. It uses SASIPREP.TBL and SASIPREP.OVL to perform some of its tasks.

SASIPREP.OVL — This file is used by SASIPREP.COM as an overlay which contains the subroutines necessary to communicate with the SR-89 SCSI interface.

SASIPREP.TBL — This file is used by SASIPREP.COM as a table of hard disk types to choose from. It contains all of the pertinent information used by SASIPREP.COM to initialize a large number of hard disk drives. The user can add to this list through the SASIPREP.COM program.

SASICCP.SPR — This is a ZCPR version of the CP/M 2.2 CCP in a relocatable form. It is used by SASIPREP.COM for part of the hard disk section in the creation of SASI.COM.

SASIBDOS.SPR — This is a modified BDOS for CP/M 2.2 in a relocatable form. It is used by SASIPREP.COM for part of the hard disk section in the creation of SASI.COM.

SASIBIOS.SPR — This is the hard disk addition to the system CP/M 2.2 BIOS. SASIPREP.COM uses it to create the SASI.COM file.

SUBMIT.COM — This is the standard batch file processor modified to be able to run from the hard disk. The conventional SUBMIT program will not work if installed on a hard disk.

D.COM — This is a variation of the many directory display programs. Most directory programs do not act correctly with large disk or large directories and will often show strange results.

The Manual does a creditable job in guiding the first-time user through the steps of formatting, partitioning and setting up an operational BIOS on a hard disk. The software appears to limit a CP/M partition size of about 8-Megabytes. I haven't been able to confirm this with C.D.R. and it isn't mentioned anywhere in the documentation. It also doesn't make use of the DMA option included on the hardware, and consequently, the transfer of data between RAM and the hard disk is considerably slower than the 1.5 Mbyte/sec allowed under the SCSI standard.

In terms of educating the user on how a hard disk functions, how you can formulate sector or cluster sizes to maximize the storage on your partitions, interfacing with controllers, and the inclusion of additional utility programs that would increase the versatility of the software package, the SASISOFT programs and documentation are not in the same league as the QUIKSTOR hard disk software from Quikdata Computer Services.

SASISOFT is a basic set of programs that get the job done, and in this regard it does it very well. I found the setting quite simple and free of anxiety inducing error messages. Quite possibly, when C.D.R. issues their full SCSI implementation, they will treat us to a more explicit manual — one that will demonstrate how multiple computers can share a common group of peripherals.

The "treasure" I mentioned earlier is in the form of a list that provides complete data on 127 hard disk drives from 23 manufacturers: number of cylinders, number of heads, write reduction information, formatted capacities, etc. This allows the user a great deal of flexibility in using any drive, rather than being limited to Shugart, Seagate, Tandon or Microscience drives. The size of the drives on the list range from 5 to 50-Megabytes, and includes not only the manufacturers listed above, but Atari, Computer Memories, Disctron, Evotek, Fujitsu, International Memories, Mitsubishi, Memorex, Micropolis, Nippon Peripherals, Olivetti, Priam, Rodime, Quantum, Densei, Vertex and Maxtor.

In a conversation I had with Herm Brooks relating to this list, I learned that C.D.R. went out and purchased a sizable number of these drives just to be able to include the pertinent data in their chart. I can appreciate this because personal experience in writing to drive manufacturers over a period of many months has netted very few responses for technical information. It's almost as if they couldn't be bothered. Hard disk drive controller manufacturers, on the other hand, have been uniformly helpful in providing me with technical manuals and other pertinent informa-

tion quite promptly. Having this information also allows the user to take advantage of some of the hard disk closeout bargains in the marketplace that are usually sold without a User Manual.

Summing Up

Viewed from an overall perspective, Super RAM 89 is one of the more significant enhancements for the H/Z-89, and ranks with the H-1000 replacement CPU board from Technical Micro Systems, plus the many boards from Magnolia Microsystems, Spool-disk 89 from FBE Research, and the Interactive Graphics Controller from SigmaSoft & Systems as add-on hardware that will pay for itself in terms of productivity many times over. And it helps to demonstrate that an 8-bit computer can be every bit as powerful as a 16-bit system. The appearance of this product also demonstrates not only the versatility of the H/Z-89, but the commitment of a handful of talented firms to continue support for this interesting machine.

* * *

Super RAM 89 Board 1 (w/o RAM) is \$190. Board 2 is \$90. Real Time Clock option is \$45. SCSI/DMA option is \$95. HDOS software is \$35. CP/M software is free with Board 1. Hard disk software — either for the Adaptec or Xebec controllers — is \$75. For current pricing on 256k RAM, hard disk drives, controllers, please contact:

CONTROLLED DATA RECORDING SYSTEMS, INC.
7210 Clairemont Mesa Blvd.
San Diego, CA 92111
(619) 560-1272

*

Welcome to *It's Great!*

DQODLER


Graphics Package

... for the Zenith Z-100 and Z-150/160

- Full feature graphics design package
- Save designs on disk for later use
- Playback mode for error correction
- Extended text capability includes...
 - User designed character fonts
 - Italic or backslant styles
 - All text may be scaled

See DQODLER at your local
Heathkit Electronic Center

...or Send \$ 79.95 directly to...



PAUL F. HERMAN

Software Graphics Tools
3620 Amazon Drive
New Port Richey, FL 33553
(813) 376-5457

Specification Sheet available on Request

"MAKE YOUR Z-150 CP/M COMPATIBLE"

*Intersecting Concepts Announces
3 Solutions To Solve Your
Computer Incompatibility!*

"But will it work on my computer?" YES!
Finally, there are three *easy* ways to exchange information, transfer files, and run CP/M software on MS-DOS machines.



1. MEDIA MASTER™ is our direct disk-to-disk format conversion program. Already an accepted industry standard, this \$39.95* program uses simple screen prompts that lets you read, write and format up to 155 different 5 1/4" diskettes from CP/M, MS-DOS and PC-DOS operating systems. So if you work on an IBM P/Compatible at the office, but use a CP/M computer at home, now you can easily transfer files that would otherwise be "foreign" to your computer's operating system. **NEW! Version 4.0 for the IBM PC and compatibles is now 300% faster!**



2. MEDIA MASTER PLUS™ features our new *Media Master 4.0*, plus a new version of *ZP/EM*, our powerful CP/M-80 emulator program. The results of this two-program set are amazing! Now you can run 8-bit Z80 CP/M programs from your Osborne, Kaypro or Zenith computers on your IBM PC or compatible. For only \$59.95, you can save your CP/M data *and* programs, and continue to get your money's worth out of your CP/M investment!

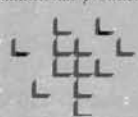


3. ACCELERATE 8/16™ dramatically improves the performance of *Media Master Plus* by tailoring the CP/M emulation around a user-installable NEC V20 microchip. Once installed, the NEC V20 chip will run your CP/M software 350% faster than *ZP/EM*! (MS-DOS programs run 15% faster). You can quickly command *Accelerate 8/16* to use 8080 software emulation, Z80 software emulation, or the V20's hardware 8080 mode to run your CP/M programs. This new version of *Accelerate 8/16* includes all the software packaged in *Media Master Plus*, V20 CP/M Emulation Software, and the NEC chip for only \$99.95!

TO ORDER

To order *Media Master*, *Media Master Plus*,
or *Accelerate 8/16*, call **800-628-2828, ext. 629.**

For additional product and upgrade information contact:



**INTERSECTING
CONCEPTS**
4573 Heatherglen Ct.,
Moorpark, CA 93021
or call 805-529-5073.



Dealer inquiries invited.

* \$99.95 for Dec Rainbow

Continued from Page 44

pens to be set exactly for my Zenith as well, given the configuration I listed above! I didn't have to make any changes whatsoever save for a margin change on my WordStar program. I have since written machine language routines called "IMAGE" and "COLOR" which, if you are interested, I will gladly send you copies of for inclusion in REMark. IMAGE is used alone or with a parameter to select from any of the eight type sizes. It has error checking and can be run from within WordStar or Condor DBMS to set the printer. COLOR is used to select any of seven printing colors that the ImageWriter II can produce given the 4 color ribbon. All programs are written with 8080 code and use CP/M so they should be compatible with just about every machine.

Anyway, thanks again for the supper. I remain

Yours sincerely,

Mr. Craig A. Pearce
2529 S. Home Avenue
Berwyn, IL 60402

HA-108 Upgrade

Dear Mr. Swayne:

I wish to thank you for your article "Installing The HA-108 Upgrade Kit On Older H/Z-100 CPU Boards". Using your article as a guide I collected the parts myself and installed them. I didn't buy the HA-108 because I had planned to use the 8 MHz NEC V-20 chip. But due to legal problems with Intel, the 8 MHz version has not been available. I miscounted by one pin on one jumper, but otherwise, the installation was very easy.

You can imagine my delight when I pressed S at the hand symbol and saw 768 k. I ran my compiled ZBASIC 10 iteration version of the Sieve of Eratosthenes. The result was 8.4 seconds. If the 8 MHz V-20 becomes available, I will make that change and see how much it helps. Now, if we just had a way to patch our software to use the 8087 I would like to make that conversion, too.

I would like to have permission to make a copy of the following pages of REMark magazine to include in my computer's manual.

- July 1985: Pages 22 - 28
- September 1985: Page 48
- November 1985: Page 40
- April 1986: Pages 17 - 22

I am looking forward to obtaining MSDOS 3 and ZPC2.

Thank you,

Howard D. Koozer
Koozer's Poultry Farm
2061 Highway 20
Sedro Woolley, WA 98284

ED: Permission Granted!

Problems With MASM V4.0 And Zenith Z-150 BIOS Routines

Dear HUG:

About a month ago, I received the MS-DOS V3 Programmer's Utility Package and tried to assemble the Z-150 version of the ANSI device driver using my MASM V4.0 that I received from Microsoft. Two problems appeared that aren't there using the MASM V3.0 that is supplied with the Utility Package.

The first problem has to do with the way the Assembler handles "NOT 0"; in the Z150ANSI.ASM code this is the FALSE and TRUE equates at the beginning of the code. Using MASM V3, the "NOT 0" is set to a-0001; in MASM V4, it is set to FFFFH. Then, when one tries to use the above equate in a DB statement, MASM V4 generates a severe error message stating that the "Value is out of Range". I reported this to Microsoft and they already are aware of this problem.

The second problem deals with multiple end-of-file characters (1AH) at the end of the Z-150 BIOS source files contained in the Utility Package. Normally, there are only one of these characters at the end of an ASCII text file, but for some reason Zenith chose to have multiple 1AH characters at the end of the file. This is not true for the Z-100 BIOS files. A different editor must have been used.

Well anyway, the Z-150 ANSI device driver code assembles with no errors using MASM V3, but I get 4 warning messages with MASM V4 stating that there are extra characters on certain lines and those are the lines with the multiple 1AH characters. Even with these 4 warning messages, the program appears to assemble correctly and produce a good assembly listing. However, if you take a close look at the file itself (dump the file contents in hex), you will see multiple 1AH characters following each of the included ".DEF" files. Then, when you try to print out this ".LST" file using PRINT.COM, the printing will stop right after reaching the first 1AH character that is in the first included ".DEF" file. COPY skips a big chunk of text because of these multiple 1AH characters.

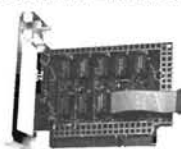



I have reported this to Microsoft in detail, but have no response to print at this time. My only question for Zenith Data Systems is why they put so many end-of-file characters at the end of the text. I did send a copy of my letter that I sent to Microsoft to Zenith Software Consultation.

Appreciate putting this into an upcoming issue of REMark. Thanks.

Sincerely,

Richard (Rich) L. Mueller, Ph.D.
11890-65th Avenue North
Maple Grove, MN 55369



HEATH/ZENITH 88, 89, 90 PERIPHERALS	
16K RAM EXPANSION CARD  Only \$65.00 Shipping & Handling \$5.00	REAL TIME CLOCK  Price \$130.00 with Batteries Shipping & Handling \$5.00 \$114.00 w/o Batteries
2 PORT SERIAL/3 PORT PARALLEL I/O CARD  Price \$199.00 2nd Oper. System Driver \$25.00 Ship. & Hdlg \$10	 <small>PRICES ARE LESS SHIPPING & TAX IF RES. OF CALIFORNIA.</small> <small>MAIL ORDER: 12011 ACLARE ST CERRITOS, CA 90701 (213) 924-6741</small> <small>TECHNICAL INFO / HELP: 8575 KNOTT AVENUE, SUITE 0 BUENA PARK, CA 90620 (714) 952-3930</small> <small>TERMS & SPECIFICATIONS SUBJECT TO CHANGE WITHOUT NOTICE VISA & MASTER CARD GLADLY ACCEPTED</small>

Part Number	Description of Product	Selling Price	Vol. Issue
DATA BASE MANAGEMENT SYSTEMS			
HDOS			
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
CP/M			
885-1219-[37]	CP/M Navigational Program	20.00	31
MSDOS			
885-6008-37	MSDOS NAVPROG	20.00	73
885-8034-37	DBZ-A Database For The Z100	25.00	69
AMATEUR RADIO			
HDOS			
885-8016	Morse Code Transceiver Ver 2.0	20.00	42

Part Number	Description of Product	Selling Price	Vol. Issue
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
MSDOS			
885-8038-37	MSDOS RFCAD Ver 3.50	30.00	73
COMMUNICATION			
HDOS			
885-1122-[37]	HDOS MicroNET Connection	16.00	37
885-8005	MAPLE (Modem Appl. Effector)	35.00	29
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
885-8012-[37]	CP/M MAPLE (Modem Program)	35.00	34
885-8023-37	CP/M-85 MAPLE	35.00	45

Part Number	Description of Product	Selling Price	Vol. Issue
MSDOS H/Z100 - H/Z150 PC			
885-3019-37	ZDOS 16 Bit MicroNET Connect	16.00	61
885-3027-37	MSDOS HUG PBBS	40.00	66
885-3028-37	MSDOS HUG PBBS Source Listing	60.00	66
885-3033-37	MSDOS HUG MCP	40.00	71
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	
885-4006	REMark Vol. VI Issues 60-71	25.00	
885-4500	HUG Software Catalog	9.75	
885-4501	HUG Software Catalog Update #1	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
885-3036-37	MSDOS TREE-ID	20.00	77
<p>NOTE: The [-37] means the product is available in hard sector or soft sector. Remember, when ordering the soft sector format, you must include the "-37" after the part number; e.g. 885-1223-37. *</p>			

Productivity Tools from SCE

Quality Products for Professional Software Development

The Seidl Make Utility (SMK) is a powerful automatic product generation utility. When changes are made to any of the modules in a product, **SMK** will execute the minimum set of commands that are *necessary and sufficient* to rebuild the product. **SMK** is loaded with features and performance you won't find in any other make utility. It allows you to define product dependencies quickly and easily using **SMK's** high level dependency definition language (DDL). The DDL supports parameterized macros, for-loops, constants, local variables and much more. **SMK** understands complicated dependencies involving nested include files and object code libraries. You have complete control over generated commands. Use any type of error recovery technique desired. Easy to learn and lightning fast, you'll wonder how you got along without **SMK!** **\$99.95**

The Seidl Version Manager (SVM) is a collection of tools that together form a state of the art version control system. **SVM** maintains a complete revision history of all source files in a project without duplication and has highly optimized text compression routines to further reduce disk storage requirements. **SVM's** high-level make facility integrates tightly with **SMK** allowing products to depend on other products. **SVM** lets you rebuild any version of any product or group of products with a single command. **SVM** includes a powerful audit trail report generator. A full screen, menu-driven shell provides a user friendly interface to the **SVM** tools. You can also use the **SVM** tools directly or incorporate them into custom applications. **\$299.95**

SAVE OVER \$20.00 when you order both, just **\$379**. Include \$3.50 shipping for **SMK**, \$5.00 for **SVM**. Multi-site licenses and educational discounts available. Dealer inquiries invited.

SEIDL COMPUTER ENGINEERING

3106 Hilltop Drive, Ann Arbor, MI 48103 (313) 662-8086

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

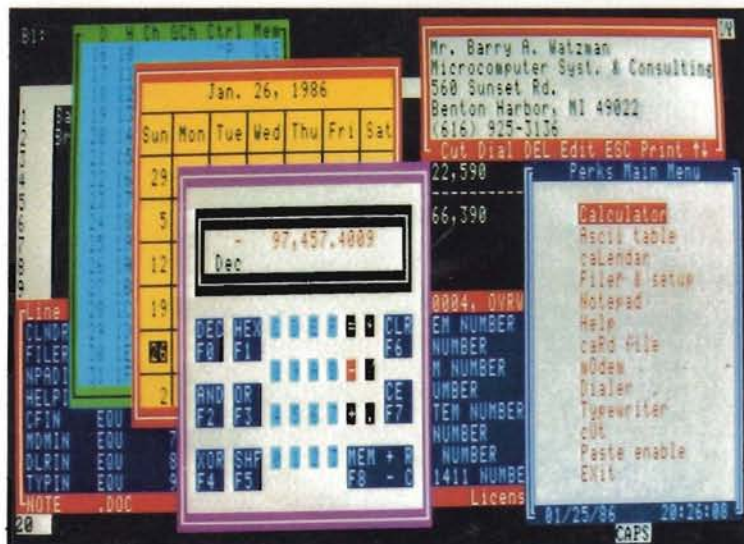
	NEW MEMBERSHIP RATES	RENEWAL RATES
U.S. DOMESTIC	\$20 <input type="checkbox"/>	\$17 <input type="checkbox"/>
FPO/APO & ALL OTHERS*	\$35 <input type="checkbox"/>	\$30 <input type="checkbox"/>
		U.S. FUNDS

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

Announcing
PERKS™

Version 2!!!

- New!** Cut/Paste Module!
- New!** Typewriter Module!
- New!** Variable Size Notepad Buffer (up to 64K)!
- New!** Modem Module!
- New!** Dialer Capability!
- New!** Card File Module!
- New!** Removable under *both* MS-DOS *and* Z-DOS!
- New!** Additional Features in Existing Modules!
- Old!** Still works with *both* MS-DOS *and* Z-DOS!
- Old!** Still small in size (79K)!
- Old!** Still only \$69.95!



"Shown above is an actual photo of Perks in operation, with Lotus 1-2-3 in the background. The main menu and windows for the notepad, calculator, ASCII table, calendar and card file modules are visible."

Now the leading Z-100 Desktop Utility is even better! Version 2 of Perks adds the most asked for additional features while retaining Perks' superior user friendliness, better documentation and compatability, ease of use, small size and ability to run under both Z-DOS and MS-DOS. If you have a Z-100, this is one program you can't afford to be without. And at it's low price of \$69.95*, you don't have to!

Perks is available at all Heath/Zenith Computers & Electronics Centers, many independent Zenith Data Systems dealers or directly* from:

BARRY A. WATZMAN Microcomputer Systems & Consulting

560 Sunset Road • Benton Harbor, Michigan 49022-7142 • (616) 925-3136

Perks is a trademark of Barry Watzman.

*Plus \$4.00 for S&H, Michigan Residents add 4% sales tax also.



Hilltop Road
Saint Joseph, Michigan 49085

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