REMark®

Volume 7, Issue 10 • October 1986
P/N 885-2081 Issue 81

Official magazine for users of HEATH ZENITH computer equipment.

Coverage Of The Fifth International HUG Conference Will Appear In November 1986

$2.50

Continued from Page 43

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
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
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

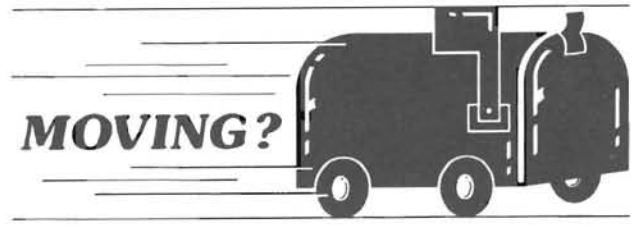
NOTE: The [-37] means the product is available in hard sector or soft sector. Remember, when ordering the soft sectored format, you must include the "-37" after the part number; e.g. 885-1223-37.

Continued from Page 42

- Multiplication
- Output to Printer (with titling)
- Row-Reduction
- Summation and Subtraction
- Transpose
- Disk Reading and Writing (ASCII or binary)

Comments: MATT was written with ease-of-use and speed in mind. The menu items are designed to be mnemonic. For instance, "M" performs multiplication, "S" performs summation, etc. In short order, the new user can be performing matrix operations without needing to look at the menu. "One touch" menu response, spreadsheet type matrix entry and editing, and input "garbage filters" make MATT very friendly. Support for the 8087 numeric coprocessor can be added by simply recompiling the source using Turbo-87 from Borland International, Inc. Z-100 owners having the MOUSEPACK by Paul F. Herman will find the spreadsheet editing compatible with their mouse system.

TABLE C Rating: (10)



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

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

CUT ALONG THIS LINE

HUG MEMBERSHIP RENEWAL FORM

HUG ID Number: _____

Check your ID card for your expiration date.

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

Name _____

Address _____

City-State _____

Zip _____

REMEMBER - ENCLOSE CHECK OR MONEY ORDER

CHECK THE APPROPRIATE BOX AND RETURN TO HUG

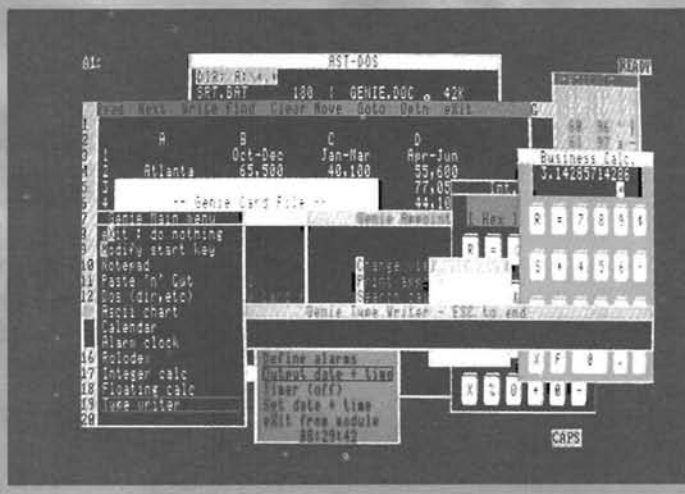
	NEW MEMBERSHIP RATES	RENEWAL RATES
U.S. DOMESTIC	$20 ☐	$17 ☐
FPO/APO & ALL OTHERS*	$35 ☐	$30 ☐ U.S. FUNDS

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

NOT PROTECTED

Genie™

Software Magic from Advanced Software Technologies



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.

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)

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 _____

Heath/Zenith Users' Group

Staff

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

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

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

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

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

REMark®

Buggin' HUG 7

Z-181 Portable Personal Computer
Rick Simpson 9

Ordering Of Unordered Data In A Spreadsheet Or MYCALC Is My Bookkeeper
Alkis J. Sophianopoulos, Ph.D. 11

Your Word Processor Speak BASIC?
Lawrence J. Durney 17

For HDOS Or CP/M Install A Switch To Swap Ports On Your H8-4 Interface Board
Jim Meyers 19

Heath/Zenith Related Products
Jim Buszkiewicz 22

EPSON FX-85 Update
Richard L. Mueller, Ph.D. 23

Getting Started With Hierarchial Directories Part 1
Erik L. Pang 27

Mainstream Computing
Joseph Katz 31

ZPLOTTER.BAS
Kenneth R. Van Wyk 37

Official magazine for users of HEATH ZENITH computer equipment.

Volume 7, Issue 10 • October 1986

ZPC Update #10
Pat Swayne 39

HUG New Products
..... 42

HUG Price List
..... 43

SEEK.COM
A CP/M Assembly Language Learning Adventure Part 4
M. D. Zapolski, Sr. 45

Two Parallel Interfaces For The H-89 (One For The H-8?)
Pierre D. Olivier, III 51

DEBUG
Jerry Furst 53

A Non-Blinking Cursor For Your PC
Pat Swayne 57

Z-37 Magnolia CP/M 4 MHz
Loren R. Porter, Jr. 60

BASIC Thoughts
John H. Mosow 63

On The Leading Edge
William M. Adney 69

Automating The Configure Feature
Charles F. Ballinger 77

Index of Advertisers

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

Advanced Software Technologies 3
American Cryptronics 62
Analytical Products 59,76
Aspen Systems 21
Bersearch Information Services 22
CDR Systems, Inc. 6
Cometsoft Technology 68
Computer-Video Service Center 21
FBE Research Company, Inc. 52
First Capitol Computer 26
Gemini Technologies 56
HUG PBBS 35
Heath Company 38
Paul F. Herman 22
Intersecting Concepts 50
KEA Systems 41
Kalltronics 15
Lightek 68
Micronics Technology 55
Quest Computing 68
RAM Technology 44,50
S&K Technology, Inc. 18
Scottie Systems 21
Secured Computer Systems 79
SigmaSoft & Systems 47
Software Wizardry 26
Veritechnology Electronics Corp. 2
Barry A. Watzman 55,84
Western Regional HUG Conference ... 16
X-10 Powerhouse 30

On The Cover: Featured this month is the just released Z-181. Read about this compact little computer on Page 9.

Controlled Data Recording Systems Inc.

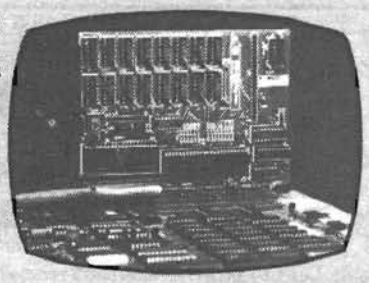
Quality Products and Support for the Heath/ Zenith Community

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!

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

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

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

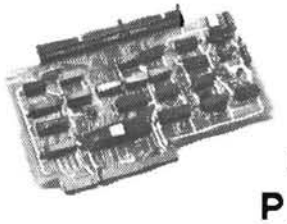
The Ram Drive Software (SRAM) allows one or two logical ram drives. The ram drive(s) can be located starting anywhere from logical A: to O: (standard drives get relocated). SRAM can be set to start at logical A: and warm boot with ram (no floppy disk accesses needed). Ram drive attatches to any of the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

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.

FOR THE H/ Z89-90 COMPUTERS

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



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

Now be able to run standard 8" Shugart compatible drives and 5.25" drives (including the H37 type) in double and single density, automatically with one controller. The FDC-880H operates with or without the Heath hard sectored controller.

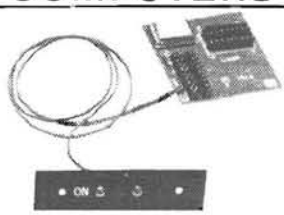
Z100 HARDISK CONTROLLER

The CDR317 is the ULTIMATE IN HARD DISK controllers for the H/Z110- H/Z120 computers

- Direct replacement for the Heath/Zenith Z217 hard disk controller package.
- Able to work with standard ST506/412 type Winchester Drives and the cartridge removable Syquest Drives.
- Works with all standard Heath/Zenith drivers, operating systems, and Z217 Boot Roms.
- Is an intelligent SCSI host adaptor for future network expansion.
- Less expensive than the Z217 controller.
- Available now through your local Heath/Zenith dealers.

FOR THE Z100 COMPUTERS

THE ORIGINAL Z100 SPEED MODULE. RUN YOUR Z100 PROGRAMS FASTER



The ZS100 runs the Z100 CPU 50% faster. (7 5 MHZ) in 8088 mode. The ZS100 installs easily with no soldering. The ZS100 is externally switchable between speed mode and normal. The ZS100 improves the time performance of applications packages with no software modifications needed The ZS100 is for all Heath/Zenith 100, 110 and 120 series computers.

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 **$495** LIST

Dealer inquiries invited.

For information about these products contact
Controlled Data Recording Systems Inc
7210 Clairemont Mesa Blvd
San Diego, Ca 92111 Phone (619) 560-1272 or contact a Heath/ Zenith dealer that carries the C D R Systems Inc line of products
New - C D R Systems Bulletin Board (619) 560-8929

BUGGIN' HUG

A Short Note

Dear HUG:

A note about the use of the WatchWORD word processor and its speller and Perks on the Z-100.

The documentation for the WatchWORD Speller states that it can't be used with Perks because both use the Shift-break to initiate them. However, Perks allows a change in that initiation sequence and it is possible to use both by setting Perks to start with, say, Cntl-break.

Sincerely,

Miriam Clifford
2535 Sevier Street
Durham, NC 27705

WordStar Directory Changer

Dear HUG:

I found Joseph Katz' article, "The WordStar Directory Changer" (REMark, April 1986) very interesting. There is a simpler way to perform the same task in DOS 3.1, however. The SUBST command allows users to assign a disk drive letter to a subdirectory of a hard disk. Using this trick, you can assign drive letters to all needed subdirectories in a BAT file and then shift between subdirectories easily.

A related matter makes this difficult for some users. Zenith has been promising an upgrade offer from DOS 2.11 to DOS 3.1 for more than a year now (see the February 1985 issue of REMark). Rumor was that the offer would allow purchase of 3.1 at $100 minus the 20% HUG discount, or a net cost of $80. IBM offers no discount trade-in policy from PC-DOS 2 to 3, but sells PC-DOS 3.1 at $85 list (versus Zenith's ridiculous $150). Zenith is long overdue in offering MS-DOS 3.1 to its customers at a reasonable price. (Incidentally, IBM's PC-DOS 3.1 appears to run perfectly on my Z-151 PC.)

Sincerely,

Richard G. Anderson
2606 Swansea Road
Columbus, OH 43221

An Easier Way To Install "Easy PC"

Dear HUG:

Reading Adney's review of the UCI Easy PC emulator board, prompted me to write to Buggin' HUG with a helpful suggestion. I recently installed the Easy PC board in my low profile Z-100. I sympathize with Adney — after a half hour spent trying to get all sets of pins lined up, I was getting desperate. I'd bent some pins, but fortunately had not broken any. I was getting eye strain and a backache. Finally, I decided to remove the motherboard from the case.

Since you have to remove almost the entire machine to install the Easy PC emulator, it is virtually no more trouble to remove the motherboard, as well. By doing this, I was able to hold both boards up to a strong light and carefully align the pins, then sandwich them together. It sill took several tries, but no broken pins.

So save yourself a headache and take the motherboard out for an easier installation.

Thanks for all the good information.

Louise Mezzatesta
924 N. Mapleton
Oak Park, IL 60302

Extended-Density Diskettes Under CP/M-80

Dear HUG:

It seems that one of the qualifications for becoming a Heath/Zenith software engineer must be an overwhelming addiction to "Adventure" games, judging by the code that they create. Every time you study the source code for the Heath/Zenith CP/M-80 BIOS or the ever-popular, but orphaned, operating system HDOS, you find more hidden treasures just waiting to be released from hiding by your incantation of the proper 'magic words'. Some of these hidden treasures, like SET HDOS STANDALONE, have become widely known and much talked-about, but others seem to have remained obscure and relatively unknown. Judging by some recent correspondence we've had with some other HUG members, there's one very useful undocumented feature of the Heath/Zenith BIOS that we've been using for years, but that many of you are unaware of. The purpose of this article is, in the analogy of Crowther and Woods from "Adventure," to "wave the rod while standing at the end of the edge of the fissure and watch the crystal bridge appear, so you can get the treasures on the other side". In simpler terms, for those poor souls who have no idea what "Adventure" is all about, we intend to shed some light on one of these hidden treasures so everyone can have access to it.

Extended Density is a method of formatting soft-sectored floppy diskettes that gives you about 25% more usable space on the diskette for your information. Owners of the H47 and H67 eight-inch disk drive systems should already know about Extended Density, since it is clearly displayed as an option when running the FORMAT program, and it is briefly mentioned in the Heath/Zenith documentation. But owners of the H37 five-inch disk drive system often don't realize that they have an Extended Density option available to them, also. The FORMAT program does not mention it, nor does the Heath/Zenith documentation. But it is real, and it is (usually) there.

"Okay," you ask, "how do I know for sure if I have Extended Density in my system or not?" Well, it's actually pretty simple. If you are using an ordinary Heath/Zenith-supplied BIOS, any of the following conditions will automatically guarantee that Extended Density is enabled in your BIOS for the H37 drives:

- Your BIOS includes H47 support
- Your BIOS includes H67 support
- You have successfully read a CP/M-85 diskette that was FORMATted on an H/Z-100 computer in double-side, double-density mode.

Well, what if none of these apply? Say you are using H17 and H37 drives, or only H37 drives, or a modified BIOS. Perhaps you don't

even know someone with a Z-100. What then? Well, there is a test, and the test is still quite simple. Get out a blank diskette (or one that has no useful data on it that can be erased). Boot up with a disk that has both FORMAT.COM and STAT.COM on it, put the blank or expendable diskette in drive B:, and proceed as outlined below. Note that in the example given, we used a double-sided, 96 TPI drive, but single-sided and/or 48 TPI drives are also okay. Your input is represented by the bold input in the examples, and comments are represented by the italicized text. Here goes:

```
A>FORMAT
Format version 2.03
This program is used to initialize a disk
All information currently on the disk will be destroyed
Is that what you want (Y/N): Y

Which drive do you wish to use for this operation: B
any solid drive letter is okay

Which density? (S=Single, D=Double): E
note that ""E'' wasn't mentioned

Number of sides? (1=Single, 2=Double): 2
""1'' for one^sided drives is okay

96 TPI drive -- 80 tracks will be formatted
no such message for 48 TPI drives

Put the disk you wish to be formatted in drive B.
Press RETURN to begin.
```

If you made it through the entire sequence above, and FORMAT did not complain when you answered the question about "density" with an "E", instead of "S" or "D", then you have probably passed the Extended Density test, but one more little experiment will tell you for sure. Try the following command:

```
A>STAT B:DSK:
```

If STAT complains that it cannot read the disk in drive B:, then you have obviously failed the last test. However, if STAT was able to read the diskette, it will report back the available space on the disk and the organization of the diskette formatting to you. Compare the data for your Extended Density diskette in B: to the data for your standard double-density diskette in A:, and you'll see how much extra space you just earned for "free".

What if the "ACID TESTS" above fail? Well, all is not lost, unless your Heath/Zenith BIOS has been modified to an unusual extreme. You can "turn on" Extended Density in your BIOS if it is now "turned off", and it involves no more than a minor change with a text editor, followed by the infamous MAKEBIOS procedure.

To "turn on" Extended Density in a BIOS that does not have it enabled already, you need to edit the file BIOS.ASM from your Heath/Zenith distribution diskette set. If there's any chance that the file is write protected, issue the command:

```
A>STAT BIOS.ASM $R/W
```

which will take off the file-level write-protection, if any. Then, call the file into your text editor. (This is no small task in itself, since the BIOS.ASM file is so huge. But there are so many different editors with such varied capabilities in this area, that it's impossible for us to tell you how to accomplish this feat. Read your editor's documentation. You may have to read in part of the file, make the change, write out the modified part, read in some more, etc.). At around line 55 or so, look for a line of assembly code that says:

```
H37ED EQU FALSE AND H37T
```

and change it so it reads:

```
H37ED EQU TRUE AND H37T
```

Save the modified file back to diskette, and then run through the entire MAKEBIOS procedure, including SYSGEN, performing a hard reset of the computer, and rebooting the newly SYSGENed diskette with the new BIOS on it. If you are unfamiliar with the MAKEBIOS procedure, refer to your Heath/Zenith documentation, and also to Pat Swayne's article in the March 1982 issue of REMark.

So what do you get from all these efforts? Well, on a double-sided, 96 TPI diskette, your usable storage capacity will rise from 624K to 782K. On a 48 TPI diskette, your storage will rise from 304K to 382K. The "magic" comes from using larger sectors (1024 bytes instead of 256 bytes per sector), and putting fewer of them on a track (5 instead of 16). Because each sector needs extra space for "housekeeping" information, it is more efficient to make them bigger and have fewer of them. With this arrangement, we get 5 kilobytes pertrack instead of 4 kilobytes, a 25% increase. The Z-100 and Z-150 series of computers take a middle-of-the-road approach — they use 8 or 9 sectors of 512 bytes per track, for 4 to 4.5 kilobytes per track, which are the IBM standards.

And speaking of the Z-100, can their owners benefit from any of this? Well, the answer is 'yes and no'. If you Z-100 owners have an H8 or H89 around the house, you can use the 8-bit machine to FORMAT diskettes in Extended Density, and then use them in the Z-100. The Z-100 can read and write to such diskettes, but it cannot FORMAT them or boot them. Note this latter restriction, at least with the Z-100s we've tried it on, with older ROMs. The ROM bootstrap code couldn't cope with a non-standard track layout.

Are there any disadvantages with this arrangement? Well, the software "wizards" at Heath/Zenith said at HUGCON a couple of years ago that they didn't document Extended Density because they didn't feel that it was as reliable a recording technique as standard double-density. However, we have Extended Density diskettes in our collection going back to 1981, and not only have they been reliable in a single H-89, but they've been swapped back and forth among at least four different H-89s with a bewildering array of 96 TPI drives made by Tandon, Shugart, Siemans, and Teac. Some of the H-89s are in home environments, and others are in work environments without air conditioning. We are satisfied that the Extended Density recording method is very reliable.

So, the next time you are ready to FORMAT a blank floppy disk, remember your 'invisible option', namely Extended Density, and use it to reduce your data storage costs by about one-fourth.

Dan Jerome
801 East 132nd Street
Burnsville, MN 55337

John Toscano
6117 130th Street, West
Apply Valley, MN 55124

Letter Of April '86 On GW-BASIC

Dear HUG:

The April 1986 issue of REMark includes a letter mentioning that ZBASIC and GW-BASIC support only two of the H/Z-100 screen escape codes listed in the Z-100 User's Manual. My experience is that many such codes (but not all) DO work with GW-BASIC.

I use GW-BASIC on my H-101 (and have not tried ZBASIC). I also have not tried all the screen escape codes in Appendix B of the user's manual. However, GW-BASIC on my H-101 will respond to all of the erasing and editing codes on page B.14, except that ESC E will only clear the display (it will not also home the cursor). However, the cursor must be ON in order that these codes work. I find this limitation quite annoying when working with fixed position screen I/O displays because that often requires considerable contortions to avoid cursor skitter on the screen.

Continued on Page 80

Z-181 Portable Personal Computer

Rick Simpson
Technical Consultant
Heath Company

If you're in the market for a portable personal computer that weighs less than 12 pounds, features large capacity 3.5" drives, includes a large screen that's readable in daylight or in a dimly lit room, runs IBM PC/XT compatible software, plus much more, then here it is!

The Z-181 built-in backlit display is one of the most legible of any portable PC. Compared to standard liquid crystal displays, brightness and contrast can be adjusted in the Zenith Z-181 to fit lighting. And the screen can be tilted to a comfortable reading angle. The result is a display that is exceptionally easy to read, in any light. Wait, here's the best part! The display used in the Z-181 is one-third larger than other desktop PC displays. Now graphics can be shown on a portable computer without the distortion found with smaller screens.

Now that I've got your attention, here's some information about the Z-181's standard features:

CPU: The 16-bit 80C88 microprocessor operating at 4.77 megahertz provides PC compatibility, while using less power.

CoProcessor: An 8087 coprocessor socket for high speed math processing. AC operation is recommended when the 8087 is used.

Memory: The system includes a full 640k NMOS RAM, the maximum memory to allow for most software needs today and for easy accommodation of a RAM disk if desired. 32k of ROM includes the standard PC compatible ZDS ROM BIOS with monitor ROM features.

Disk Drives: When you open up the Z-181 computer you will find two 3.5 inch floppy drives neatly tucked away inside. The pop-up design provides maximum environmental protection. Each 3.5 inch drive is formatted for 720k per disk, or 1.44 Megabytes in all. Since the disk drives are in front of you as you type, rather than on the side, you can see the lights come on when the drives are in use.

Display: The new, large size "supertwist" LCD with backlighting, offers both contrast AND brightness controls for excellent screen visibility in ANY lighting even at wide viewing angles. The Z-181's 25 line × 80 character backlit display achieves a 12:1 contrast ratio. This comes much closer to the 18:1 ratio of monochrome TV monitors than traditional LCD displays whose contrast ratio is in the 3:1 ballpark unilluminated, and 7:1 illuminated with a backlight.

The 10.5 inch diagonal screen is backlit by a blue electroluminescent panel, and there's one other improvement over other LCD screens: true aspect ratio. The screen's width is 1.33 times its height, which is the exact same proportion as CRT video monitors. When running graphics software on the Z-181, they make a circle a circle, instead of the squashed elipse which would appear on most LCD screens.

The backlight shuts off automatically after two minutes if no key on the keyboard is touched. This saves both battery power and screen life. The display also features adjustable brightness and contrast.

The Z-181's non-glare display can be tilted into any position over a 150-degree range. The screen's resolution is 640 by 200 pixels which is capable of four different brightness levels, thus improving graphics capability and allowing highlighting in word processing applications. When running software designed for color displays, the LCD screen will assign different grey levels for the three primary TV colors.

Keyboard: The Z-181 keyboard is a professional, full size keyboard. Each key is clearly labeled and sculptured to minimize input errors. There are ten function keys along the top, and four cursor control keys in the top right corner. Since there's no room for a separate numeric keypad, the seven, eight, and nine keys form the top row of a numeric keypad which can be activated using the Num Lock key. The U, I, O, J, K, L, AND M keys then become number keys for number crunching.

The Delete key is located on the lower right hand corner and the Caps Lock key is located just to the left of the A, where the Ctrl key is usually placed. The Control key is even further left. Everything else is about normal. The keyboard feel is crisp and free of bounce.

Interface Capabilities: The Z-181 provides standard I/O ports protected by a latch door on the back of the computer. An RS-232 serial port is provided for connections to printers, external modems and other I/O devices. One Centronics parallel port, a new 9 pin D connector video interface that supports both RGB color and composite monochrome output, and a 5.25 inch external floppy drive interface.

Power: An external AC adapter/recharger is included along with a Nicad battery pack offering 2 to 4 hours of uninterrupted power before it requires recharging. Total time between recharges depends on disk access and backlight brightness usage. The battery is an overnite recharge, is easily removed for replacement of a fresh battery pack, and the system may be operated while charging the battery.

Software: The Z-181 laptop computer will ship with MS-DOS 3.2 operating system. A demonstration disk will also be included to introduce the user to their new computer. A diagnostic program will be made available for those individuals that feel technical.

Before I forget, the system also includes a real time clock easily accessed by MS-DOS and other software applications.

THAT'S AN IMPRESSIVE LIST OF FEATURES FOR SUCH A LIGHT-WEIGHT COMPACT SYSTEM!

Here's some planned options for the Z-181 computer.

1. A 5.25 inch external floppy disk drive (ZA-181-8) will offer full IBM disk compatibility.
2. An external battery pack recharger (ZA-181-12) will recharge an optional extra battery (ZA-181-14).
3. An internal 300/1200 baud Hayes-compatible modem (ZA-181-5) will allow it to telecommunicate.
4. A 230-volt AC adaptor (ZA-181-4) will make it easier to travel the globe.
5. For travelling, there is a cigarette lighter adapter/recharger (ZA-181-7).
6. Because the 9-pin video D-connecter includes both the RGB and composite signal, a special monochrome video cable (ZA-181-9) will be available.

Other planned options for the very near future are a carrying case with handle and shoulder strap, a direct connect PC file transfer kit, and an optional 8087 coprocessor chip.

Z-181 Product Specifications

Processor: CMOS 80C88 16-bit processor operating at 4.77 MHz. Socket for 8087 numeric coprocessor.

Operating System: MS-DOS

Memory: 640K RAM standard: each bank is 8 bits wide with no parity bit: 32k ROM: 16k Video RAM.

Drives: Shock mounted dual 3.5 inch double-sided/double-density/double-track drives. PC-DOS compatible, formatted capacity 720k each.

Display: Electroluminescent backlit LCD, 25 line × 80 characters. 10.5 inch diagonal viewing area with true aspect ratio. Adjustable contrast and brightness. 150 degree adjustable tilt. Backlighting auto-time out feature.

Display Grpahics: LCD 80 characters × 25 lines. 640 × 200 pixels. Displays the ASCII 96 character subset; full descenders on lower case letters; reverse video.

Video: RGB video interface standard (with intensity) plus composite monochrome interface. A standard 9 pin D connector is provided for video ouput. "Color" displays are represented by a corresponding gray level.

Keyboard: Full Keyboard, QWERTY typewriter style with 4 cursor control keys; full ASCII character set; auto repeat; ten user programmable function keys; calculator keypad overlaid onto keyboard and numeric lock key.

Date/Time Support: Battery-backed real time clock and calendar.

I/O Ports

Serial Port:
- Male DB25 RS-232C connector; asynchronous RS-232C compatible.
- User software selectable
- Number of start bits — 1
- Number of data bits — 5,6,7,8
- Number of stop bits 1 or 2
- Baud rate — 110,150,300,600,1200,2400,4800,9600
- Signals monitored-received data, clear to send, data set ready, ring detect and carrier detect.
- Signals controlled — transmit data, request to send, data terminal ready.
- Supports both full and half duplex operation; odd, even or null parity.

Parallel Port: Female DB25 connector; Centronics compatible.

External Drive: Interface for 5.25 external drive.

Physical: 11.8lbs with battery pack standard (removable). 13.4"W × 11.6"D × 3.1"H (34.0cm × 29.5cm × 8cm)

Power

AC Power: Either 115v (60hz.) or 220v (50hz.) (Adapter/charger unit included)

Battery Power: 12 volt Nicad battery pack standard (removable). Low battery indicator. Computer is operational while recharging.

Operational

Temperature Range: 42-95 degrees F; (5-35 degrees C)

Conclusion

Well, it looks like Zenith has done it again. First the ZP-150 Laptop computer, then the Z-170 portable, now the NEW Z-181 portable personal computer. It would appear that Zenith intends to be a leader in the portable marketplace.

So if you're in the market for a light weight portable computer, with a super sharp display and much, much more, then the Zenith Z-181 computer is the right choice. *

Ordering Of Unordered Data In A Spreadsheet

Or MYCALC Is My Bookkeeper

Alkis J. Sophianopoulos, Ph.D.
2994 McCully Drive, N.E.
Atlanta, GA 30345

If your present software can organize your personal finances to your satisfaction, then skip this. However, I like this one for two reasons. You can tailor it to your specific needs and you can change it easily. The other reason is that it illustrates how one can enter data randomly in a spreadsheet and end up with a fully ordered form. Thus, one has built in the advantages of "what if" questions without the complicated formatting of some other programs. I have used it only with MYCALC(TM) from Software Toolworks, with an H-89A with 64K, at 2 Mhz, using HDOS. You might be able to use another spreadsheet, if it has the following features: 1) a sorter; 2) automatic update and reference to formulas throughout the sheet; 3) "lookup" and "if" formulas.

Brief Survey Of The Formats

The spreadsheet is made up of three parts which are functionally distinct, FIXFORM, TOTFORM and CHKFORM. In a PC computer with plenty of memory all three can be combined into one. In my discussion, I shall treat them as being combined to simplify things. For those of us still using 8-bit computers, the three parts must be kept as separate files and used as I shall describe later on. Figure 1 shows a small spreadsheet which would fit in any computer and which has all three parts combined. It is shown as it would normally look when printed out. Since you may tailor your version to your own needs, I shall discuss only the contents of the template which must be included in any template you prepare. The three parts of the sheet, either combined as in Figure 1 or separate, are saved with the minimum amount of data necessary. Each month you start with these parts, you enter your data and then save them under a different name. For example, CHKFORM could be saved as CHK286.MC for February 1986.

In Figure 1 in the CHKFORM part, column b lists the code numbers assigned to the various categories. Column a is suppressed (set to WIDTH 0) but is an exact duplicate of column b. Columns d through g contain the date, check number, payee or payer and amount. Column c, which is normally suppressed, duplicates via a formula the amounts in column g. You may also notice that there are several lines where columns c through f are blank and column g has 0. These listed features whose purpose is not obvious are part of what makes the sheet work, as we shall see.

You determine the various categories to describe your transactions. There may be as many categories, subcategories and subsubcategories as desired, and each one is assigned a unique code number. The smallest number must have at least as many digits as the number of classes of categories and subcategories defined. For example, if there are three levels of categories, the smallest number would be 100. The code numbers are defined as NUMBERs, not LABELs. All this is necessary because of the way the sorter works. Otherwise, you may have some unpleasant sorting results.

TOTFORM summarizes the totals in each category. FIXFORM includes single transactions which usually occur once a month. These single transactions are also summed up by category in the TOTFORM. In TOTFORM, entries in columns b and d are for convenience and are optional. Figure 2 gives examples of the various formulas used. Notice that the formulas in column i of the TOTFORM sum entries between two lines which contain only 0. Column b of these two lines shows that the code numbers of the transactions fall between the code numbers of these two lines. I call these the delimiting lines, or delimiters.

Summary Of The Initial Set Up Of The Forms

I shall describe the initial steps in setting up the forms with the help of Figures 1, 2 and 3. The minimum content of the

INISHEET.MC

```
 b     c  d      e            f                      g             h          i
                      FIXFORM, FIXED EXP  February        1986
                        CATEGORY            BILL                           AMT  PD
 110  o5              CHARGE ACCTS.       Macy                               35.00
 130  o6                                  Heath-Zenith                       40.00
 140  o7                                  J.C. Penney                            0
 180  o8                                  Sears                                  0
 310  o9              UTILITIES           Gas                                    0
 340  o10                                 Electr                                 0
 370  o11                                 Phone                                  0
      o12
CODE  o13 RNG         TOTFORM, TOTALS     February        1986
 110  o14  180        CHARGE ACCOUNTS                                        75.00
 310  o15  370        UTILITIES                                                  0
 600  o16             CASH                                                       0
1010  o17             GROCERIES                                                  0
1810  o18      SALARY   Al                                                       0
1820  o19               Judy                                                     0
      o20
      o21 DATE CHK.NO  TO OR FROM WHOM          AMT.
 100  g22                                            0 SAL TOT               0
 110  g23 2/1      768 Macy                      35.00 EXP TOT           75.00
 130  g24 2/1      769 Heath-Zenith              40.00 NET              -75.00
 200                                                 0
 599                                                 0
1000                                                 0
1800                                                 0
1815                                                 0
1900                                                 0
1820  g31 1/31          Judy                    238.74
1010  g32 2/12     812 Kroger                    72.36
 340  g33 2/3      783 Ga  power                 72.36
 600  g34 2/14          inst bank                50.00
1810  g35 1/31          Al                     1453.23
 600  g36 1/31          cash frm salary          85.00
 600  g37 2/22          inst bank                70.00
 180  g38 2/1      771 Sears                     50.00
 140  g39 2/1      770 J.C  Penney               30.00
 310  g40 2/3      782 Atl  Gas                  56.74
1010  g41 2/1      765 Kroger                    56.74
 370  g42 2/3      784 Phone                     27.66
1010  g43 2/21     819 Kroger                    64.86
```

```
 b     c  d      e            f                      g             h          i
                      FIXFORM, FIXED EXP  February        1986
                        CATEGORY            BILL                          AMT. PD
 110  o5              CHARGE ACCTS.       Macy                               35.00
 130  o6                                  Heath-Zenith                       40.00
 140  o7                                  J.C  Penney                        30.00
 180  o8                                  Sears                              50.00
 310  o9              UTILITIES           Gas                                56.74
 340  o10                                 Electr                             72.36
 370  o11                                 Phone                              27.66
      o12
CODE  o13 RNG         TOTFORM, TOTALS     February        1986
 110  o14  180        CHARGE ACCOUNTS                                       155.00
 310  o15  370        UTILITIES                                             156.76
 600  o16             CASH                                                  205.00
1010  o17             GROCERIES                                             193.96
1810  o18      SALARY   Al                                                 1453.23
1820  o19               Judy                                               1238.74
      o20
      o21 DATE CHK.NO  TO OR FROM WHOM          AMT.
 100  g22                                            0 SAL TOT 2691.97
 110  g23 2/1      768 Macy                      35.00 EXP TOT  710.72
 130  g24 2/1      769 Heath-Zenith              40.00 NET     1981.25
 140  g25 2/1      770 J.C. Penney               30.00
 180  g26 2/1      771 Sears                     50.00
 200                                                 0
 310  g28 2/3      782 Atl. Gas                  56.74
 340  g29 2/3      783 Ga. power                 72.36
 370  g30 2/3      784 Phone                     27.66
 599                                                 0
 600  g32 1/31          cash frm salary          85.00
 600  g33 2/14          inst bank                50.00
 600  g34 2/22          inst bank                70.00
1000                                                 0
1010  g36 2/1      765 Kroger                    56.74
1010  g37 2/12     812 Kroger                    72.36
1010  g38 2/21     819 Kroger                    64.86
1800                                                 0
1810  g40 1/31          Al                     1453.23
1815                                                 0
1820  g42 1/31          Judy                   1238.74
1900                                                 0
```

Note: To facilitate locating lines in this Figure, lines 5–21 of column c contain a dummy formula. The resident formulas in the CHKFORM part serve this purpose.

Figure 1
A sorted worksheet in its final form.

CHKFORM are the delimiting lines. Figure 3 shows a portion of the initial setup to illustrate the relationships between TOT and CHKFORMs. The transactions of the month, with their assigned code numbers, are entered at the bottom of the CHKFORM, after the delimiting lines. At that stage, the code numbers of the transactions are not ordered. With the automatic update on, the sheet is sorted by defining as range the first and last entry in column b. The result is shown in Figure 1, that is, all the transactions of a specific category are gathered between the two delimiting lines which are defined in the formulas of column i of the TOTFORM. If you compare Figures 1 and 3, you will notice that although the line numbers are different, in both cases these line numbers refer to the same delimiting lines. I shall give more necessary details for this initial setup, but we can examine now why and how this ordering takes place.

The Formulas That Make It Work

The reason that upon ordering the transactions of each category fall between the delimiting lines specified in the "sum" formulas in column i of the TOTFORM is that their variables are set as absolute. As I understand it, "absolute" refers to the contents of a specific cell, which is located at the time of definition at some line. Usually, the position of this line remains unchanged. An example is the range in the "lookup" formulas in Figure 2. In our case, when by sorting the location of the line changes, all the formulas that refer to the contents of this line are adjusted so as to refer to the new location of the line. To summarize, TOTFORM works because we have introduced delimiting lines and because we set the variables in the formulas in column i of TOTFORM which refer to these lines as absolute.

In FIXFORM each transaction has a unique code number and these numbers must be the smallest ones possible in the sheet, for reasons to be explained. After sorting, all the transactions in FIXFORM are at the top of the CHKFORM data. The problem is to

find the line of a transaction with a given code number in the CHKFORM and thus transfer the dollar value from that line to FIXFORM. The "lookup" formula is one tool. The "lookup" formula is described in the MYCALC manual and has been explained further in an excellent article in REMark (H. W. Bauman, REMark Issue 61, p.7 (1985)).

Cell	Formula	variables Rel. or Abs.	Range where formula occurs.
i5	if(k5=b5,j5,0)	rel, rel, rel	i5 - i11
j5	lookup(b5,b22.b50)	rel, abs, abs	j5 - j11
k5	lookup(b5,a22.a50)	rel, abs, abs	k5 - k11
i10	if(k10=b10,j10,0)	rel, rel, rel	i5 - i11
j10	lookup(b10,b22.b50)	rel, abs, abs	j5 - j11
k10	lookup(b10,a22.a50)	rel, abs, abs	k5 - k11
i14	sum(g22.g27)	abs, abs	i14
i17	sum(g35.g39)	abs, abs	i17
i22	sum(i18.i19)	abs, abs	i22
c35	=g35	rel	c22 - c43 (1)

Note 1: The cells in column c of delimiting lines need not contain this formula.

Figure 2
Examples of essential formulas of the spreadsheet in Figure 1.

Briefly, the "lookup" formula is given a number K to lookup in a column of numbers arranged in increasing order and when it reaches a number L which is larger than K, it "goes back" to the previous entry and returns the value in the column adjacent to it, to the right. This is true for MYCALC but other sheets may do it a bit differently. Notice that the code number of each transaction in the FIXFORM is given in column b of the FIXFORM. In column j, the formula searches column b of the CHKFORM for a specific code number and returns the value in column c, which is a duplicate of the dollar values in column g. If the "lookup" formula stopped always at the true number, this search would be sufficient. Unfortunately, as seen from the definition, the "lookup" formula actually stops at a number J <= K. Thus, to pin the desired number down we must use an additional trick. In column k, the "lookup" formula searches column a of the CHKFORM and it returns the value in column b. Column b is identical to column a, so the number found is the code number located in the same line whose dollar value is given in column j. Using all this information, the "if" formula in column i can now resolve the issue. If the code number in column k of a particular line is the same as the code number in column b of the same line, then the dollar value in column j is the correct one and is printed in column i. If not, the bill has not been paid and column i shows 0. It took some doing, but MYCALC did find the needle in the haystack. Since one of the requirements of the "lookup" formula is that no duplicate numbers are present in the column to be searched, the transactions which are to appear in FIXFORM must be assigned the smallest code numbers possible, so that they occur first in the CHKFORM data list, before multiple entries with the same code number, such as "cash," are encountered by the "lookup" formula.

More On The Setup And Operations Of The Spreadsheet

Now that we understand why and how the spreadsheet works, I shall give some details on the setup. As we know, the minimum number of entries in the CHKFORM are the delimiting lines. The data for each month would be entered last, at the bottom of the CHKFORM. The filled sheet would then be saved under a different file name indicating the particular month. If you prefer to type in all names in column f each month, you should save CHKFORM with the minimum contents. Those of us who are slow typists might save the CHKFORM part of the sheet with some minimum amount of entries such as the names which occur in the FIXFORM each month and a few others such as "cash." The CHKFORM to be saved and used each month includes no dates, check no. and dollar values (except for the 0s in the delimiting lines.) It is helpful to keep the 0s in the delimiting lines for identifying the delimiters. After arriving at a desired format of the CHKFORM, the spreadsheet is printed with columns c and i set to TEXT format so the line numbers shown in the formulas in column c of the CHKFORM can be used to locate the line numbers of the delimiting lines. The range of each sum(...) formula in column i is set so that the entries of each category would fall between two successive delimiting lines.

The variables in the formulas of FIXFORM include the line numbers in which each entry in the FIXFORM occurs and a range in CHKFORM whose first variable is the line number of the first delimiting line and the last one is large enough to include at least all entries which would appear in FIXFORM. As Figure 2 indicates, both variables in the range are set as absolute. Note that those who do not want to use the FIXFORM part can eliminate columns a, c, j and k.

After you enter the data in CHKFORM, you need to fill columns a and c in the lines of the new data entries. For column c you GET the formula and you PUT it over the range, setting it as relative. For column a, those of you with computer memory to spare could simply fill column a with the formula for line x as "=bx", again set relative. If you need to conserve memory, you GET the range of new entries in column b and PUT it in column a. You then save the filled sheet under the file name of the month. It is advisable to save the sheet before sorting and also make a backup copy of it.

Sorting

The main things one should keep in mind is, first, NEVER delete or add columns or lines, or SORT without the auto update on. Data can be entered at the end of the CHKFORM and values entered in existing lines in the CHKFORM with the manual update on, to save time. The column to be sorted is column b, the one which contains the code numbers. The sorter in MYCALC will sort in ascending order only if the pointer is located at the bottom cell of column b. The range specified must be the cell locations of the first and last code numbers in CHKFORM. Please, do not use as the starting cell an empty cell above the first entry. Your sorting will never end, as bitter experience taught me.

Procedure For Computers With 64K

With SYSGEN/MIN the amount of free memory is about 12K, not enough to have simultaneously all three parts of the sheet together plus a fair amount of entries. Thus, the three ???FORMs are saved as separate files. The rest of this describes ways which can be used to handle over 100 entries (including the delimiting lines.) The procedures and precautions described above hold also here. The CHKFORM used should have as many lines empty at the top as the number of lines of the longest FORM of FIX or TOT. Thus, when either FIX or TOT are merged with the CHKFORM, there is no overlap of lines.

First, the entries of a particular month are entered in the CHKFORM and SAVEd as a separate sheet, like CHK286, without sort-

ing, to preserve the references of the formulas in TOTFORM to the original line locations in CHKFORM. The CHK sheet still in memory is then reduced in size by deleting the contents of columns d through f (not the columns themselves), that is, date, check number and payee or payer. If more memory is needed, one could also delete the contents of columns a and c, necessary only for the FIXFORM. With the auto update on, the TOTFORM is merged via READ and then SORTed. If you operate at 2 Mhz like I do and you have about 120 entries, it is time to go for a 15 minute coffee break. Sorting is slow because updating takes place constantly during sorting. For the fun of it, sort a spare sheet with the auto update off. The sorting will be fast, but the results disastrous. To have a sorted, detailed record of your monthly transactions, you would sort your CHK286. Since there would be several empty lines at the top of CHK286, in specifying the sorting range, make sure to use the first line which has a code entry in column b. To repeat, if you start at an empty line, your coffee break will last till you give up and reboot!

To obtain the values with the FIXFORM, you would proceed as above, with the following differences. Before merging the FIXFORM with a copy of the SORTed CHK286, you need to reduce the size of the CHK286. Since all your data for the FIXFORM should be near the top of a SORTed data file, you can delete all entries with code numbers higher than the highest one in FIXFORM. Additional memory can be made available by deleting the contents of columns d–f as above. With the auto update on, you then merge the FIXFORM with this version of CHK286. There are several other details which I am not elaborating on because I assume you are already familiar with the operations of MYCALC. For example, in setting up or modifying your ???FORMS, you need to have either FIX or TOTFORM merged with CHKFORM, to adjust the formula parameters. You would then need to save, for example, two different files, a new or modified CHKFORM and also a TOTFORM or FIXFORM. The manual describes how to save portions of a spreadsheet.

```
CODE o13 RNG        TOTFORM, TOTALS February   1986
 110 o14  180       CHARGE ACCOUNTS            sum(g22.g25)
 310 o15  370       UTILITIES                  sum(g25.g26)
 600 o16            CASH                       sum(g26.g27)
1010 o17            GROCERIES                  sum(g27.g28)
1810 o18     SALARY    Al                      sum(g28.g29)
1820 o19               Judy                    sum(g29.g30)
     o20
     o21 DATE CHK.NO TO OR FRM WHOM  AMT
 100 g22                             Ø SAL TOTsum(i18.i19)
 110 g23            Macy               EXP TOTsum(i14.i17)
 130 g24            Heath-Zenith       NET     i22-i23
 200                                   Ø
 599                                   Ø
1000                                   Ø
1800                                   Ø
1815                                   Ø
1900                                   Ø
```

Note: For the contents of lines 22 -24 see "yearly summary" in text. The CHKFORM part is in its minimum configuration.

Figure 3
The TOTFORM and CHKFORM parts of Figure 1 before entering transactions.

Optional Checkbook And Other Functions

Customizing the spreadsheet is, of course, up to you. Since many use a checkbook spreadsheet, I shall suggest an easy way to use CHKFORM for that purpose. The configuration of CHKFORM should be the minimum one, that is, it should contain only the delimiting lines. During a month, you would keep one copy of CHKFORM with the monthly entries for use as a "checkbook" and another one for your monthly totals. Your checkbook copy is never sorted. Let me point out that a recent article in REMark, MOVIT, described a set of programs in BASIC to link two MYCALC spreadsheets. This article also contained formulas in MYCALC format which can be used for checkbook functions (L. M. St.Martin, REMark issue 70, p.27 (1985)). You would expand CHKFORM by utilizing columns past column i for the checkbook functions. There are mainly two ways to do this.

In the first, you would use a separate column to enter income or credits, as in MOVIT. You would then adjust the sum(...) formulas in TOTFORM to pick income entries from that column. The other way is to keep both debits and credits in one column, g, as in this spreadsheet. You could then flag credits by entering, for example, "1" as check number in column e. An "if" formula in the balance column could be used to determine whether the amount should be added or subtracted. To repeat, you do not SORT this sheet. When you want to sort to obtain your totals, you make a copy of it for sorting. Those of you with 64K memory who would have the CHKFORM as a separate file, would need to also delete those additional "checkbook" columns before merging (READing) the CHKFORM and TOTFORM for sorting, to save memory.

Another useful addition which was suggested to me by my daughter, a budding CPA, is to keep track of expenses which include sales tax. In the TOTFORM you may add a line, "total of sales tax items", and enter a formula to sum up all expenses which include sales tax. Some expenses such as car repairs might be a mixture of services and products. You could divide them into two parts, if you care to.

Summary Of The Whole Year

You may have noticed that in Figure 1 column i extends into the range of the CHKFORM where data are further summarized. In my TOTFORM, I have categories under "deductible" (not shown in Figure 1) such as "computer", "donations" and "dues." I summarize all these as "deductibles" in a line in column i which extends into the CHKFORM range. It does not look pretty, but it is there. Thus, for reasons of convenience, all information for a monthly summary is found in column i. One of the features of MYCALC is that it can refer to other sheets, provided they reside on the currently logged disk. See p.31 of the MYCALC manual.

Figure 4 is a spreadsheet, let's call it SUM86, which illustrates how to collect the data for two months into a summary. In practice, this is extended to twelve months. The first column lists the various categories. The column for February lists the values obtained from the data of Figure 1. The column for January lists the formulas, via FORMAT TEXT, to illustrate the various formulas used. Those with hard-sectored, 94K diskettes could not fit all monthly files in the logged drive. In this case, a separate file is created which includes only the range of column i of the main sheet which has the totals. For February, I labelled this SUM286 and it is shown in Figure 4 in MYCALC format. This small file is then transferred to the logged disk. Especially, those of you with 64K who may have this column extend into the CHKFORM range, note that you either have no check entries in the overlap lines of CHKFORM or entries with the smallest code numbers, and sorted. Otherwise, upon sorting your summary lines would also be moved elsewhere and indicate "Error!" Compare the two entries in Figure 4 with those in Figure 1.

Part A
Spreadsheet file SUM86, yearly summary, but shown for two months.

a	b	c	d
	TOTALS	1986	
CATEGORY	January	February	TOTALS
CHARGE ACCTS	sum186[i14]	155.00	283.00
UTILITIES	sum186[i15]	156.76	339.11
CASH	sum186[i16]	205.00	485.00
GROCERIES	sum186[i17]	193.96	366.52
SALARY	sum186[i22]	2691.97	5383.94
EXPENSES	sum(b5.b8)	710.72	1473.63
INCOME	sum186[i22]	2691.97	5383.94
NET	b12-b11	1981.25	3910.31

Part B
Listing of file SUM286.MC in MYCALC format.

```
afgt
>i14:=sum(g22.g27)#1.55E+02
>i15:=sum(g27.g31)#1.5676E+02
>i16:=sum(g31.g35)#2.05E+02
>i17:=sum(g35.g39)#1.9396E+02
>i18:=sum(g39.g41)#1.45323E+03
>i19:=sum(g41.g43)#1.23874E+03
>i22:=sum(i18.i19)#2.69197E+03
>a23:\aflt
>i24:=sum(i14.i17)#7.1072E+02
>a24:\aflt
>i25:=i22-i24#1.98125E+03
```

Figure 4
Files used in summarizing transactions for the year.

In Case Of Difficulty

I could not have been a HEATHKITite for 25 years and not include such a section. I have used this sheet for four plus months and it has worked fine. I have added and deleted categories nearly every month and I made occasionally two main mistakes. One is to forget to have the auto update on when needed. The other is not setting the formulas in TOTFORM as absolute after editing them. The result is that the ranges in the formulas are different from the actual positions of the delimiting lines and must be edited. This may not give an "Error!" message. Print your sheets with columns c and i set in TEXT format which shows the formulas and thus locate the appropriate lines. When editing, MYCALC may not ask "absolute or relative?" Finish editing the formulas and go back, GET each formula and PUT it on the same spot. This forces MYCALC to ask the above question.

Another source of error is deleting a delimiting line. The formulas which refer to it will indicate "Error!" This happens whenever formulas elsewhere in the sheet refer to a line which is deleted. See p.50 of the MYCALC manual. If the spreadsheet you use has the features of MYCALC, it might work. You might check it first with a small sheet as in Figure 1. I would appreciate knowing if someone found some bugs. Last summer I had my first exposure to spreadsheets when I got MYCALC and I am not an expert in this area since I am a chemist by profession. So, do ask me questions but please, do not ask me any "deep" ones.

A List Of The Categories

I keep a file of categories with their code numbers, which helps in entering code numbers in my checkbook. You could prepare such a list by eliminating all unnecessary columns in a copy of FIXFORM and TOTFORM and reshuffling the appropriate lines. I have more code numbers than those included in my two FORMS, such as code numbers for each one of our regular doctors and gasoline companies. These would be typed into the list and the list sorted by code numbers.

On Ordering Random Data And MOVIT

I understand that PC software packages which include a database, spreadsheet and text editor can take data in any form and with the proper manipulation can put them in any desired form in proper places in a spreadsheet. With less powerful software, this has been a problem. The power of spreadsheets depends on having the proper data in defined, known cells. On the other hand, programs written in BASIC or another language can find a needle in a haystack. An ingenious approach to transferring data from one spreadsheet to another, called MOVIT and referred to above, is a set of four programs written in BASIC. MOVIT shows one way by which a language, such as BASIC, can communicate with such a different piece of software as a spreadsheet. Note, however, that this is possible because MYCALC saves its files in ASCII form and can be edited by and read into almost any program which can use ASCII files. MOVIT uses another language to manipulate and transfer data from one sheet to another externally, that is, it serves as a link. On the other hand, the sheet presented here manipulates and puts in order data internally. In a sense, CHKFORM may be thought of as a small database in which data can be stored. The "language" however is the same, MYCALC format, so the data storing unit, CHKFORM, can simply be merged with the data processing units, TOTFORM and FIXFORM. I hope that the basics of it could be used for ordering data in other cases, so that users of spreadsheets might have greater freedom in entering and manipulating data. *

Western Regional

Heath/Zenith Users' Group

Conference

- LATEST TECHNOLOGY
- NEW SOFTWARE
- HEATH/ZENITH VENDORS
- GUEST SPEAKERS
- PANEL DISCUSSIONS
- DINNER/PRIZE DRAWING

NOVEMBER 14 - 16, 1986

AT... Disneyland® Hotel ANAHEIM, CA.

The Official Hotel At The Magic Kingdom

1986 WESTERN REGIONAL HEATH/ZENITH USERS' GROUP CONFERENCE

A CONFERENCE OF HEATH/ZENITH COMPUTER USERS AND LOCAL USER GROUPS.

SPECIAL GUEST SPEAKERS - PANEL DISCUSSIONS - HEATH/ZENITH COMPUTERS - ELECTRONICS - AND HEATH/ZENITH RELATED VENDORS.

CONFERENCE OPENS FRIDAY AFTERNOON ON NOVEMBER 14TH AND CLOSES AT NOON ON SUNDAY, NOVEMBER 16TH.

☆ *[Advance Registration Rates Good Thru October 15th]* ☆

NOVEMBER 14 - 16, 1986

DINNER/PRIZE DRAWING $25.00
WITHOUT DINNER $7.50

THE DISNEYLAND HOTEL IN ANAHEIM, CALIFORNIA

— WIN A COMPUTER —

THOSE WHO ATTEND THE DINNER* ON SATURDAY, NOVEMBER 16TH ARE ELIGIBLE FOR NUMEROUS PRIZES INCLUDING

(1) HS 241-81 (Extended Drive) With Video Card and Monitor

(2) ZP150 **(1) Hero Jr.**

— SPECIAL DISNEYLAND HOTEL RATES —

Single — $76.00
Twin/Double — $76.00
Triple/Quad — $86.00

*You must be present to WIN!!

PLEASE COMPLETE FOR EACH REGISTRANT:

MAKE CHECKS PAYABLE TO: **WESTERN REGIONAL HUG** CONFERENCE
Mail To: ● 1555 N. Orange Grove Avenue, Pomona, CA 91767

NAME _____
ADDRESS _____
CITY,STATE,ZIP _____

CHECK ONE
__ Admission includes Dinner/Prize Drawing $25.00 ($28 after 10/15)
__ Admission WITHOUT Dinner $7.50 ($10 after 10/15)

NAME _____
ADDRESS _____
CITY,STATE,ZIP _____

CHECK ONE
__ Admission includes Dinner/Prize Drawing $25.00 ($28 after 10/15)
__ Admission without Dinner $7.50 ($10 after 10/15)

NAME _____
ADDRESS _____
CITY,STATE,ZIP _____

CHECK ONE
__ Admission includes Dinner/Prize Drawing $25.00 ($28 after 10/15)
__ Admission WITHOUT Dinner $7.50 ($10 after 10/15)

NAME _____
ADDRESS _____
CITY,STATE,ZIP _____

CHECK ONE
__ Admission includes Dinner/Prize Drawing $25.00 ($28 after 10/15)
__ Admission WITHOUT Dinner $7.50 ($10 after 10/15)

Your Word Processor Speak BASIC?

Lawrence J. Durney
12 Sunset Drive
North Caldwell, NJ 07006

Working as a consulting chemist, I have had to write a lot of specialized BASIC programs to store, and/or search through different types of information — toxicology files, raw material files, literature files, etc., etc.

Because progress in many of these areas is so rapid, there is a frequent need to change and/or add to this information. Additionally, since I am not the world's greatest typist, typographical errors are not uncommon. Since most of the files do not lend themselves to fixed field filing, sequential files have to be used, and making corrections, changes and additions through the BASIC program can be time-consuming and/or frustrating.

Now I use Magic Wand, my word processing program from Peachtree.

A Few Tips On How To Make It Work

1. Generally, all the information must be in ASCII form including the filename. Write your programs so everything is stored in simple strings. For example if your program uses a line

   ```
   INPUT "ENTER FILENAME ";F$
   ```

   and you enter B:ID#108, your word processor will probably call it up. If instead you use the line,

   ```
   INPUT "ENTER ID#";A
   F$="N:ID#"+A
   ```

   your word processor may not be able to access the resulting file.

   Programs using DATA statements present a similar problem, but they are easy to correct with the BASIC editor, so we can forget them.

2. You must use extreme care NOT to add any extra "'s or CR's by mistake.

3. In the beginning for sure, and even after lots of practice occasionally, correcting this way will throw a file out of kilter. You will add a CR or a ", or drop out a string, or forget to adjust a loop counter, or something else. Then when you try to run the program, you will get an error message, generally "OUT OF DATA" or "INPUT PAST END", and you will be faced with tracking down where the error was made. Now that I have been using the system for several months, I have worked out techniques for quickly pinning down the location of the error, but in the beginning it was very time-consuming. So never work with an original copy of an important file. Copy it to a temporary disk, correct it, and run it to make sure it is O.K. Then copy it back over the original.

What Can You Do With This System?

1. Correcting Typos — Call the file into the word processor in edit mode. Move through the file making corrections as needed. Save it back onto the disk. Erase the backup copy the word processor created.

   This procedure works great for deleting material also. Just be careful not to delete less than nor more than a full "WRITE" unit or your program will suddenly go crazy.

   If you only have to correct a single typo, it is really easier to go through the BASIC program. The most efficient use of this system is to wait until you have several corrections to make, and then do them all at once.

2. Rearranging Sequential Files — Do you have a sequential file that is a) cumbersome to sort, or b) a sorted file to which you want to add something?

   A. Call the file into edit. Use the block move function to rearrange as desired. Save it to disk. Erase the backup. Watch this procedure carefully. It is very easy to add extra CR's when moving blocks around. Generally, it is good practice after you have completed all your moves to quickly review the entire file for this problem before putting it back on disk.

B. Run your BASIC program and add the material to the end of the file. Use A above to relocate the material in its proper position.

   This procedure is especially helpful with sequential files storing data that varies in length, e.g. abstracts, toxicology citations, specifications, procedures, etc. This is usually done by using a loop and a loop counter to input and keep track of the number of lines of data in each entry. You can also add or delete lines to or from existing entries, as long as you remember to adjust the stored loop counter value to reflect the changes made.

3. Want to quickly make some changes in a BASIC program to adapt it to another use?

   Call the program into memory through the BASIC interpreter. Save it in ASCII form (SAVE "FILENAME",A). Call it into edit in the word processor. Change as desired. Save to disk under a new filename (don't forget the .BAS designation). Call it into BASIC, and save in normal BASIC form (takes a little less space on the disk). For major alterations, this is much faster than making the changes line-by-line with the BASIC editor. For example, you can use the search function to locate lines that need changing (as long as they all contain at least one of the same characters), or the search and change function to automatically change all occurrences of a word or a string designation — e.g. "Disk#" to "Code#" or C$ to C1$.

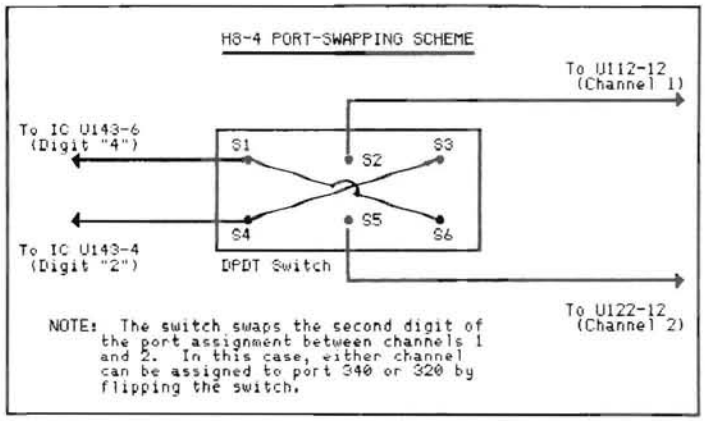
4. Some files getting too long? Would they benefit from being split into shorter files? Call into edit. Use the "block extract" function to extract chunks onto another disk under a different filename. Now you have the shorter files you wanted.

   Or, would you like to have specialized files made up of groups from the master file? Easy! Call into edit. Delete everything you don't want (the "block delete" function is very rapid for this). Review the selected file to make sure that it is the way you want it. Save under a different filename. Now you have the selected file without losing the original. (Of course, you could probably write a short BASIC program to do this for you, but I have found this system to be even faster.) This works especially well for address lists (get state listings or even city listings), specification files (get all the specifications pertaining to a single subject), etc.

5. Want to include a BASIC program in an article or report? Easy! Load the program into memory and save in ASCII form (see 3 above). Load your article or report in the edit mode of your word processor. Use the "include" function to incorporate the BASIC program. Make whatever cosmetic changes are needed and print. (There is the added advantage to this system that there is no chance of a typo while copying the program from a regular listing.)

   There must be lots of other things that can be done too. I just haven't found them all yet. But the ones I have found sure make life easier. Try it – you'll probably like it! *

For HDOS Or CP/M. Install A Switch To Swap Ports On Your H8–4 Interface Board



Jim Meyers
13A Riggs Parkway
Las Vegas, NV 89115

Are you driving two printers off of your H-8 computer and H8–4 serial interface card? If so, then you know how annoying it is to discover that the printer you intended to use for a particular task is inadvertently not the one that is currently active as the LST device. Ever wish you could just flip a switch to select your primary listing device? Been thinking of buying an A–B switch? If you answered "yes" to any of these questions, then you should continue reading this article.

I needed a method to drive my TP–II printer and H–25 printer as the LST device under CP/M. In CP/M, it's particularly inconvenient to use STAT and CONFIGUR continually to make one printer or the other the LST device. Since I can make both of my printers have identical RS–232 configurations through internal DIP switches, it seemed obvious that an A–B switch could provide a good solution to my problem.

An A–B switch, for those readers who may not know, is a peripheral control device that primarily serves two functions. It can be connected to switch one peripheral device between two computers or it can be switched to select either of two peripheral devices from a single computer output port. Since printers fall into the category of peripheral devices, the A–B switch solution seemed promising.

For my application with the A–B switch I imagined that I would run a cable from my H8–4 card into the A–B switch, two RS–232 cables would then emerge from the A–B switch, each going to one of my printers. Turning the knob on the A–B switch would make either printer active. It was a great plan until the time I checked the price of A–B switches; seventy–five to one hundred dollars! It didn't seem possible that a switch to do such a simple function could cost that much.

My computer budget is always limited, so I realized I was going to have to innovate to get what I wanted. I didn't want to rig some type of switch to swap only two of the RS–232 lines as many people had done. Instead, I wanted to properly swap every RS–232 line just as would an A–B switch.

Typically, the reaction of most of us hobbyists when faced with such a problem is to build anything that we can't afford to buy assembled. I've always considered myself a hobbyist so I began to check the prices of a switch that could flip all those lines ($25.95), the cable ($24.95), the connectors ($5.00 each, need 3), the box ($4.50), the little rubber feet ($1.95) . . . wait! I soon realized that I couldn't afford to build the darn thing either. No wonder they cost so much.

So the promise of the easy hardware solution faded and I applied myself toward working the change through software (typical hobbyist "Alternate Plan B"). The only cost to me in following this track would be dozens of hours consumed in modifying the CP/M BIOS.

I dropped this idea when I convinced myself that even if I did find a way to make another LST device port I'd still have to use STAT and CONFIGUR or possibly SUBMIT and XSUB to swap printers. A switch could also provide a direct means of control and by simply observing the direction of the switch toggle, I could verify which printer was selected before I pressed RETURN to send my file on its way. My focus returned to the switch idea and I puzzled over just how to achieve my goal inexpensively. The result is this modification which uses only four wires and a small DPDT switch.

This mod also works for CP/M systems, but since it's much more convenient to drive different devices under HDOS than CP/M, you may not want to bother implementing the hardware change. However, if you find that it's a nuisance to keep track of which program is using which device driver, then this mod will help you to maintain control.

In the interest of full disclosure, I caution you that if your printers can't be set up so they can be driven similarly then this mod won't solve your problem. Also, if you were looking to free up a port on your H8–4 card by using an A–B switch, then this modification won't solve that problem either. Lastly, if you can afford the big bucks for a printer switch and need one, then go ahead and buy it. They're truly handy and versatile devices.

The mod instructions assume that you have one printer connected to port 340 on channel 2 and another connected to port 320 on channel 1, and that these are the devices you would like to swap with your "printer switch." If you followed Heath recommendations when you installed both your printers, those are the ports you probably selected. The channel numbers are silk-screened on the H8-4 card and are simply used to identify which of the four locations your I/O cables are attached.

Each channel on the H8-4 was originally programmed by jumpers to assign a certain port to that particular channel. Moving the jumpers enabled you to assign the ports to certain channels. This mod allows you to "swap the jumper" of the second digit between two channels by flipping the switch. Here it will swap the "2" and the "4" so that what was port 320 will become 340 and vice versa.

So it's IMPORTANT that you remove the original jumpers at digit two and four since the switch is being installed to replace them. If you forget to remove the two jumpers, the mod won't work.

How the switch works is simple. Notice that the digits we want to swap are connected to the stationary contacts on the DPDT switch. Each of the two wires from S2 and S5 are connected to the channel numbers you wish to use during the port swapping. The switch is wired so that it reverses the connections when it is thrown. Everytime you flip the switch it's the same as if you swapped the jumpers for the second digit.

You can grasp the concept of how this modification functions by referring to your H8-4 schematic and observing where the new wires will be connected. Then, you'll be able to go a step further and swap any ports between any two channels by connecting the wires to different digits on IC-143 and connecting the switch wires from S2 and S5 to whichever channels you choose to control.

If you study the H8-4 schematic, you may notice that pins exist for the connections that I've chosen to solder in this mod. So if you have the necessary shells and contacts, you may wish to make the mod removable without desoldering by making all connections with plugs. I may do this myself someday, but right now I'm having too much fun working efficiently with two LST printers to get the soldering iron out again.

H8-4 Port Swapping Modification

Materials Required:

1. About 8 feet of small gauge stranded wire cut into 2 foot lengths.
2. Two short pieces of small gauge insulated solid wire, such as wire-wrap wire, to use to jumper the DPDT switch.
3. A small DPDT switch such as Radio Shack 275-626.
4. Some small nylon cable ties or lacing cord.
5. Electrical tape or silicone sealant to insulate switch terminals.

Tools Needed:

1. Diagonal cutting pliers.
2. Low wattage soldering iron and rosin core solder.
3. 1/4-inch standard screwdriver.

Optional, but recommended:

1. Bright work light.
2. Magnifying glass.

( ) Unplug the H-8 from AC power and disconnect all cables to the computer. Label all disconnected cables. For safety's sake don't plug the H-8 back in until it is called for in these instructions. You won't gain anything by injuring yourself or damaging your computer. Be careful.

( ) Remove the two screws from the top cover. Lift off the top cover.

( ) Loosen the two screws at the back panel strain relief.

( ) Label all cables to the H8-4 board and remove them from the board.

( ) Loosen the screws at the top of the heat sink tie bar and remove the screw from the rear of the tie bar. Slide the tie bar off and out of the computer.

( ) Place the computer on its side to expose the screws on the bottom of the H-8 which hold the circuit boards in place.

( ) Remove the screw which holds the H8-4 card in place.

( ) Slide the H8-4 card out of the computer and remove it to your work surface.

( ) Remove each of the original hardware jumpers installed at the second digit position from the two channels as explained in the text.

( ) Install two short, insulated wire jumpers on a small DPDT switch as shown in the illustration to connect S1 to S6 and S4 to S3.

Note: Be careful when clipping wires with diagonal pliers. Hold the end of the wire so it won't fly into your eyes. Also, don't let wire clippings accumulate around the circuit card or your work area. Imagine what a headache a small piece of stray wire, shorting out IC pins or circuit board traces, can cause you.

( ) Connect four long wires (length to suit your purpose, recommend about 2 feet) as follows:
1. U143-6 to S1
2. U143-4 to S4
3. U112-12 to S2
4. U122-12 to S5

Bundle the wires from the switch to the card together to make a neat cable.

( ) Route the switch and its wires through the strain relief in the rear panel of the H-8 from inside the computer and slide the H8-4 card in place.

( ) Reinstall all I/O cables on the H8-4 board. Use extreme care to line up connectors carefully and orient them properly. You did label everything, didn't you?

( ) Secure the circuit board using the tie bar and the screws. Secure the strain relief. Replace the cover.

Hint: I never liked the way the H-8's metal strain relief pressed into the cables that passed through it, so long ago I lined both plates of the strain relief with 1/2-inch wide foam self-stick weather-stripping. This helps to hold all the cables securely without using excessive clamping force.

( ) Mount the switch to suit your preference in a suitable location. It can be mounted in a small plastic parts box or pill bottle with a hole drilled in it or in a commercially available

hobby box. If you intend to just let the switch dangle until you get around to mounting it, use electrical tape or silicone sealant to insulate the connections on the back of the switch.

( ) Reconnect all system cables to the computer and apply AC power. Turn on the power switch. The front panel should illuminate as usual. Turn off the computer. If you experience difficulty, check the list below.

( ) Configure your printers so they're alike. Mine are both set to 4800 baud using high ready polarity (Heath standard for H-25 printer).

( ) Boot up your system. Check that printing action can be directed to either printer by flipping the switch. An easy method is to type Control-P in CP/M and type a few Carriage Returns. Note which printer is active and then flip the switch. Type a few more Carriage Returns and verify that the other printer becomes active. If OK, you're finished.

If you have any problems, check for:

1. Solder bridges on the H8-4 or DPDT switch.
2. Miswiring.
3. Printer configured incorrectly.
4. Printers not connected to the channels used in this mod (i.e. channels 1 and 2).
5. Software not configured for the ports used in this mod (i.e. ports 320 and 340). Run CONFIGUR for CP/M systems or SET LP: for HDOS systems.

Feel free to write to me if you have any comments about this mod. If you expect a reply, a stamped self-addressed envelope is essential. Good luck. *

Related Products

Jim Buszkiewicz
HUG Software Developer

NOTE: *The following information was gathered from vendors' material. The products have not been tested nor are they endorsed by HUG. We are not responsible for errors in descriptions or prices.*

Conversational Computer Systems of San Antonio, Texas is offering a productivity enhancement tool for Turbo Pascal users. SUPERMATH is a 32 bit assembler coded arithmetic toolbox for Turbo Pascal, and includes over 40 different routines and functions. TURBO PACKAGE now enables your Turbo Pascal programs to access 640kb of memory for program code and data. TURBO PACKAGE turns Turbo Pascal into a modular programming system that rivals Modula 2 and Ada. Included free with TURBO PACKAGE is SUPERMATH for $39.95. For more information, contact Conversational Computer Systems, 5371 Verbena Road, San Antonio, Texas 78240, or call between 8:00 am and 10:00 pm CST, (512) 692-0353.

System Peripherals Consultants of San Diego, California are introducing the DISKIT PORTABLE SERIES of hard disk drives, with battery backup for the Zenith Z-171 series of computers. These hard disks feature auto-booting, 20 megabytes, average access time of 65mS, and a patented head lifter which actuates upon power down, preventing the heads from ever contacting the disk surfaces. The drives will withstand up to 60 g's of shock in non-operative mode, and 10 g's during operation. The Z-171 compatible model simply plugs into the expansion port on the Zenith computer, and may be placed underneath or beside the computer. Housed in a cabinet only two inches high, it is small enough to carry in a briefcase. Current models are AC powered, although DC adaptors are available. Prices range from $1095 to $1295, depending on the options ordered. For more information, contact System Peripherals Consultants, 9747 Business Park Avenue, San Diego, California 92131, or call (619) 693-8611.

With the football season practically here, **Generic Computer Products Inc.** is introducing FOOTBALL and FOOTBALL-PICKS. FOOTBALL is a full-featured football game you can play against the computer. FOOTBALL-PICKS is for those who want more excitement out of the final score. Using information found in most daily newspapers, you can obtain game predictions and point spreads for as few or as many NFL games as you want. Each package sells for $24.95, and will run on any Heath/Zenith PC compatible. For more information, contact Generic Computer Products Inc., P.O. Box 790, Marquette, Michigan 49855, or call (906) 249-9801.

EPSON FX-85 Update

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

How many of you Heath/Zenith users out there have an EPSON FX-80 or an FX-100 printer? I'll bet there are a fair number of you, and that includes myself (or it did include myself). Today, I am a proud owner of an FX-85; really it is my old FX-80 upgraded to an FX-85. I use my FX printer, which is connected to both of my machines, an H-100 and a Z-160 via a switch, very heavily, especially for printing of word processing files.

At the Fourth International HUG Conference back in August 1985, I saw an FX-85 there, but really did not get a close look at it or get a clear understanding of what the differences were between it and an FX-80. The gentleman in the booth stated that the big change was the addition of the Near Letter Quality (NLQ) print capability. I didn't pay much attention to the FX-85 (or the FX-185) after that since I was quite happy with my FX-80 with all the various print modes/styles.

However, in late August, I picked up a local Minneapolis area computer newspaper and saw an ad for upgrading an FX-80 printer to an FX-85. The price was $99.00 and NO installation charge if the work was done in September. This seemed real reasonable since I did see somewhere (I can't remember where) that a kit was available for $89.00.

Well, I figured $99.00 with NO installation charge was a "darn good deal", so I called the computer store to get a feeling of what some of the differences were besides the addition of the Near Letter Quality capability. Let me list the major new features/capabilities here and then go through them later in more detail.

New features/capabilities of the FX-85 over the FX-80 (this is also true for the FX-185 over the FX-100):

- Near Letter Quality (NLQ) print capability.
- SelecType was added. This allows the user to select the various print modes/styles via the use of the external buttons on the control panel already part of the FX printer.
- IBM-PC printer capability. This allows printing the IBM character set.
- An 8K input buffer instead of the 2K on the FX-80 and FX-100.
- Cut-Sheet Feeder support.

After hearing the changes, needless to say, I went ahead and had my FX-80 updated to an FX-85. I took the printer in one day and it was ready the next day. Fast service. The only thing that one can see externally that was changed was the template over the buttons. In addition to the ON-LINE/OFF-LINE, FF, and LF button labels, the new template also labeled the FF button as NLQ and the LF button as DRAFT. More about this shortly. My "new" printer came with an EPSON FX-85/185 User's Manual. The only thing that I was not happy with was that there was no documentation on what was changed internally to the FX-80. The service technician stated that there was absolutely none and the EPSON Service Manual was required to install the parts. He also said that only Service Centers were allowed to install the upgrade kits.

However, the service technician did give me some information on what was changed:

- Slave CPU was replaced.
- Mask ROM was replaced.
- Character ROM was changed.
- New board added for the NLQ and IBM character set capabilities.

The User's Manual doesn't explain any of this, so I really can't say much on these internal changes. I ordered a Service Manual to find out a little more on what was changed and how the printer works internally, but it is on back order.

After doing some testing, the original printer functions, modes, and styles work the same as they have been. The new features work as advertised. The Near Letter Quality print is just super. It is really good. See Figure 1 for all the "old" print styles, as well as the NLQ print. In my opinion, the NLQ print alone, warrants the upgrade.

Now let me cover the various new features, that I stated before, in a little more detail starting with the Near Letter Quality capability. How do you select the NLQ print mode/style? It is selectable in two ways. First, via the normal Escape Sequence "ESC x". This sequence is a toggle sequence. Entering this sequence for the first time puts the printer in NLQ mode. Entering it again puts the printer back in DRAFT mode.

The second method for selecting the NLQ print mode, and for that matter, the DRAFT mode, is via the external control panel buttons on the top lower right hand side. It's the "FF" and the "LF" buttons that are used. When the printer is in ON-LINE mode, pressing the "FF" button selects NLQ. Pressing the "LF" selects DRAFT mode. In OFF-LINE mode, the "FF" and "LF" but-

tons perform the same functions that they did before: top-of-form (page eject) and line-feed, respectively.

Before going to the next major feature, let me talk a little on a feature that is part of the NLQ feature, and that is "Justification". Basically, the NLQ feature will be used for letters, articles like this one, and other documentation. Normally, we select the line justification that we want in the wordprocessing package that we use. Likewise, with centering a title or other headings. However, some documents may be created using a non-wordprocessing package, such as a text editor. When these latter documents are printed, it would be nice to have various line justification options available. That is what is provided with the FX-85 and the NLQ capability.

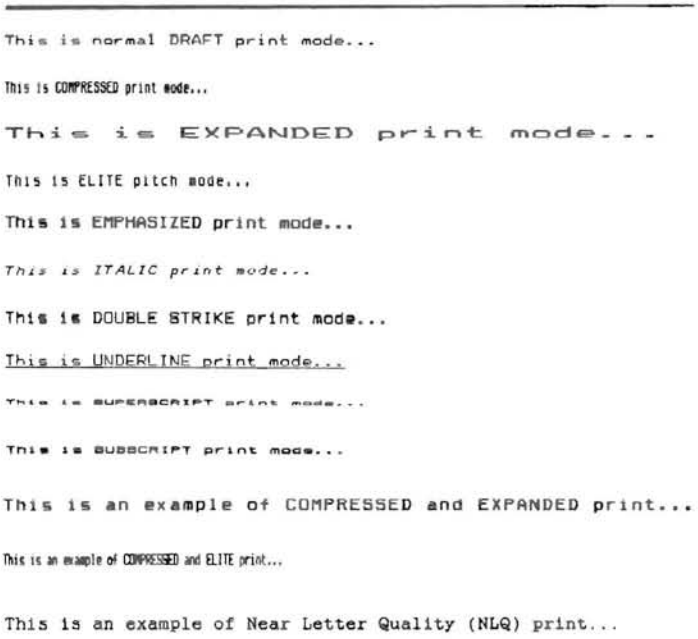


Figure 1
Various Print Modes

There are four choices for formatting (in EPSON mode only; not available in the IBM mode) your document lines:

- ESC a 0 Left Justification
- ESC a 1 Center
- ESC a 2 Right Justification
- ESC a 3 Full Justification

Left justification is the normal or standard mode of operation. This leaves the text as we typed it; even margin on the left and jagged on the right. Right justification is the opposite: jagged on the left and even margin on the right. The Center option is for centering titles and headings. Full justification is the mode that we are used to with wordprocessing packages. Blanks are added where needed to have even Left and Right margins.

Next, the SelecType feature will be discussed. In Figure 1, you will note that all the print modes/styles were done in DRAFT mode, except for the NLQ style, just the "normal" NLQ style. Emphasized and Expanded print modes can also be used with NLQ to produce still more styles. Superscript and Subscript will work with the NLW feature. Italic, Compressed (Condensed), Double-Strike, and Elite modes are not available with the NLQ feature. Remember, SelecType is designed to control the printing of an entire document or file. However, if the document itself has control sequences, they will override the SelecType option(s) that have been selected.

As I said earlier, SelecType uses the external buttons to select the various print styles in either DRAFT or NLQ mode. To use SelecType, you first turn it on, select the print style, turn it off, put the printer in ON-LINE mode, and print your file/document. Before going through the steps involved in the SelecType operations, look at the following Table which shows which print styles or functions are possible and their respective codes.

Code	Print Style or Function
1	Compressed/Condensed
2	Expanded Print
3	Elite Print
4	Emphasized Print
5	Italic Print
6	Double-Strike Print
7	Underline Function
8	Superscript Print
9	Subscript Print
10	Skip-Over-Perforations

With the above Table in mind, we can now proceed to go through the steps necessary to use SelecType. Remember that in ON-LINE mode, the "FF" and "LF" buttons have one meaning and in OFF-LINE mode they have another. However, the buttons have still another meaning when they are used in SelecType mode.

Since all the print styles listed in the Table are not possible in NLQ mode, we will use the DRAFT mode for now. First of all, make sure the printer is in ON-LINE mode and DRAFT print has been selected. Now press the "OFF-LINE" and "FF" buttons simultaneously and hold them down for at least a second. This will put you in SelecType mode. If the printer beeps twice, the "FF" button was pressed before the "OFF-LINE" button. The printer is in NLQ mode now because you were in ON-LINE mode and pressed the "FF" button. To try again, make sure the printer is back in ON-LINE mode, press the "LF" button to select DRAFT mode and press the "OFF-LINE" and "FF" buttons at the same time.

After releasing the "OFF-LINE" and "FF" buttons, you will be notified that you are in SelecType mode by:

- The printer beeps once.
- The READY light is turned off.
- The ON-LINE light begins flashing.

Now that you are in SelecType mode, each of the three external buttons has a particular function:

- The "OFF-LINE" button selects the desired print style or function.
- The "FF" button sets (locks in) or clears the print style/function.
- The "LF" button terminates the SelecType selection mode.

Now with the above functions in mind, we can proceed with the steps to select a particular print style or function. Once you are in the SelecType selection mode, perform the following steps:

- Press the "OFF-LINE" button the number of times corresponding to the code in the Table above. Be sure the printer beeps each time you press the "OFF-LINE" button.
- Press the "FF" button to lock in the print style/function.
- Press the "LF" button to exit (terminate) the SelecType selection mode. The control panel is now available for the normal ON-LINE/OFF-LINE functions.

- Press the "OFF-LINE" button to put you ON-LINE. You are now ready to print your document/file.

In the steps above, I stated that the "FF" button either sets (locks in) or clears a print style or function. If a particular print style or function is currently set, the "FF" button pressed in the steps above will clear that function. All set functions can be cleared at one time by pressing the "FF" button before pressing the "OFF-LINE" button upon entering the SelecType selection mode. Another way to clear all functions is to turn your printer off, then back on again.

In the steps above for SelecType, only one style was selected. However, multiple styles can be used together. For example, Expanded and Emphasized. Perform the following steps:

- Put the printer into SelecType mode by pressing the "OFF-LINE" and "FF" buttons together as before. Make sure the printer is in SelecType mode before proceeding (the signs indicating this are stated above).
- Press the "OFF-LINE" button twice (Expanded Print has a code of 2).
- Press "FF" to lock in this request.
- Press the "OFF-LINE" button twice again (Emphasized Print has a code of 4; remember you already entered a code of 2, so all you need to enter is two more presses of the "OFF-LINE" button.
- Press "FF" to lock the second print style in.
- Press the "LF" button to exit the SelecType mode.
- Press the "OFF-LINE" button to put the printer on-line.

The user can now print using the Extended/Emphasized print style (in either DRAFT or NLQ).

In addition to the normal FX-80 print styles and modes, the FX-85 update adds the IBM Mode. This will allow the user to print the standard IBM-PC character set, as well as the IBM alternate character set. The IBM character sets are somewhat different from the standard Epson character sets in appearance, as well as with some of the function control sequences.

There is no way to select the IBM mode either from the printer control panel buttons, or via control sequences. It can only be selected via a DIP switch (inside the printer, under the DIP switch cover). DIP switch 1-4 in the FX-85 has taken on a new meaning. If this switch is on, the printer is in EPSON mode; if the switch is off, the printer is in IBM mode. This means that if you want to switch back and forth between the EPSON mode and the IBM mode, you have to change DIP switch 1-4 each time.

If the International DIP switches 1-6, 1-7, and 1-8 are all "on", then the normal IBM character set is used. However, if any of the International switches are "off", then the IBM alternate character set is used. The alternate character set can also be selected via a control sequence code: "ESC 6". Please note that the NLQ feature is NOT available in IBM mode. Also some of the print styles are not available in the IBM mode, such as Italic print, Elite print and Pica print. The international character sets are only available in the EPSON mode, however, both of the IBM character sets do contain some international characters (see the User's Manual for the various international characters that are available).

In addition to some of the print styles not being available in IBM mode, some of the many EPSON functions are also not available, such as Set Left/Right Margin, special line spacing functions, Tab functions, "dot positioning" functions, etc.

As was stated above, DIP switch 1-4 took on a new meaning: selecting either EPSON mode or IBM mode. In the FX-80, this switch was used to either have the data characters sent from the microcomputer go directly to the print buffer (switch off) or have the data characters stored first in the 2K input buffer, then sent to the print buffer (switch on). Since the old function of the switch was eliminated, the input buffer (now increased to 8K) takes on a somewhat different meaning/purpose. There is no option as before; all data sent from the microcomputer is first stored in this 8K input buffer and then sent to the print buffer for printing.

This speeds up the printing considerably. If a file is less than 8K, the entire file is sent to the printer input buffer very rapidly and then the system prompt appears on the CRT, ready to accept another command. The printer then prints the file while the user can use the microcomputer to perform another task. If a file is longer than 8K, the microcomputer keeps sending data as fast as the printer input buffer can accept it. When the microcomputer is finished sending the data for the entire file, the printer has about 8K of data yet to print. This frees up the microcomputer very quickly for other tasks at hand.

Files are not the only printing that is speeded up. When you send screen prints (both text only and graphics) or use the CTRL-P feature to print information that is being displayed on the CRT, the printing also is speeded up. Try a DIR on a disk file (press CTRL-P first) and see the difference in print speed and how fast your microcomputer is freed up to do other tasks.

The last feature added is for Cut-Sheet Feeders. There is very little information on this feature. DIP switch 2-2, which used to control the buzzer, now controls the optional Cut-Sheet Feeder. When the switch is "on", the Cut-Sheet Feeder is enabled. When it is "off", the Cut-Sheet Feeder is disabled. This switch is normally "off" unless you are using a Cut-Sheet Feeder.

The only other place that the Cut-Sheet Feeder feature is mentioned is in the function code section. Control sequence "ESC EM n" controls the Cut-Sheet Feeder. If n is 0, the feeder is turned "off"; when n is 4, the feeder is turned "on". That's all the information on Cut-Sheet Feeders.

Overall, I would say that the FX-85/185 User's Manual is quite good. However, the FX-80 Manual was better in some ways, such as giving programming examples for most control sequences. Together the two Manuals make a "near-excellent" reference set for the FX-85/185. Only the lack of good documentation on the Cut-Sheet Feeders prevents the two Manuals together from becoming an "excellent" reference set.

After using my "new" FX-85 for a few weeks now, I am extremely pleased with the new features and would recommend the upgrade to all FX-80 and FX-100 owners/users.

Happy printing..... *

Getting Started With Hierarchial Directories

Part 1

Eric L. Pang
1530 Nehoa Street
Honolulu, HI 96822

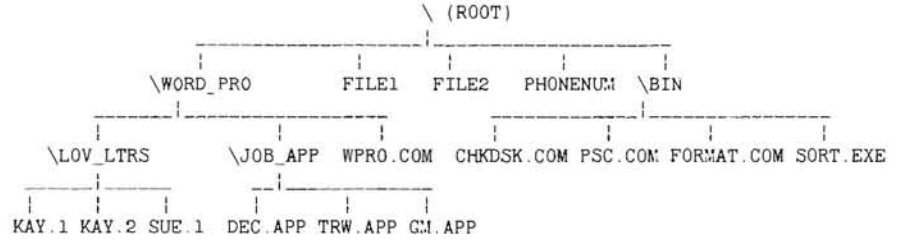


Figure 1
Example of a hierarchial directory structure, with associated files and subdirectories.

Introduction

Organizing the files on your disk is not unlike organizing any other items you want to keep track of. For example, you do not stuff all your bills into a paper bag, but instead, keep them grouped together by the company sending them, electricity, medical, department store, gasoline, etc. Within each group, the company's bill may even be separated out further, such as by month.

Similarly, the files contained on a floppy or harddisk can have enough in common between them to allow them to be separated into a number of distinct categories, example correspondence, spreadsheets, BASIC programs, games, utilities, etc.

The release of MS-DOS version 2 introduced the hierarchial directory structure to microcomputers. Instead of having all of your files lumped together in one directory, you now have the ability to organize files much more efficiently. You may also hear the term "tree-structure" in reference to this type of directory structure. It is an upside down tree, however, where the foundation or "root" is at the top and growth of the branches is downward. (**Note:** Hierarchial directories are not new, as they were already present on minicomputers for many years and CP/M did allow you to create the first layer of "branches.") MS-DOS allows entries for 112 files and/or subdirectories in the root directory on a double-sided, double-density floppy disk, and 480 entries for files and/or subdirectories in the root directory on a harddisk partition.

A hierarchial directory file structure, along with an example of how subdirectories and files can be organized, is presented graphically in Figure 1. In this figure, you will see a number of names with a "\" (backslash) preceeding them. These names represent a subdirectory entry. The lone backslash at the top of the "tree" is the top or "root" directory. \WORD_PRO is a subdirectory under the root directory, while \LOV_LTRS is a subdirectory under the subdirectory \WORD_PRO. At each level, we can have a mix of both subdirectories and files. FILE1 is a file under the root directory, and is at the same "level" as the subdirectory \WORD_PRO.

The "complete" file specification associated with any given file is of the form "\DIRECTORY\DIRECTORY\FILENAME". You can consider the complete file specification as being consisting of two parts, the path name, "\DIRECTORY\DIRECTORY", and the actual name of the file, "FILENAME". The path name points MS-DOS to the correct directory within the hierarchy to find the desired file, and the file name identifies the desired file within that directory. For example, the letter to Sue would have a complete file specification of

```
\WORD_PRO\LOV_LTRS\SUE.1
```

with a path name of "\WORD_PRO\LOV_LTRS" and a file name of "SUE.1". MS-DOS must have this complete file specification in order to access it, e.g.

```
TYPE  \WORD_PRO\LOV_LTRS\SUE.1
```

Commands will be introduced later that will allow you to use short cuts, saving you the trouble of typing the complete file specification. The point to remember from this discussion is that complete file specification for any given file consists of two parts, the path name and the file name.

Manipulating Directories

Several commands are available under MS-DOS that are used specifically to manipulate (e.g. create and remove) directories. When a disk is first formatted, the root directory, "\", is created. To create or "make" a new subdirectory under the current work-

ing directory, in this case, the root directory, the "MKDIR" command is used,

```
MKDIR WORD_PRO
MKDIR BIN
```

Subdirectories can be created at any time, the disk does not have to be empty. Of course, other files can be copied to the root directory too. Keeping Figure 1 in mind, a file directory of this disk is presented in Figure 2.

```
Volume in drive B has no label
Directory of  B:\

WORD_PRO     <DIR>      12-08-85   8:25p
FILE1           2585     4-04-84   1:50p
FILE2          24840    10-31-84   1:06p
PHONENUM        1945    10-31-84   1:17p
BIN          <DIR>      12-08-85   8:26p
       5 File(s)    243712 bytes free
```

Figure 2
List of files found under root directory.

As you can see, subdirectory "files" are distinguishable by the label "<DIR>". Don't go overboard creating subdirectories though, each subdirectory by itself will take up a little more than 1K of disk space.

Root, "\", is our current working directory. We now want to go into subdirectory \WORD__PRO and create our love letter and job application subdirectories. We can first move down the directory tree and make \WORD__PRO the current working directory. This change in directory is accomplished with the command, "CHDIR",

```
CHDIR  WORD_PRO
```

Notice there is no backslash preceeding "WORD__PRO". MS-DOS knows that WORD__PRO is a subdirectory, and that in this context, we want to move down one level of the directory hierarchy. The command "CHDIR \WORD__PRO" would also work in this case, but you should make the distinction between these two commands. The first command, CHDIR WORD__PRO, would only work if the subdirectory, here, WORD__PRO, is in the current working directory. In the second command, the complete name of the directory, \WORD__PRO, was specified and CHDIR \WORD__PRO would work anywhere within the directory structure. You might think of these two examples in terms of "relative" and "absolute" jumps, where the first example would mean go down one level from the current level (relative), while the second example would mean go specifically to the named directory (absolute).

Being in \WORD__PRO, MKDIR is used to create two more subdirectories. Having created the love letters and job application subdirectories, and making a copy of the word processor, the file directory of \WORD__PRO is shown in Figure 3. The "." entry represents the file name or "place holder" for the current working directory, \WORD__PRO. The ".." entry refers to the file name of the "parent" directory, in this case, root. If you see these dots in a file directory, you will know that you are in a subdirectory. Figure 4 is an example of what the contents of this directory might look like after several letters are typed.

If we wanted to copy the word processor to \WORD__PRO, we would "normally" have to specify the full file specification for the file.

```
COPY A:WPRO.COM  \WORD_PRO\WPRO.COM     or
COPY A:WPRO.COM  \WORD_PRO
```

(The first command means copy WPRO.COM on drive A: to the file named WPRO.COM in the subdirectory WORD__PRO on the default drive, while the second command means copy WPRO.COM on drive A: to the subdirectory \WORD__PRO on the default drive.)

Each time we use CHDIR to change directories though, we set the MS-DOS path name for the current working directory, here, \WORD__PRO. This means that we do not have to type the full file specification for the file each time we want to manipulate it. Since we already changed the directory to \WORD__PRO and MS-DOS remembers the path name \WORD__PRO, I could have just used the command

```
COPY  A:WPRO.COM B:
```

```
Volume in drive B has no label
Directory of  B:\word_pro

             <DIR>      12-08-85   8:25p
             <DIR>      12-08-85   8:25p
LOV_LTRS     <DIR>      12-08-85   8:26p
JOB_APP      <DIR>      12-08-85   8:26p
WPRO     COM    41216   3-31-84   9:18a
       5 File(s)    243712 bytes free
```

Figure 3
List of files found under subdirectory WORD__PRO.

If you forget what the current working directory is set to, you can type CHDIR and the working directory name will be displayed on the screen. You can set one working directory path name for each disk drive present on your system.

We can move back up to the previous directory, in this case the root directory, in two ways. One is to specifically state the directory's name in the CHDIR command, "CHDIR \". The other is to use a "relative" notation, "CHDIR ..", which would mean move up to the previous or parent directory. This notation can also be used in combination with directory names, e.g.

```
CHDIR   \  \BIN
```

which means move up two levels in the directory structure, then across and down one level to the subdirectory BIN. This can save some typing for long file specifications.

Although using CHDIR can help us manipulate files between subdirectories very easily, our applications (machine language programs or batch files) or "transient commands" do not make use of the path names set by CHDIR. These programs or transient commands follow other MS-DOS rules, those rules set forth by the PATH command.

The PATH command tells MS-DOS to search a designated path or subdirectory list for transient commands, if they are not found in the current working directory. You can tell MS-DOS where to look for transient commands by setting the path as follows,

```
PATH  \WORD_PRO
```

When a transient command is typed, e.g. WPRO, MS-DOS will first look in the current working directory and execute the program. If the program is not found, the subdirectory \WORD__PRO is then searched to see if the program is present there. If the program is not found there either, an error message will be displayed. You can check the current path setting by typing "PATH" with no arguments. The screen will display

```
PATH=\WORD_PRO
```

You can also have multiple paths, with each path name being separated by a semicolon,

```
PATH \WORD_PRO;\BIN;\
```

```
 Volume in drive B has no label
 Directory of  B:\word_pro\lov_ltrs

 .            <DIR>      12-08-85   8:26p
 ..           <DIR>      12-08-85   8:26p
 KAY      1       2176   6-28-85   12:23p
 KAY      2       1920   6-28-85   12:54p
 SUE      1       4864   9-22-85    3:36p
        5 File(s)     243712 bytes free
```

Figure 4
List of files found under subdirectory LOV_LTRS.

If MS-DOS does not find the transient command in the current working directory, it will first look in \WORD_PRO on the current drive. If the command is not found there, it will then search \BIN on the current drive. (\BIN stands for binary files, and contains all our utilities, such as CHKDSK.COM, FORMAT.COM, etc.) It will finally look in the root directory of the current drive if the command is not found after the first two tries. You can also specify a drive name in the PATH command, e.g.

```
PATH A:\BIN;A:\WORD_PRO;B:\BIN
```

Note: if the program you use requires overlays to run, you are in for a little let down. Although PATH can be used to find the command file, e.g. WPRO.COM, program overlays do not make use of it. In this situation, the only quick solution to this problem is to have copies of the overlay in each subdirectory which the program will be used.

The only command left which can be used to manipulate directories is "RMDIR". This command removes subdirectories no longer needed. The syntax for this command is simply "RMDIR directory_name". Before you remove a directory though, all files under that directory must be deleted first.

A list of the MS-DOS directory manipulation commands, along with a description of the command and command syntax, can be found in Table 1.

Command	Description	Syntax
MKDIR (or MD)	Make new subdirectory	MKDIR dir_name
CHDIR (or CD)	Change working directory	CHDIR dir_name
	Display working directory	CHDIR
PATH	Set MS-DOS command path.	PATH \path1;\pathx
	Show MS-DOS command path	PATH
RMDIR (or RD)	Remove subdirectory	RMDIR dir_name

Table 1
Commands for directory manipulation under MS-DOS. (The abbreviated commands within the parentheses can also be used.)

An example of a simple batch file (which resides in \BIN, of course) that can be used for word processing is presented in Figure 5. When "WP" is typed on the currently logged drive, the PATH is set to \BIN and to \WORD_PRO. This tells MS-DOS where to look for your programs. Next, \LOV_LTRS is made the current working directory. The word processor is then invoked automatically.

```
rem wp.bat
path = \bin;\word_pro
chdir word_pro\lov_ltrs
wpro
```

Figure 5
A simple batch file for word processing.

Summary

This article should have given you: 1) an idea of the hierarchial directory structure, 2) the commands used to create and maintain subdirectories under MS-DOS and 3) an example of how hierarchial directory structures can be used.

Subdirectories are a powerful operating system tool that can be very useful in keeping your files organized, e.g. having one subdirectory for word processing files, one for spreadsheets, one for games and another for miscellaneous utilities. Though the use of subdirectories might be considered almost a necessity on harddisks, there are many applications where a hierarchial directory structure can be put to good use on a floppy disk-based system.

Addendum

Perhaps MS-DOS could have been made smarter so program overlays could be found in the same manner which transient commands are found. On the other hand, application writers could have made their programs smarter by checking the path name set in the MS-DOS "environment" (the "environment" can be displayed by typing "SET").

In any case, two commercial products have been introduced. They contain programs that are extended versions of the current MS-DOS PATH command, allowing program overlays to be found in a similar way transient programs are found. One product is FilePath 3.0 by SDA Associates, P.O. Box 36152, San Jose, CA 95158 (408) 281-7747 at $37.50. The other is SuperPATH by Martin Scot Development Corporation, 4515 Purdue N.E., Seattle, WA 98105 (206) 527-9605 and is $39.95.

Although SuperPath costs a little more, the package does contain a number of other MS-DOS utilities, such as an extended file directory replacement and a rename directory command. Although it claims to be for H/Z-151-type computers (IBM PC to you non-Zenith readers), it will work with the H/Z-100s. The online help screens will not function though, as they use DOS interrupts to clear the screen. The help screens will display properly under ZPC.

. . . if you don't need this stuff, then forget it.

*

X-10® POWERHOUSE™

NUMBER ONE IN HOME CONTROL

Use the latest electronic technology to control lights and appliances throughout your home. It's now possible with our radio home control system.

Transmitter
Works from up to 150ft. away, even from outside your home!

Experience remote control of your home via radio control

Imagine turning on the lights in any room of your house from your bed or from your car as you enter your driveway. With the BC-5000 Radio Control System it's easy to put all your lights and electric appliances at your beck and call.

This dynamic system includes a Radio transmitter (BC-504) for transmitting radio frequency signals, and a Radio Transceiver/Appliance Module for receiving the signals to turn electrical devices on or off. Because it is a "Transceiver" it re-transmits, to standard X-10 Modules, over your house wiring, the signals it receives from the Radio Transmitter. **This lets you control up to 8 X-10 modules (sold separately) from the Radio Transmitter.**

Plus you can have control from any other X-10 controller, such as our Timer, Maxi Controller, or Home Control Interface.

BC-5000 Radio Remote Control system consisting of one Radio Transmitter and one Radio Transceiver/Appliance Module

Just $49.99 Complete

Enjoy Computer-Programmed Home Control

Now you can use the power of your computer to control all the lights and appliances in your home! **All without tying up your computer. The BC-290 Home Control Interface** is an RS-232 Controller which you program from your computer and *then disconnect,* freeing up your computer for other uses. Your interface sends signals over your house wiring to control lights and appliances plugged into the X-10 modules.

• Interface with menu-driven software for the IBM PC and compatibles. **BC-290P *Was* $59.99 *Now Only* $49.99** ***• Interface with icon-based software for the Apple Macintosh*** **BC-290M *Was* $69.99 *Now Only* $59.99** ***• Interface with color graphics software for the Apple IIe and IIc.*** **BC-290A *Was* $59.99 *Now Only* $49.99**

SPECIAL BUYS!

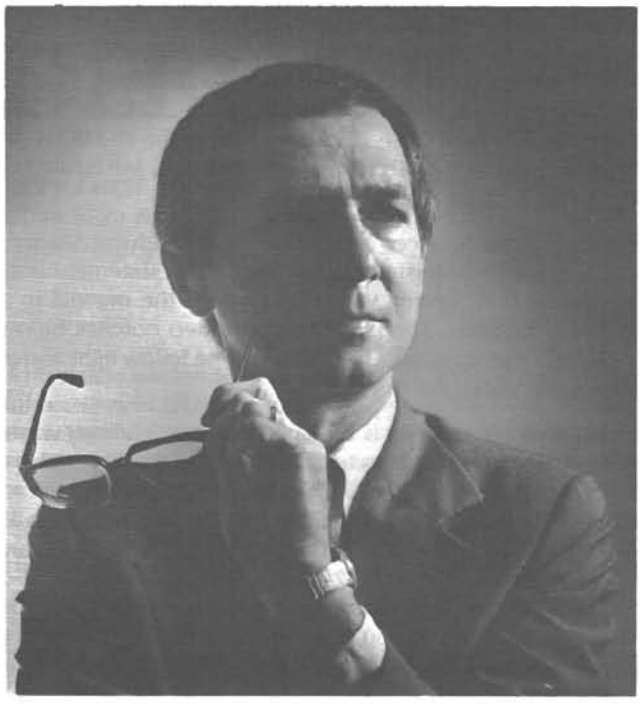
- **Purchase one BC-5000 RF Radio Remote Control System plus any 2 modules* and receive one Lamp Module FREE...A $14.99 Value! *You Pay Only $79.97!***
- **Purchase one computer interface, your choice, plus any 2 modules* and receive one Lamp Module FREE...A $14.99 Value! *You Pay Only $89.97* (Apple® IIe/IIc and IBM) or $99.97 (Macintosh™).**
- **Purchase one Maxi Controller plus any 2 modules* and receive one Lamp Module FREE...A $14.99 Value! *You Pay Only $54.97!***

***Choice of either Lamp Module, Wall Switch, or 3-PIN Appliance Module.**

Prices good through December 31, 1986

Available at your nearest Heath/Zenith Computers & Electronics Store

See pages 28 & 29 of our current Heath/Zenith catalog for more details on our exciting line of X-10 POWERHOUSE products.



Mainstream Computing

Joseph Katz
103 South Edisto Avenue
Columbia, SC 29205

One good thing about the heat wave here this summer was it gave me time to explore new hardware on my H-158. By "new hardware" I mean memory boards, hard disk drives, hard disk controllers, and that sort of thing — the sort of thing that requires endless sessions of mindnumbing fiddling to explore. Back up the hard drive, pull the boards, pull the floppy diskette drives, put in the new hard drive, replace the floppy diskette drives, replace the boards, prep the new drive, partition it, format the partitions, restore the files, and watch the bad sectors develop when something goes wrong. That's the drill over and over and over again. It takes time to do and the results take time to establish. Well, it was more intolerable outside than it was inside and I suppose I needed the exercise.

What came out of all this activity was enough of value to repay my time and work. My H-158 now has one external floppy diskette drive in addition to the standard internal floppy diskette drive. It has two internal hard disk drives, one with a fixed disk and one with removable hard disk cartridges. It has a hard disk controller that allows me to change the mix of drives without much pain when I find drives I like better. It has 1.6 MB of RAM that I can expand inexpensively. And it's faster than the average *XT* compatible, turning in a Performance Index of 2.6. Interesting?

I should say that while some of what I've called "new hardware" really is "new," some of it might better be called "recent" and other things might more appropriately be called "new to me." The last two categories include things that have been sitting here waiting until I had the time to give them more than cursory attention. We don't do that in this column.

What's curious is that none of the parts I've added or replaced would deserve a feature story by itself in most computer magazines. The advances are evolutionary, incremental, and undramatic. Neither are they so pricey as to catch anyone's attention. Most things actually are less expensive than any of their competition, but price alone was not my reason for choosing them. Perhaps it's too early to say that these are best of breed. Rarely do I think that way anymore, except in the first burst of enthusiasm.

Since I'm still under enthusiasm's spell, though, I will confess to thinking these things among the best in their class right now. Each product individually strikes me as very good indeed. Put them together as I did, moreover, and the result is serendipity: the whole is greater than the sum of its parts. These things work together very nicely. Best of all, I came up with the plan and executed it myself. In fact, one reason why my machine deserves attention is that the modification was done with no outside help. Perhaps I'm getting better; perhaps the industry is. My hunch is that both are true.

The H-158 For Learning

The basic machine began life as a stock *XT*-compatible Heath H-158 — the kit version of the Zenith Z-158. Here's one powerful reason why I really think everyone interested in microcomputers ought to build at least one of them from a kit: you get a sense of how the thing works together as a machine. It stops being a magic box and resolves into a series of elements fit together according to a scheme devised by people and comprehensible to people. As the kit takes shape, therefore, the construction implies a conceptual framework for thinking about microcomputers as rational devices. Ancients and wizards sneer at the H-158 and other recent Heathkits as "screwdriver kits." Almost anyone with patience can put one together in relatively little time. In short, they're no challenge. And that's just what I like about these Heathkit computers: because there's no impossible barrier from mechanical skills I lack, these new kits offer a way for me to benefit from intellectual skills I own. That way I learn as I go. What I learn is neither soldering nor hardware debugging: I learn concepts essential to understanding how the darned thing works. With those concepts I can better understand what I need to make my computers work better for me.

It took six and one-half hours to put together my H-158, my very first kit of any kind (if you don't count some bicycles and kid's toys). Would I do it again? I did. I subsequently built an H-241, in one and one-half hours. What I've learned from the kit building has proven so valuable to me that I welcome new opportunities of the kind. No, I'm not an "old hand" now and never will be. What advantages I have over someone who has never done it before, with these new kits, are based on overcoming the apprehension with which I faced my first box of parts.

Big Storage

I'm certainly not a hard disk or controller maven and until now I thought there were few occupations more boring than to hear talk about their details. If you're that way too, prepare to be bored. I've just turned a 10 MB hard disk into a 15 MB hard disk and a 20 MB hard disk into a 30 MB hard disk. Then my eyeballs clicked to attention.

The key is Adaptec's *ACB-2070A* hard disk controller for *XT*-compatibles. (It can't be used on *AT*-compatibles like the Z-241, but Adaptec is supposed to be working on something for those machines.) Oddly enough, I was first interested in the *ACB-2070A* for another reason entirely: it's among the first controllers to allow mixing different drive types without requiring a special ROM (Read-Only Memory) for each. I had a SyQuest *SQ312RD* drive, the kind that uses removable 10 MB hard disk cartridges, in the Z-151. I originally thought it would go into my 158 to supplement the 20 MB fixed hard disk installed there, but I never could find the time to do the necessary. The *ACB-2070A* will control two drives and has an EPROM (Eraseable Programmable Read-Only Memory) with tables for many of the major types of drives used in *XT*-compatibles. By changing jumpers and using an internal prep program accessible with *DEBUG*, you can get it to handle still other kinds of drives. That's why I was first interested in the *ACB-2070A*.

Simultaneous with the arrival of this controller came a 20 MB Seagate *ST238* fixed hard disk. It's a new drive that Seagate said is "based on ST225 technology which has been modified to work with Seagate approved RLL 2,7 controllers." Since the Adaptec manual said the *ACB-2070A* was a 2,7 RLL controller, I decided to swap the 20 MB Seagate *ST225* then in the 158 for the SyQuest in the 152, pair the new *ST238* with the SqQuest and the *ACB-2070A*, and give this mixed bag a shake to see what happened.

What happened eventually was that my 10 MB SyQuest cartridges turned into 15 MB cartridges and my 20 MB Seagate *ST238* became a 30 MB hard disk. Ah, life has many precious moments. One of them was when I realized the advantages of 2,7 RLL encoding.

My previous controllers used the MFM (Modified Frequency Modulation) encoding system for laying down data tracks on the disk. MFM is the standard system with hard disks used on microcomputers. Unless you bought a hard disk in the past few months and you, your dealer, or your consultant knew then of 2,7 RLL encoding, the likelihood is that your present hard disk drives also use MFM encoding.

2,7 RLL (Run Length Limited) encoding is a new development. It packs 50% more data into the same space as MFM encoding. It also allows faster data access, because it uses a data transfer rate of 5 MHz (instead of the *XT*-standard 7.5 MHz) and an interleave factor of 3 (instead of the *XT*-standard interleave factor of 6). Hard disk drives that are low-level formatted (the kind of formatting done by Zenith's *PREP* and IBM's *FDISK*) with the 2,7 RLL system perform more effectively than with the MFM system.

Not every hard disk can tolerate the more strenuous demands of 2,7 RLL encoding, however. "Based on Adaptec's experience in testing drives," says their manual, "many drives that use thin film-plated and sputtered media meet these requirements. Most drives that use oxide media do not." Fortunately, the *ST238* and the *SQ200* cartridges for my SyQuest both meet those requirements. So the *ST238*, which starts out unformatted as a 38 MB drive, formats to 30 MB under 2,7 RLL instead of the 20 MB I'd get under MFM, and the *SQ200* cartridges now hold 15 MB instead of 10 MB: a 50% increase in space.

Installation is a breeze once you have fought through the Adaptec manual. You won't understand the hilarity in that statement until you do succeed in fighting your way through the manual to a successful installation. Then you'll put me two notches below Mark Twain in the humorist's hall of fame. The fellow right above me is whoever wrote the manual.

Most necessary information is there, I guess, but whoever wrote the darned thing seems always to have forgotten what he had just written. There's no coherence, no sense that each detail fits an overall plan. For example, on p. 2-5 the *ST238* is referred to as a 25 MB drive while on p. 2-6 it's a 20 MB drive. On p. 2-5 you are directed to Table 3 of Table 2-2, then a comment to Table 2-2 directs you to a Table 2-3 for jumper selection but since the *ST238* is in Table 0 of Table 2-2 you don't know whether to use the 0 jumpering in Table 2-3 for Table for Drive 0 or the 3 jumpering in Table 2-3 for Table for Drive 0. These instructions could melt bricks. I spent a long day and part of a night moving from step to step, following blind alleys, backtracking, starting again, and eventually working my way through the maze. At least I think I did. I'm still not sure. But the drives have been working, so I guess I was right.

You, therefore, might be interested in a brief summary of what turned out to work for me. If you have different drives you might need to change some of the following details, but at least you'll have something by way of a road map towards something like the right general direction.

The Adaptec has tables for four drive types built into the controller ROMs. Fortunately, my drives were among them. Unfortunately, that grace seemed not to matter for the SyQuest. Fortunately, it didn't matter once I figured out what was going on. The first thing to do is jumper the board to select the number of drives and their types. There are two sets of jumper banks: those labelled "M-T" are used for standard fixed drives; those labelled "A-L" are for removable drives. It turned out that my Seagate required either no jumpers on the "M-T" or jumpers on "M-N" and "O-P." I tried both ways and it worked the same both ways. The SyQuest required a jumper on "G-H." Next I daisy-chained the 34-pin cable from J2 on the Adaptec to the Seagate and back to the SyQuest, used a 20-pin cable to connect the Seagate to connector J0 (for Drive 0, first in the sequence) on the controller and used the other 20-pin cable to connect the SyQuest to connector J1 (for Drive 1) on the controller. Then I put the controller into a vacant slot in the backplane board, drew power from a "Y" tap in plug P2 to each of the hard disk drives, reassembled the computer, and booted from a floppy disk in Drive A.

With the hardware fiddling behind me, I turned to fiddling with software to do the three stages required to initialize the hard disks: prepare, partition, and format.

Forget Zenith's low-level formatter, *PREP*: it gets confused by 2,7 RLL encoding. Instead use the low-level formatter built into the

controller itself and accessed through *DEBUG*: *g=c800:ccc* is the command needed to start things going. You get Adaptec's banner and the instructions to enter all information in decimal, then face a series of questions. Here are the parameters that worked for the Seagate: interleave factor is 3; drive id is 0 (to make it the first drive of the two); no, don't use the defaults for that drive type; logical units is 1; step rate is 3 (not an absolute value but a code); head count is 4; number of cylinders is 615; use MFM encoding; enter defects in the format "cylinder/head/sector"). Then you are supposed to specify any defects from the list supplied by the manufacturer. There were none on my drive, so I pressed RETURN and the program did its work. My later experiences suggest that you may not need to specify defects even when there are any on the drive, because the Adaptec program seems to figure things out as it goes and makes all needed corrections. Its way of compensating for drive defects, by the way, is delightful to watch.

For each SyQuest cartridge drive I used the same answers except as follows: interleave factor is 4; drive id is 1 (to make it the second drive of the two); head count is 2. My four SyQuest cartridges came to me in a variety of conditions ranging from one with no defects to one with 23: the Adaptec controller formatted them all satisfactorily, writing alternate tracks when necessary to squeeze maximum space from each cartridge. A relatively slow process, because the program seems to test and retest, it's still an order of magnitude faster than Zenith's *PREP*. Figure on 45 minutes for my Seagate and 15 minutes for each of my SyQuest cartridges. Then I used Zenith's *PART* the usual way to partition the Seagate into Drive C and D, and Zenith's *FORMAT* to format each drive, putting a system on Drive C so I could boot from it. Remember you must use *CONFIGUR* to set the system for manual partitioning and then use *ASGNPART* to assign partition 1:1 to Drive D before you can format it. At last I partitioned and formatted each cartridge drive. Over and over and over again, until I figured out what the manual was trying to say. I want a raise, a big one.

There are some eccentricities with my system.

One is that Zenith's *Z100-PC Disk Based Diagnostics* can't make up its mind about whether or not I have any hard drives at all. In some tests it can't communicate with the controller; in other tests it evidently does. Never does it believe there's a bad sector table on any of the drives. So forget using the Winchester diagnostics with 2,7 RLL encoded drives: the diagnostics never heard of the new system.

The second eccentricity — or maybe it's just the way these things work — is that I have to use *ASGNPART* to reassign the SyQuest each time I change removable drives. Otherwise DOS retains an image of the previous drive's directories. I had the SyQuest as the only drive in my 152 for a year or so, and I could change cartridges without fiddling. Then, however, I was using a Xebec controller. The change of controllers seems to have something to do with this eccentric behavior. Or maybe I just don't understand something obvious?

The third eccentricity really is my own fault: since some cartridges are partitioned into one drive while others are partitioned into two, I had to create two different batch files to run *ASGNPART* for setting them up. I can live with all these deviations from what I had been accustomed to. It's all custom and artifice anyway.

Because it is, by the way, I can't use software protected with copy-protection schemes like Softgard's SoftLok, which won't install to hard disk drives E or F. I've, therefore, had to abort my reviews of programs using those schemes because I don't feel the obligation to reorganize my entire computer system so I can continue. Pity. There were some things I'll miss pursuing, like Computer Support Corporation's *Diagraph*. What my system illustrates, however, is that copy-protection really does place extraordinary limitations on the usability of software. Pass it by, folks. Pass it by, or recognize that you use your system at the pleasure of a software manufacturer concerned exclusively with his own interests and not at all with yours.

Thanks For The Memory

It's hard to believe that 640 KB of RAM seems too small for me now. Too small it is though, in part because I work with big programs, big files, and as many RAM-resident programs (like *SideKick*) as I can pack into memory. So I want all the memory I can get, even if whatever goes above 640 KB of RAM needs planning to use. I've put in Boca Research's *BOCARAM*, which gives me a full 1 MB to play with above the 640 KB on the standard memory board in the Z-158. To understand why I selected a *BOCARAM* (of which you've probably never heard before) instead of some better-known memory board, you'll have to follow me the long way around. You need to remember that I have an *AT*-compatible Z-241 waiting in the wings and, therefore, tend to look for things that can be used with *AT*s, as well as *XT*s.

Fairly high on the list of things I don't understand is why the "enhanced" *AT* and its most faithful compatibles, like the Zenith Z-200 series, allow only 512 KB of RAM on the system board. Of course, an even higher place on my list of incomprehensibles is why the original *AT* came with only 256 KB, but that puzzle really is none of my business. Mine, and yours too eventually, is why in the world you're required to buy an additional board to get 640 KB of RAM on a Z-200. I don't know.

It's a real puzzle in a hot machine that derives much of its caloric potential from the ability of the 80286 microprocessor to work with up to 16 MB of system RAM. My puzzlement is increased because really good *XT*-compatibles, like Zenith's Z-158, now allow you to put 640 KB of RAM in the machine by just plugging in additional banks of nine RAM chips. In contrast, you need a supplementary board to hold the two banks of 64 KB chips it takes to "top off" the base RAM in a true *AT*-compatible. I don't know a technical reason why, and neither did Pat Swayne or other engineers I've asked. Marketing? A way to make a few more bucks? The passion for compatibility unchecked by good sense? Who knows.

Odd problems usually provoke equally odd solutions at first. The immediate approach to solving this particular odd problem is exemplified by Zenith's Z-405 extended memory board. (We'll get to the meaning of "extended memory" in a few seconds.) Although it has a total RAM capacity of 1.5 MB, what you find on the board right out of the box is only the 128 KB needed to top off the base RAM. At $399 for the Z-405, that's a pretty expensive bunch of bytes if all you want is a memory topper.

In fact, an *AT*-compatible might benefit from nothing more than a topper right now, because right now there's no sure path to gaining real advantages from the machine's potentially great memory. The extended memory scheme, earliest of the three major schemes we have now for addressing memory above 640 KB of base RAM, seems to offer the possibility for little more than a giant RAMDISK or print spooler. That's why the Expanded Memory Specification (abbreviated "EMS" and also known as *LIM* because it is sponsored by Lotus, Intel, and Microsoft) is more likely one of the two other paths to pursue right now. Unfortunately, it's still not well travelled by software manufacturers. I know only a couple dozen

packages that use Expanded Memory. They include applications software such as *Ready!*, version 2.0 of *ThinkTank*, version 2.0 of *1-2-3*, version 2.1 of *SuperCalc 3*, *Framework II*, and *Javelin*, and environments such as *DesqView* and *Windows*. The third major path is the Enhanced Expanded Memory Specification ("EEMS," but not "AQA" although it was developed by AST Research, Quadram, and Ashton-Tate). EEMS has most possibilities of the three, but right now I don't know of anything written especially for it. Fortunately, EEMS will support that EMS software too. But if you don't need or use those specific packages, the only significant thing you can do with Expanded Memory or Enhanced Expanded Memory is what you can do with the lowly extended memory: run a RAMDISK or print spooler.

So you have to think of the 640 KB of base RAM as the dividing line it really is. You ought to cross that line only if the trip will pay off in your situation. In that case, the immediate way to go might well be to start stuffing banks of 256 KB RAM chips onto an Expanded Memory board like the Z-405. But it's an extravagant way to go if all you want is a topper that brings you to the 640 KB boundary recognized by most standard applications packages for MS-DOS right now. Until now, though, you had no choice: a board like the Z-405 was the only choice you had. Its price and approach define the first generation solution to this odd problem.

Now a relatively new company, Boca Research, has defined the second generation solution with its *TophAT* and *BOCARAM* boards. The idea behind these two boards is simple, sensible, and nicely cost-effective. *TophAT* is a memory topper for *AT*-compatibles. This board comes in two versions. Z-200 owners need the less expensive version, which comes with the 128 KB of RAM required to fill the system RAM to 640 KB. The 128 KB version of *TophAT* costs $145: $254 less than the Z-405. (For the record, there's a 384 KB *TophAT* at $195 for owners of compatibles like the original IBM *AT*, which have only 256 KB of system RAM, but Z-200 owners can ignore it.) For owners of *AT* compatibles, *BOCARAM* starts where *TophAT* leaves off, adding Expanded Memory without overlapping the base RAM.

Here's where the approach taken by Boca Research pays off for us. By separating the two functions of that first generation of memory boards, Boca Research makes *BOCARAM* valuable to owners of almost any compatible — *XT* or *AT*. My Z-241 is dead again, so I pulled the *BOCARAM* out of it and put it in my *XT*-compatible H-158. It's awfully nice for a writer to have 1 MB of Expanded Memory in addition to the 640 KB of base memory. At this very moment I have 64 KB of the Expanded Memory available as a print spooler, an outliner (*Ready!*) running in 160 KB more of the Expanded Memory, and *NewWord 3*, its overlays and dictionaries, and this column on a 700+ KB RAM disk. *Ready!* uses 3 KB of the base RAM, leaving all the rest of that memory for *MS-DOS* and *NewWord 3* to run. With all that going on, I still have about 270 KB of base RAM available. Everything runs really fast, nothing feels the least bit cramped. The most serious problem on my mind is what to do when the Z-241 returns. I guess by then I'll have to get another Expanded Memory board for it. It may well be a *BOCARAM*, but it doesn't have to be: Boca Research's *TophAT* liberates the computer it's on. You can use any Extended or Expanded Memory board with it.

There are five possible configurations of *BOCARAM*: a basic board with 256 KB of RAM costs $245 while one with 1 MB of RAM costs $395; a 1 MB board that piggybacks onto the basic board, giving a maximum total of 2 MB, costs $345; a 2 MB board (really a 1 MB basic board and a 1 MB piggyback) costs $745; and an unpopulated basic board costs $195. (Chips are soldered, not socketed, on the piggyback, so it is not available unpopulated.) You may add up to four of the basic/piggyback combinations for a total of 8 MB of Expanded Memory. The populated boards come with 150 NS dynamic RAMs, good enough for the IBM *AT*. At least the early versions of the Z-241 run with no wait states, so if you have one of those, you might want to buy unpopulated *BOCARAM* basic boards and use faster 120 NS RAMs instead. Or give Boca Research a call and ask about the availability of basic and piggyback boards already populated with 120 NS chips: I asked and they said they would think about providing such. The entire approach of Boca Research allows you to move in logical, economical stages defined by what you really need. It's a nice approach.

Installation of the *TophAT* is equally nice because there are no decisions to make: just slip the board into any of the vacant 16-bit bus slots on the Z-200, run the *Setup* program to configure the machine for 640 KB, and boot the computer. That's it. The only potentially-confusing part of the installation is that you don't remove a metal blank at the rear of the computer as you do when installing most other boards: take out the retaining screw but leave in the blank, then install the *TophAT* so its retainer covers the top of the blank, and finish by replacing the screw through the retainer and the blank so you secure both in place. It's the only part of installation that the *TophAT* manual doesn't make clear.

You'll need to do just a little more to install a *BOCARAM*, as you must with any Expanded Memory board, because there are more factors to the installation. It's not really complicated, though, if you stay with *BOCARAM* boards. With any of these memory boards you need to make sure they are jumpered for synchronous memory addressing. Follow the instructions in the *BOCARAM* manual and you shouldn't go wrong. The basic board goes into any vacant slot, but this time you do remove the metal blank associated with it. Then you edit the CONFIG.SYS file to add to your boot sequence any of the *BOCARAM* device drivers you want to use. The *BOCARAM* comes with a disk containing drivers that initialize the Expanded Memory (BREMM.SYS), provide a RAMDISK from it (BRDISK.SYS), and use it as a print spooler (BRSPOOL.SYS). You must use BREMM.SYS; you can use either or both of the other drivers if you want what they do. Then you reboot the computer and start working. *BOCARAM* comes with some good, easy-to-use software. In addition to the three drivers, there is a program to test the board's installation, a program to install or deinstall RAMDISKs in the Expanded Memory, and a program to install or deinstall a print spooler in the Expanded Memory. My experience is that hardware makers often supply clunky software, but this stuff is pretty good. It passed Katz's First Test: the test program worked without my reading the manual, and the other two programs delivered help messages good enough to get the RAMDISK and print spooler working when I was in a similar state of ignorance. The hardest part was taking off and replacing the computers' covers. I was up and writing within about ten minutes of opening the boards' boxes.

I like these boards. I confess, though, that their low prices gave me pause at first. Remember that snide remark, "If you're so smart why ain't you rich?" The parallel seems to be, "If they're so inexpensive why not call them "cheap?" Well, they're "inexpensive" but not "cheap." These two boards reflect an evident concern for high quality. My thinking moved next to an equally deplorable concern with the quality of the company. After all, who has ever heard of Boca Research? It's certainly not a name like AST or Quadram. Well, it turns out that although Boca Research is indeed relatively new (launched in January 1985), its credentials are excellent: Tim Farris, President of Boca Research, founded Quadram. Boca Re-

"Have Your Computer Talk To Ours!"

"HUGPBBS is on line, 24 hours a day, with over 10 megabytes of free software available for downloading. There's software for every Heath/Zenith operating system, with the majority being for MSDOS, and specifically the Heath/Zenith PC compatible computer systems. Also included is software for HDOS, CP/M, and MSDOS for the H/Z-100 computer system. In addition to this software is a message base through which you can exchange information with other HUG members. Have your computer call (616) 982-3956, 'The Heath Users' Group Personal Bulletin Board System', and make connection at 300, 1200, or 2400 baud. Type a carriage return several times to get my attention. Registration requires that you supply your human's first name, last name, HUG ID number, and some sort of secret password (up to 16 characters). Alternatively, your human can call Jim Buszkiewicz at HUG, and register via voice connection at (616) 982-3837. Call today! All it takes is a computer, modem, and a phone call for your computer to talk to ours!"

search's avowed plan is to produce innovative, inexpensive memory and graphics boards for compatibles. These first two products show they're off to a splendid start.

There are two things you should know, though. One is that I don't have a piggyback to go on the basic *BOCARAM* and so I haven't tried the combination. I therefore don't know whether there's enough clearance between boards on either the Z-158 or the Z-241. If I get a piggyback, I'll let you know the results. The second thing is that *BOCARAM* supports EMS, not EEMS. It really doesn't matter now, but might in the future. I just don't know for sure, and I don't know anyone who does.

From time to time I'll be talking about the rest of what I've done with the 158. Stay tuned. In the meantime . . .

Amnesia?

You may remember my talking about *FastBack*, the pricey fast-backup utility program from Fifth Generation Systems. (See "Mainstream Computing" in the July 1986, *REMark*) My reactions were mixed. My experiences with *FastBack* since I wrote that column are even more mixed.

One thing I mistrusted about *FastBack* was its notorious "Greetings" that informed you it had diagnosed a bad 8237 DMA controller chip and suggested you buy a "DMA-FIX" board from Fifth Generation Systems for $40. I became interested because there was a great flap — in fact a real scare — among people on CompuServe's HUG SIG who had received the Greetings and the sales pitch. Many of them apparently bought the "DMA-FIX." Pity.

I was working with an IBM *XT* owner whose system *FastBack* diagnosed as having a bad DMA controller after I wrote that column. Darned if it didn't seem true, too, because just before the installation of *FastBack* there had been intermittent data-recording errors — just the kind of thing *FastBack*'s Greetings suggests might happen. That kind of thing is scary. But my hunch is that maybe *FastBack* is trying to scare up a little side money. One thing that puzzles me, for example, is that there seems nothing in this disk-intensive program to diagnose a flakey floppy diskette drive, no error message that says something like "Wobble, wobble, wobble" or "Spin it faster, Jack." To check my hunch we fixed the floppy diskette drive on the *XT* and the "bad DMA" error went away, just as I thought it might. There have been no data-recording errors since then. Bad DMA my foot.

My H-158 received *FastBack*'s Greetings at about the same time. Remember that I've been backing up my hard disks frequently, sometimes twice a day, because of the drive-swapping I've been doing this summer. The Greetings came just once. I nearly fell off my chair laughing when I saw the message: the 158 does not have, and never did have, an 8237 DMA controller chip. It isn't built that way. *FastBack*'s Greetings say "This is the only diagnostic we know which will find the problem." Hmm. That's some diagnostic. It probably sells a lot of "DMA-FIX" boards.

Products Discussed

Heathkit *HS-158-1* Microcomputer Kit $1,399.00

Heath Company
Benton Harbor, MI 49022
(800) 253-0570

Adaptec *ACB-2070A* Hard Disk Controller $ 154.00

Adaptec, Inc.
580 Cottonwood Drive
Milpitas, CA 95035
(408) 946-8600

Seagate *ST238* Hard Disk Drive $ 995.00

920 Disc Drive
Scotts Valley, CA 90566
(408) 438-6550

SyQuest *SQ312RD* Cartridge Hard Disk Drive Price N/A
SyQuest *SQ200* 10 MB Hard Disk Cartridge $ 119.00

47923 Warm Springs Boulevard
Fremont, CA 94539
(415) 490-7511

TophAT Base RAM Topper (256 KB) For H/Z-241 $ 145.00
BOCARAM Expanded Memory Management Bds:
Unpopulated Board $ 195.00
256 KB board $ 245.00
1 MB Board $ 395.00
1 MB Piggyback $ 345.00
2 MB Combination Board And Piggyback $ 745.00

Boca Research, Inc.
6401 Congress Avenue
Boca Raton, FL 33431
(305) 995-6227

"Mainstream Computing"

Joseph Katz
REMark, Volume 7, Issue 9, September 1986

Products Discussed

Timeslips (Version 2.22) $99.00
North Edge Software Corp.
P.O. Box 286
Hamilton, MA 01936
(617) 468-7358

Mirror (Version 3.6) $49.95
SoftKlone Distributing Corp.
1210 East Park Avenue
Tallahassee, FL 32301
(904) 878-8564

DOG (Version 1.01) $20.00
(Shareware program, fee unknown)
George Allen Morris, III
1411 10th Avenue
Oakland, CA 94606

Disk Optimizer (Version 1.6) $49.95
SoftLogic Solutions
530 Chestnut Street
Manchester, NH 03101
(800) 272-9900

Monografx (Version 3.2) $99.95
M.A.C.E. (Version 1.10) $30.00
Analytix International, Inc.
1365 Massachusetts Avenue
Arlington, MA 02174
(617) 641-0400

ZPLOTTER.BAS

Kenneth R. Van Wyk
68 New Garden Avenue
Lancaster, PA 17603

In studying mechanical engineering, I frequently am faced with an answer in the form of a mathematical function. Quite often the function represents an approximation to a real world occurrence and is rather long and difficult to visualize. One answer, here at Lehigh, is to run down to the computer center and write a FORTRAN program which calls our handy plotter. This approach, while still a good one, can be somewhat tedious after just a couple of times. And besides, sometimes I don't need a neat, ultra accurate plot, I just want a quick picture of it, so that I can better understand the answer which I just calculated, and of course, to see whether it is feasible or not.

When I bought my H-100, I planned to write a short, simple program which would be the answer to this problem. My old H-89 certainly wasn't particularly good at plotting a function on the screen even though it is an excellent computer, in its own right. So, to make a long story short, here is my plotter program written in ZBASIC. I could've opted for FORTRAN which would be a bit quicker or even assembly language, but that would be defeating the purpose of this exercise. I wanted to be able to quickly run a plotter and get a pretty good rough idea of what the function actually looks like, and I wanted to be able to possibly change the function easily. Well, the ZBASIC full screen editor is perfect for this. While it may not be the fastest language, it suited my needs perfectly for this program. I didn't even compile the program because that, too, would make changing the function quickly rather tedious.

The program itself is rather short. It does what it was meant to do (and quite well), but nothing else. It will even plot a rather difficult function with surprising accuracy. It will plot up to about 15000 data points under the interpreter with 192k RAM, which is more than enough to get a rather good picture of the function.

The program is in two parts, a routine to load an array full of values along the function, and the plotter itself. Try running it as is and then try putting different functions in. The function is found on line number 30. Lines 80 through 140 load the array full of values. Line 150 calls the plotter. The first few lines of the plotter draw and label the axes, and finally lines 52500 through 53200 draw the function on the screen. The program then waits for any input from the keyboard upon which it clears the screen and lists the function part of the program, thus making it even easier to change the function.

I've found the program to be perfect for my needs. It automatically sets the upper and lower extremes of the function being plotted. If these extremes exceed the PRINT USING format size, an overflow will show, but the function will still be plotted. I think this program could also be used as an educational tool for students learning about functions because of its ease of operation.

If anybody has any questions or suggestions about the program, please let me know.

One final note to all of those who work at HUG or contribute to REMark. Keep up the good work! I've been a member of HUG for a couple of years now and I always get worthwhile information out of the magazine. My thanks to everyone who makes it worthwhile!

ZBASIC Listing

```
10 CLS
20 N=400:DIM Y(N)          'N is the number of points
30 DEF FNY(X)=SIN(X)       'The function to be plotted
40 YMIN=0:YMAX=0           'Initialize all minimum and
50 XMIN=-4*3.14159         'maximum limits  All of which
60 XMAX=4*3.14159          'can be manually altered
70 J=0:K=0                 'Initialize counters
80 FOR Z=XMIN TO XMAX STEP (XMAX-XMIN)/N 'Loop to fill
90     J=J+1:K=K+1                       'array with values
                                         'to be plotted
100    IF J>N THEN 140
110    Y(J)=FNY(Z)
120    IF Y(J)>YMAX THEN YMAX=Y(J)
130    IF Y(J)<YMIN THEN YMIN=Y(J)
135    IF K=10 THEN LOCATE 13,36:PRINT J;"/";N;:K=0
140 NEXT Z
150 BEEP:GOSUB 50000       'Array full, now call plotter.
160 END
50000 'Subroutine to plot an arbitrary function
50100 ' The following values must be set as follows
50200 ' XMAX, YMAX, XMIN, YMIN max and min values for
      each axis.
50300 ' Array to be plotted Y with N values along X axis.
50400 '
50500 '        Written by Kenneth R. Van Wyk
50600 '
50700 '
50800 IF YMAX=YMIN THEN YMAX=.5:YMIN=-.5  'Just in case
50900 CLS:LINE(40,5)-(40,215),5           'array is empty
51000 LINE(40,215)-(630,215),5
51100 COLOR 6
51200 LOCATE 13,1:PRINT "Y",       'Draw axes and max/min
51300 LOCATE 25,42:PRINT "X",      'values.
51400 LOCATE 1,1:PRINT USING "###.#";YMAX;
51500 LOCATE 24,1:PRINT USING "###.#";YMIN;
51600 LOCATE 25,4:PRINT USING "####.#";XMIN;
51700 LOCATE 25,75:PRINT USING "####.#";XMAX,
51800 LINE(36,47)-(39,47):LINE(36,89)-(39,89),
      LINE(36,131)-(39,131)
51900 LINE(36,173)-(39,173):LINE(158,216)-(158,218):
      LINE(276,216)-(276,218)
52000 LINE(394,216)-(394,218):LINE(512,216)-(512,218)
52100 COLOR 7
52200 YDELT=YMAX-YMIN     'Initialize some values used by
52300 XSTEP=580/N         'plotter and a counter
52400 ICOUNT=1
52500 Y1=INT(215-(((Y(ICOUNT)-YMIN)/YDELT)*210)) 'X-Y point
52600 FOR I=40 TO 620 STEP XSTEP                 'to screen
                                                 'position
                                                 'algorhythm
52700  ICOUNT=ICOUNT+1 IF ICOUNT>N THEN 53200
52800  Y2=INT(215-(((Y(ICOUNT)-YMIN)/YDELT)*210))
52900  IF INT(I)=INT(I+XSTEP) AND Y1=Y2 THEN 53200
53000  LINE(INT(I),Y1)-(INT(I+XSTEP),Y2)
53100  Y1=Y2
53200 NEXT I
53300 A$=INPUT$(1):CLS:LIST -49999
53400 RETURN
```

HUG Members only!

HUGE HUGGER DISCOUNTS now available on these selected products...exclusively for members only!

That's right. We're offering HUG members these close-out, last call and limited quantity specials from Heath Company.

The items listed are not necessarily representative of complete systems. please call your nearest Heath/Zenith Computers & Electronics Center or Heath Company toll-free at 1-800-253-7057 for more information before ordering. Alaska and Michigan residents call: 616-982-3411.

Watch for additional MEMBER ONLY discounts in upcoming issues of REMark®!

REMark Discount List

	Item	Last Adv'd Price	HUG Member Price
ADP-1501	Stereo Synthesizer	99.95	74.95
GDP-1433	Digital Golf Trainer	199 95	149.95
H-19-3	H-19 to H-88 Mod Kit	220.00	199.95
H-88-1	Floppy Disk Accessory	150.00	99.95
H-88-9	Drive Mounting & Inst	25.00	14.95
H-89A	W/O CRT, Ser I/O, Disk Cont	----	399.95
HCA-9	Print Stand For H-125	99.00	59.95
NEP-1210	Assem. Set-back Thermostat	79.95	39.99
PC-131	USR PCM-5 1200B Modem (PC)	349.95	239.95
PC-200	USR 1200B Modem Card (S100)	399.00	279.00
PCA-120-1	6-PAK Plus Game Acc'y	39.95	19.95
PM-111	Stat-Attack Anti-Static Kit	24.95	9.95
PP-215	EPSON JX-80 Color Printer	699.95	449.95
PP-240	EPSON LQ-1500	1299.00	999.00
PP-400	Koalapad Touch Tablet	99.95	69.95
PPA-240-3	LQ-1500 Interface: Serial	149.95	129.95
PPA-240-4	LQ-1500 Interface: Parallel	94.95	79.95
PPA-400-1	Koala Speedkey Software	59.95	29.95
ZA-100-4	S-100 Extender Board	59.00	19.95
ZA-219	Amber CRT/H-121 (Tube only)	75.00	14.95
ZW-219	White CRT/H-121 (Tube only)	75.00	9.95

Heathkit®

Heath Company

A subsidiary of Zenith Electronics Corporation.
Some items may not be available at all store locations
Prices, product availability and specifications subject to change without notice

CP-236

ZPC Update #10

Pat Swayne
HUG Software Engineer

This is the tenth in a series of articles about ZPC, a program that allows you to run IBM compatible software on H/Z-100 (dual processor) computers. In this edition, which will be a bit short due to other work that had to be done, I will correct a mistake from ZPC Update #8, provide new patches for Lotus 1-2-3 Release 2 and Symphony, and talk about the ZPC Hardware Support circuit. I will also say a few things about using mice.

Update #8 Correction

In ZPC Update #8, the Control-Break patch for ZPC contains an error in the DEBUG version of the patch. Near the top of the text in the second column of page 41, the line that says CMP AL,2 should be changed to CMP AL,A2.

Lotus And Symphony Patches

When you run Lotus 1-2-3 and Symphony under ZPC after patching them for correct operation, they still do not work the way they should, because the cursor does not move to where it should. That is because these software products write directly to the hardware to control the position of the cursor. I have modified the FIXLTS program presented in ZPC Update #8 so that it patches the way 1-2-3 and Symphony address the cursor. The following BASIC program, when run, will produce the new FIXLTS.COM.

```
10 REM THIS PROGRAM CREATES FIXLTS.COM
20 DEFINT A-I:OPEN "O",1,"FIXLTS.COM
30 S=0:S1 = 28784 :FOR I=1 TO 280
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 190,13,2,191,93,0,185,11,0,252
80 DATA 243,164,191,124,0,51,192,185,3,0
90 DATA 243,171,186,92,0,180,15,205,33,254
100 DATA 192,117,4,205,32,0,0,199,6,106
110 DATA 0,1,0,186,24,2,180,26,205,33
120 DATA 139,14,6,0,129,233,24,2,186,92
130 DATA 0,180,39,205,33,60,2,115,220,137
140 DATA 14,35,1,191,24,2,176,186,187,218
150 DATA 3,242,174,117,19,38,57,29,117,247
160 DATA 131,199,2,81,185,10,0,176,144,243
170 DATA 170,89,235,228,139,14,35,1,191,24
180 DATA 2,176,186,187,217,3,242,174,117,11
190 DATA 38,57,29,117,247,198,69,2,144,235
200 DATA 241,139,14,35,1,191,24,2,187,216
210 DATA 3,242,174,117,11,38,57,29,117,247
220 DATA 198,69,2,144,235,241,139,14,35,1
230 DATA 191,24,2,176,184,187,80,0,186,247
240 DATA 226,242,174,117,21,38,57,29,117,247
250 DATA 38,57,85,2,117,241,79,190,239,1
260 DATA 185,22,0,144,243,164,199,6,125,0
270 DATA 0,0,139,14,35,1,186,92,0,180
280 DATA 40,205,33,10,192,116,2,205,32,186
290 DATA 92,0,180,16,205,33,254,192,116,243
300 DATA 186,5,2,180,9,205,33,205,32,138
310 DATA 234,131,250,255,117,3,185,0,0,30
320 DATA 184,64,0,142,216,137,14,80,0,31
330 DATA 203,68,111,110,101,46,13,10,36,49
340 DATA 50,51,32,32,32,32,32,83,69,84
```

The assembly source for FIXLTS follows this article. To use FIXLTS to patch 1-2-3 Release 2, you must first install 1-2-3. Set ZPC to mode 7 (PC 7), and run the INSTALL program in Lotus 1-2-3, and answer prompts and questions as follows:

```
First Time Installation
Can your computer display graphs? Yes
How many monitors do you have? One Monitor
IBM color card, single-color monitor
(Answer printer questions as required)
Do you want to name your driver set? No
```

After you have completed the installation procedure, set ZPC to the Z-100 mode, and run FIXLTS as in this example:

```
A>FIXLTS B:
```

This example assumes that FIXLTS.COM is on a disk in drive A:, and that your 1-2-3 System Disk is in drive B:. Now you can run 1-2-3 by setting ZPC to the default PC mode (mode 3). It will display graphs in high resolution monochrome, and text in color.

You can also patch Lotus Symphony using FIXLTS. Create a batch file called FIXSYM.BAT that contains these lines:

```
REN %LOTUS.SET 123.SET
FIXLTS %1
REN %1123.SET LOTUS.SET
```

This file and FIXLTS.COM should be on the same disk. You can install Symphony using the procedure shown above for 1-2-3, but an additional question will be presented:

```
Do you want graphs and text together? No
```

You can also select either of the following from the monitor list:

```
IBM color card, single-color monitor
IBM color card, color monitor
```

After the installation is complete, fix Symphony as in this example:

```
A>FIXSYM B:
```

This example assumes that FIXLTS.COM and FIXSYM.BAT are on the disk in drive A:, and that your Symphony System disk is in drive B:.

ZPC Hardware Circuit Woes

Some users reported that the changes I presented in the last ZPC update to the ZPC Hardware Support Circuit do not work in all H/Z-100s. They report that the status line I use in the circuit for decoding ports should not be used by themselves for such, but should be ANDed with the input strobe line, pDBIN (pin 78). I tried that, and when I ran ZHSTST (see last month's ZPC Update), it reported that my 8259s were too slow. I have not tried replacing them as of this writing. It may be difficult, perhaps even impossible, to produce a ZHS circuit that works on all H/Z-100s under all conditions. If any users have been experimenting with this, I would appreciate your input.

I had intended to present some additional modifications to the ZHS board that would produce an interrupt on outputs to ports 3D8 and 3D9, so that programs that control video modes, colors, etc. by writing to those ports would run without patching. I guess those changes will have to wait till next month.

Some people have reported that they are working on commercial versions of the ZHS circuit. I hope that they solve the problems that I have mentioned in these articles, and that they will share their discoveries with the rest of us. One thing is certain: I will not sanction a manufacturer who does not work with me on the project, or who does not share any discoveries on getting the circuit to work properly with ZPC users at large.

ZPC, Mice, And Serial Ports

Paul Herman, creator of the Mouse Pack software reviewed in a previous REMark issue, reports that his product works fine under ZPC. It should, therefore, be possible to operate some graphic programs, such as PC Palette with a mouse under ZPC. If the product can be operated using the arrow keys, then it should be possible to operate it with a mouse and Mouse Pack.

Another solution to using mice would be to add a serial port to the ZPC Hardware Support circuit (and some appropriate software support). Then you could use a real PC mouse driver, or software that had mouse support built in. I have mentioned to those planning commercial ZHS boards that it would be a good idea to put a serial port on the board.

An attendee at the HUGCON this year told me that he had written an INT 14 driver (serial port driver) for ZPC that provides such a complete implementation that he is able to run a PC modem program that uses INT 14 rather than going directly to the ports. He said that he would give me a copy. Unfortunately, I don't remember his name, so I hope that if I have not received the software by the time this is printed, that he will send it.

Upgrade Your ZPC

If you are still using ZPC version 1, you're missing the power and versatility of ZPC Version 2. You can upgrade by sending your original HUG ZPC disk and $20.00 to Heath User's Group, Attn: Nancy Strunk, Hilltop Road, St. Joseph, MI 49085. If you have both the old ZPC and the ZPC Support Disk, send both in and $15.00 for the upgrade. Make checks payable to: Heath Users' Group.

FIXLTS Source Code

```
        PAGE    ,132
;       FIXLTS -- FIX LOTUS 1-2-3 RELEASE 2 TO WORK UNDER
;       ZPC WITHOUT ZHS.
;
;       BY P. SWAYNE, HUG  13-JUN-86

;       DEFINITIONS

OPENF   EQU     0FH             ;OPEN FILE
CLOSEF  EQU     10H             ;CLOSE FILE
SETDTA  EQU     1AH             ;SET DTA ADDRESS
READF   EQU     27H             ;READ FILE (RANDOM BLK.)
WRITEF  EQU     28H             ;WRITE FILE (RANDOM BLOCK)

CODE    SEGMENT
        ASSUME  CS:CODE,DS:CODE,ES:CODE,SS:CODE
        ORG     6
BUFSIZ  LABEL   WORD
        ORG     50H
CURPOS  LABEL   WORD
        ORG     5CH
FCB     LABEL   BYTE            ;DEFAULT FCB ADDRESS
        ORG     100H

START:  MOV     SI,OFFSET LTSNAM;POINT TO "123.SET"
        MOV     DI,OFFSET FCB+1 ;PUT IT HERE
        MOV     CX,11           ;11 CHARS
        CLD
        REP     MOVSB           ;MOVE IT IN
        MOV     DI,OFFSET FCB+32;POINT TO RECORD COUNTERS
        XOR     AX,AX
        MOV     CX,3
        REP     STOSW           ;CLEAR RECORD COUNTERS
        MOV     DX,OFFSET FCB
        MOV     AH,OPENF
        INT     21H             ;OPEN FILE
        INC     AL
        JNZ     OPENED          ;OPEN OK
FILERR: INT     20H             ;ELSE, EXIT
PCHSIZ  DW      0               ;STORE PATCH SIZE HERE
OPENED: MOV     WORD PTR FCB+14,1;SET RECORD SIZE TO 1
        MOV     DX,OFFSET BUFFER
        MOV     AH,SETDTA
        INT     21H             ;SET DTA TO BUFFER
        MOV     CX,BUFSIZ       ;GET BUFFER SIZE
        SUB     CX,OFFSET BUFFER;REMOVE PROGRAM SPACE
        MOV     DX,OFFSET FCB
        MOV     AH,READF
        INT     21H             ;READ FROM FILE
        CMP     AL,2
        JNC     FILERR          ;FILE ERROR
        MOV     PCHSIZ,CX       ;STORE FILE HERE
```

```
        MOV     DI,OFFSET BUFFER
SCODLP0:MOV     AL,0BAH         ;SEARCH FOR THIS
        MOV     BX,3DAH         ,AND PORT 3DAH
SCODLP: REPNZ   SCASB           ;SEARCH FOR IT
        JNZ     NXPCH           ;NOT FOUND, CHECK NEXT
        CMP     ES:[DI],BX      ;IS THIS IT?
        JNZ     SCODLP          ,NO, KEEP LOOKING
        ADD     DI,2            ;MOVE TO PATCH AREA
        PUSH    CX
        MOV     CX,10           ;PATCH 10 BYTES
        MOV     AL,90H          ;WITH "NOP"
        REP     STOSB           ,MAKE THE PATCH
        POP     CX
        JMP     SCODLP0         ,LOOK FOR MORE
NXPCH:  MOV     CX,PCHSIZ       ;GET SIZE AGAIN
        MOV     DI,OFFSET BUFFER
        MOV     AL,0BAH         ,SEARCH FOR THIS
        MOV     BX,3D9H         ,AND PORT 3D9H
SCODLP1:REPNZ   SCASB           ;SEARCH FOR IT
        JNZ     NXPCH1          ;NOT FOUND, CHECK NEXT
        CMP     ES:[DI],BX      ;IS THIS IT?
        JNZ     SCODLP1         ;NO, KEEP LOOKING
        MOV     BYTE PTR 2[DI],90H;MAKE THE PATCH
        JMP     SCODLP1         ;LOOK FOR MORE
NXPCH1: MOV     CX,PCHSIZ       ;GET SIZE AGAIN
        MOV     DI,OFFSET BUFFER
        MOV     BX,3D8H         ;FIND PORT 3D8H
SCODLP2:REPNZ   SCASB           ;SEARCH FOR IT
        JNZ     NXPCH2          ;NOT FOUND
        CMP     ES:[DI],BX      ;IS THIS IT?
        JNZ     SCODLP2         ;NO, KEEP LOOKING
        MOV     BYTE PTR 2[DI],90H;MAKE THE PATCH
        JMP     SCODLP2         ;LOOK FOR MORE
NXPCH2: MOV     CX,PCHSIZ       ;GET SIZE AGAIN
        MOV     DI,OFFSET BUFFER
        MOV     AL,0B8H         ;SEARCH FOR THIS
        MOV     BX,50H          ;AND THIS
        MOV     DX,0E2F7H       ;AND THIS
SCODLP3:REPNZ   SCASB           ;SEARCH FOR IT
        JNZ     PCHDN           ;NOT FOUND, DONE
        CMP     ES:[DI],BX      ;IS THIS IT?
        JNZ     SCODLP3         ;NO
        CMP     ES:2[DI],DX
        JNZ     SCODLP3
        DEC     DI              ,BACK UP TO PATCH AREA
        MOV     SI,OFFSET FIXCUR;POINT TO CURSOR FIX
        MOV     CX,FIXSIZ
        REP     MOVSB           ;MOVE IN PATCH
PCHDN:  MOV     WORD PTR FCB+33,0;RESET RECORD ADDRESS
        MOV     CX,PCHSIZ       ;COUNT OF BYTES TO WRITE
        MOV     DX,OFFSET FCB
        MOV     AH,WRITEF
        INT     21H             ;WRITE TO FILE
        OR      AL,AL
        JZ      PCHOK           ;WRITE OK
WRERR:  INT     20H             ;ELSE, EXIT
PCHOK:  MOV     DX,OFFSET FCB
        MOV     AH,CLOSEF
        INT     21H             ;CLOSE FILE
        INC     AL
        JZ      WRERR           ;ERROR IN CLOSING
        MOV     DX,OFFSET DONE
        MOV     AH,9
        INT     21H             ,ELSE, SAY "DONE"
        INT     20H             ;AND EXIT

;       CURSOR FIX PATCH CODE

FIXCUR  PROC    FAR
        MOV     CH,DL           ;COPY ROW TO CH
        CMP     DX,-1           ,IDLE CURSOR?
        JNZ     NOTIDL          ;NO
        MOV     CX,0            ,ELSE, POS. IS 0
NOTIDL: PUSH    DS              ;SAVE DS
        MOV     AX,40H
        MOV     DS,AX           ;POINT TO BIOS RAM
        MOV     CURPOS,CX       ;SAVE CURSOR POSITION
        POP     DS
        RET
FIXEND:
FIXCUR  ENDP
FIXSIZ  EQU     FIXEND-FIXCUR

;       DATA AREA

DONE    DB      'Done ',13,10,'$'

LTSNAM  DB      '123     SET'

BUFFER  LABEL   BYTE            ;BUFFER STARTS HERE

CODE    ENDS
        END     START
```

*

HUG NEW PRODUCTS

HUG P/N 885-8045-37

MATT $20.00

Introduction: MATT is a Turbo Pascal program designed to facilitate operations on one and two-dimensional matrices. The program is entirely menu-driven, and uses a spreadsheet type of display to make matrix entry and editing very fast and easy. Turbo Pascal is a product of Borland International, Inc.

Requirements: Two versions of MATT are included. One is for the H/Z-100 (not PC) and requires MS-DOS version 2 or above. The program uses the H-19 graphics and, therefore, requires the graphics version of ALTCHAR.SYS be on the boot disk. The other version of MATT is for the PC compatible series using MS-DOS version 2 or above. Color is not required for either version, and only the basic 128k memory is needed.

The following files are included on the disk:

MATT.DOC — Complete indexed documentation with examples.
WSMATT.DOC — WordStar formatted version of the documentation.
README.DOC — HUG disclaimer and any last minute updates.
MATT.PAS — Source code for the Z-100 version.
MATT.COM — Z-100 executable version.
PCMATT.PAS — Source code for the PC compatible version.
PCMATT.COM — PC executable version.

The following are Z-100 specific include files:

BOX	.PRO	MYIO	.PRO
PAGESWAP	.PRO		

The following are PC specific include files:

BOX	.PC	MYIO	.PC

The following are included in both versions:

DIREC	.PRO	INV	.PRO
SUM	.PRO	MULT	.PRO
TRANS	.PRO		

Author: Dennis K. Greer

Program Content: MATT is designed to provide a very user-friendly environment in which to perform most operations on one and two dimensional matrices. Matrix operations are performed on two input matrices with the results stored in a third. Maximum size for each of the matrices is 59 rows by 59 columns. Among the operations available are:

- Determinants and Inverses
- Element, Row, and Column editing
- File and Directory Operations
- Matrix Initialization and Copying

Continued on Page 83

TABLE C
Product Rating

10 - Very Good
9 - Good
8 - Average

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

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

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

ORDERING INFORMATION

For Visa and MasterCard phone orders; telephone Heath Company Parts Department at (616) 982-3571 Have the part number(s), descriptions, and quantity ready for quick processing By mail; send order, plus 10% postage and handling ($1.00 minimum charge, up to a maximum of $5.00. UPS is $1.75 minimum -- no maximum on UPS. UPS Blue Label is $4.00 minimum.), to Heath Company Parts Department, Hilltop Road, St. Joseph, MI 49085. Visa and MasterCard require minimum $10.00 order.

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

NOTE

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

Note: All special update offers announced in REMark (i.e. ZPC II update) must be paid by check or money order, payable to the Heath Users' Group. **NO CREDIT CARDS ACCEPTED.** ZPC II contains only one disk. It is a combination of ZPC I and the ZPC Support disk plus added improvements. Thank you.

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
HDOS HARDCOPY SOFTWARE			
885-1008	Volume I Documentation	9.00	
885-1013	Volume II Documentation	12.00	
885-1015	Volume III Documentation	9.00	
885-1037	Volume IV Documentation	12.00	8
885-1058	Volume V Documentation	12.00	
MISCELLANEOUS HDOS COLLECTIONS			
885-1032	Disk V H8/H9	18.00	8
885-1044-[37]	Disk VI H8/H89	18.00	
885-1064-[37]	Disk IX H8/H89 Disk	18.00	
885-1066-[37]	Disk X H8/H89	18.00	10
885-1069	Disk XIII Misc H8/H89	18.00	
885-1135-[37]	HDOS Variety Pkg.	20.00	76
GAMES			
HDOS			
885-1010	Adventure Disk H8/H89	10.00	4
885-1029-[37]	Disk II Games 1 H8/H89	18.00	8
885-1030-[37]	Disk III Games 2 H8/H89	18.00	8
885-1031	Disk IV MUSIC H8 Only	20.00	25
885-1067-[37]	Disk XI H8/H19/H89 Games	18.00	12
885-1068	Disk XII MBASIC Graphic Games	18.00	10
885-1088-[37]	Disk XVII MBASIC Graph. Games	20.00	14
885-1093-[37]	D&D H8/H89 Disk	20.00	16
885-1096-[37]	MBASIC Action Games H8/H89	20.00	18
885-1103	Sea Battle HDOS H19/H8/H89	20.00	20
885-1111-[37]	HDOS MBASIC Games H8/H89	20.00	23
885-1112-[37]	HDOS Graphic Games H8/H89	20.00	23
885-1113-[37]	HDOS Action Games H8/H89	20.00	23
885-1114	H8 Color Raiders & Goop	20.00	23
885-1124	HUGMAN & Movie Animation Pkg	20.00	41
885-1125	MAZEMADNESS	20.00	41
885-1130	Star Battle	20.00	45
885-1133-[37]	HDOS Games Collection I	20.00	59
885-8009-[37]	HDOS & CP/M Galactic Warrior	20.00	32
885-8022	HDOS SHAPES	16.00	45
885-8026	HDOS Space Drop	16.00	49
885-8032-[37]	HDOS Castle	20.00	59
CP/M			
885-1206-[37]	CP/M Games Disk	20.00	11
885-1209-[37]	CP/M MBASIC D&D	20.00	19
885-1211-[37]	CP/M Sea Battle	20.00	20
885-1220-[37]	CP/M Action Games	20.00	32
885-1222-[37]	CP/M Adventure	10.00	35
885-1227-[37]	CP/M Casino Games	20.00	38
885-1228-[37]	CP/M Fast Action Games	20.00	39
885-1236-[37]	CP/M Fun Disk I	20.00	55
885-1248-[37]	CP/M Fun Disk II	35.00	69
ZDOS			
885-3004-37	ZDOS ZBASIC Graphic Games	20.00	37
885-3009-37	ZDOS ZBASIC D&D	20.00	50
885-3011-37	ZDOS ZBASIC Games Disk	20.00	52
885-3017-37	ZDOS Contest Games Disk	25.00	58
885-8042-37	ZDOS/MSDOS Poker Party	20.00	77
UTILITIES			
HDOS			
885-1022-[37]	HUG Editor (ED) Disk H8/H89	20.00	20
885-1025	Runoff Disk H8/H89	35.00	
885-1060-[37]	Disk VII H8/H89	18.00	
885-1061	TMI Load H8 ONLY Disk	18.00	
885-1062-[37]	Disk VIII H8/H89 (2 Disks)	25.00	
885-1063	Floating Point Disk H8/H89	18.00	
885-1065	Fix Point Package H8/H89 Disk	18.00	10
885-1075	HDOS Support Package H8/H89	60.00	
885-1077	TXTCON/BASCON H8/H89	18.00	
885-1079-[37]	HDOS Page Editor	25.00	15
885-1080	EDITX H8/H19/H89 Disk	20.00	
885-1082	Programs for Printers H8/H89	20.00	
885-1083-[37]	Disk XVI Misc H8/H89	20.00	11
885-1089-[37]	Disk XVIII Misc H8/H89	20.00	20
885-1090-[37]	Disk XIX Utilities H8/H89	20.00	22
885-1092-[37]	Relocating Debug Tool H8/H89	30.00	14
885-1098	H8 Color Graphics ASM	20.00	19
885-1099	H8 Color Graphics Tiny PASCAL	20.00	19
885-1105	HDOS Device Drivers H8/H89	20.00	24
885-1116	HDOS Z80 Debugging Tool	20.00	27
885-1119-[37]	BHBASIC Support	20.00	29
885-1120-[37]	HDOS 'WHEW' Utilities	20.00	33
885-1121	HDOS Hard Sec Sup Pkg 2 Disks	30.00	37
885-1123	XMET Robot Cross Assembler	20.00	40
885-1126	HDOS Utilities by PS:	20.00	42
885-1127-[37]	HDOS Soft Sector Support Pkg	30.00	45
885-1128-[37]	HDOS DISKVIEW	16.00	46
885-1129-[37]	HDOS CVT Color Video Terminal	20.00	46
885-8001	SE (Screen Editor)	25.00	28
885-8003	BHTOMB	25.00	28
885-8004	UDUMP	35.00	28
885-8006	HDOS SUBMIT	20.00	31
885-8007	EZITRANS.	30.00	30
885-8015	HDOS TEXTSET Formatter	30.00	42
885-8017	HDOS Programmers Helper	16.00	42
885-8024	HDOS BHBASIC Utilities Disk	16.00	46
CP/M			
885-1210-[37]	CP/M ED (same as 885-1022)	20.00	20
885-1212-[37]	CP/M Utilities H8/H89	20.00	21
885-1213-[37]	CP/M Disk Utilities H8/H89	20.00	22
885-1217-[37]	HUG Disk Duplication Utilities	20.00	26
885-1223-[37]	HRUN HDOS Emulator 3 Disks	40.00	37
885-1225-[37]	CP/M Disk Dump & Edit Utility	30.00	40
885-1226-[37]	CP/M Utilities by PS:	20.00	40
885-1229-[37]	XMET Robot Cross Assembler	20.00	40
885-1230-[37]	CP/M Function Key Mapper	20.00	42
885-1231-[37]	Gross Ref Utilities for MBASIC	20.00	43
885-1232-[37]	CP/M Color Video Terminal	20.00	46
885-1235-37	CP/M COPYDOS	20.00	54
885-1237-[37]	CP/M Utilities	20.00	55
885-1245-37	CP/M-85 KEYMAP	20.00	63
885-1246-[37]	CP/M HUG File Manager & Utilities	20.00	64
885-1247-37	CP/M-85 HUG Bkgrd Print Spooler	20.00	67
885-5001-37	CP/M-86 KEYMAP	20.00	51
885-5002-37	CP/M-86 HUG Editor	20.00	52
885-5003-37	CP/M-86 Utilities by PS:	20.00	54
885-5008-37	CP/M 8080 To 8088 Trans. & HFM	20.00	64
885-5009-37	CP/M-86 HUG Bkgrd Print Spool	20.00	66
885-8018-[37]	CP/M Fast Eddy & Big Eddy	20.00	43
885-8019-[37]	DOCUMAT and DOCULIST	20.00	43
885-8025-37	CP/M-85/86 Fast Eddy	20.00	49
ZDOS/MSDOS			
885-3005-37	ZDOS Etchdump	20.00	39
885-3007-37	ZDOS CP/EMulator	20.00	47
885-3008-37	ZDOS Utilities	20.00	47
885-3010-37	ZDOS Keymap	20.00	51
885-3022-37	ZDOS/MSDOS Useful Programs I	30.00	63
885-3023-37	ZDOS/MSDOS EZPLOT	20.00	63
885-3026-37	MSDOS SMALL C Compiler	25.00	65
885-3031-37	ZDOS/MSDOS Graphics	20.00	69
885-3037-37	MSDOS Z-100 PC Emulator II	60.00	76
885-8029-37	ZDOS Fast Eddy	20.00	53
885-8035-37	MSDOS DOCUMAT and DOCULIST	20.00	70
885-8041-37	ZDOS/MSDOS Orbits	25.00	75
H/Z100 ZDOS/MSDOS - H/Z150 PC MSDOS			
885-3012-37§§	ZDOS HUG Editor	20.00	52
885-3014-37§§	ZDOS/MSDOS Utilities II	20.00	54
885-3016-37§	ZDOS/MSDOS Adventure	10.00	57
885-3020-37§	MSDOS HUG Menu System	20.00	62
885-3021-37§§	ZDOS/MSDOS Cardcat	20.00	63
885-3024-37§	ZDOS/MSDOS 8080 To 8088 Trans	20.00	64
885-3025-37§§	ZDOS/MSDOS Misc. Utilities	20.00	64
885-3029-37§§	ZDOS/MSDOS HUG Bg. Print Spool	20.00	66
885-3035-37§§	MSDOS SPELL5 & SPELL5F	20.00	72
885-3038-37§	ZDOS/MSDOS DEBUG Support Util	20.00	77
885-8039-37§§	MSDOS DPATH	20.00	74
885-8040-37§§	MSDOS HELP Programs	20.00	74

§ All program files run on both

§§ Program files run partially on both

Part Number	Description of Product	Selling Price	Vol. Issue
PC/IBM COMPATIBLE			
885-6001-37	MSDOS Keymapper	20.00	59
885-6002-37	CP/EMulator II & ZEMulator	20.00	59
885-6003-37	MSDOS EZPLOT	20.00	65
885-6004-37	MSDOS CheapCalc	20.00	67
885-6005-37	MSDOS Skyviews	20.00	67
885-6006-37	MSDOS Cardcat	20.00	69
885-6007-37	MSDOS DND (Dung. & Dragons)	20.00	70
885-6009-37	MSDOS Screen Saver Plus	20.00	76
885-8033-37	MSDOS Fast Edit	20.00	62
885-8037-37	MSDOS Grade	20.00	70
885-8044-37	MSDOS TCSpell	20.00	79
PROGRAMMING LANGUAGES			
HDOS			
885-1038-[37]	Wise on Disk H8/H89	18.00	
885-1042-[37]	PILOT on Disk H8/H89	19.00	
885-1059	FOCAL-8 H8/H89 Disk	25.00	13
885-1078-[37]	HDOS Z80 Assembler	25.00	21
885-1085	PILOT Documentation	9.00	
885-1086-[37]	Tiny HDOS PASCAL H8/H89	20.00	13
885-1094	HDOS Fig-Forth H8/H89	40.00	18
885-1132-[37]	HDOS Tiny BASIC Compiler	25.00	59
885-1134	HDOS SMALL-C Compiler	30.00	63
CP/M			
885-1208-[37]	CP/M Fig-Forth H8/H89 2 Disks	40.00	18
885-1215-[37]	CP/M BASIC-E	20.00	26
BUSINESS, FINANCE AND EDUCATION			
HDOS			
885-1047	Stocks H8/H89 Disk	18.00	
885-1048	Personal Account H8/H89 Disk	18.00	
885-1049	Income Tax Records H8/H89 Disk	18.00	
885-1055-[37]	MBASIC Inventory Disk H8/H89	30.00	
885-1056	MBASIC Mail List	30.00	
885-1070	Disk XIV Home Fin H8/H89	18.00	
885-1071-[37]	MBASIC SmBusPk H8/H19/H89	75.00	17
885-1091-[37]	Grade/Score Keeping H8/H89	30.00	14
885-1097-[37]	MBASIC Quiz Disk H8/H89	20.00	18
885-1118-[37]	MBASIC Payroll	60.00	30
885-1131-[37]	HDOS CheapCalc	20.00	47
885-8010	HDOS Checkoff	25.00	32
885-8021	HDOS Student's Statistics Pkg	20.00	44
885-8027	HDOS SciCalc	20.00	50
CP/M			
885-1218-[37]	CP/M MBASIC Payroll	60.00	31
885-1233-[37]	CP/M CheapCalc	20.00	47
885-1239-[37]	Spread Sht. Contest Disk I	20.00	
885-1240-[37]	Spread Sht. Contest Disk II	20.00	
885-1241-[37]	Spread Sht. Contest Disk III	20.00	
885-1242-[37]	Spread Sht. Contest Disk IV	20.00	
885-1243-[37]	Spread Sht. Contest Disk V	20.00	
885-1244-[37]	Spread Sht. Contest Disk VI	20.00	
885-8011-[37]	CP/M Checkoff	25.00	32
885-8036-[37]	CP/M Grade	20.00	70
ZDOS/MSDOS H/Z100 ONLY			
885-3006-37	ZDOS CheapCalc	20.00	47
885-3013-37	ZDOS Checkbook Manager	20.00	54
885-3018-37	ZDOS Contest Spreadsheet Disk	25.00	58
885-8028-37	ZDOS SciCalc	20.00	50
885-8030-37	ZDOS Mathflash	20.00	55
885-8043-37	MSDOS Calc	20.00	80

Continued on Page 83

ZENITH data systems

COMPUTERS AND ACCESSORIES

ZENITH COMPUTERS

ZF-138-42R:	Dual Drives, 640K RAM	$1550.00
ZF-148-41R:	One Drive, 640K RAM	$1175.00
ZF-148-42R:	Dual Drives, 640K RAM	$1400.00
ZF-158-41R:	One Drive, 640K RAM	$1700.00
ZF-158-42R:	Dual Drives, 640K RAM	$1925.00
ZW-158-42R:	One Drive, 640K RAM, 10.5 MB H/D	$2825.00
ZFL-171-42R:	Dual Drive, 640K RAM	$2075.00
ZF-241-81R:	One Drive, 512K RAM, Color or Monochrome Card (Your Choice)	$3150.00
ZW-241-82R:	One Drive, 512K RAM, 20 MB, H/D, Color or Monochrome Card (Your Choice)	$4350.00

PLEASE NOTE: The above systems are strictly Zenith manufacture. If you wish to configure a system to your specifications, additional savings may be realized. Please call or write us for a quotation.

MONITORS & VIDEO CARDS

ZVM-1220-A:	Non-Glare, Amber Monitor	$119.00
ZVM-1230-A:	Non-Glare, Green Monitor	$119.00
ZVM-1240:	Non-Glare, Amber Monitor, TTL	$169.00
ZVM-1200-1:	Tilt Swivel Base	$ 20.00
ZVM-1330:	High Resolution Color, RGB	$559.00
ZVM-1360:	High Resolution Color, RGB, Long Persistence Phosphor	$719.00
ZVM-135:	High Resolution Color, RGB & Composite	$450.00
Princeton SR-12:	High-Resolution Color, RGB, 31 KHz. Bandwidth (For Use With Sigma Color 400)	$599.00
Mitsubishi Analog Color Monitor (1024 x 1024)		$2460.00
Everex Edge Monochrome Display Adapter		$325.00
Sigma Color 400 Board W/PC Paint (640 x 400)		$596.00
Control Systems Artist I Analog Color Graphics Board (1024 x 1024)		$1999.00

CENTRAL PROCESSOR UNITS

INTEL 8087-2 — 8 MHz. Math Co-Processor	$129.00
INTEL 80287 — 6 MHz. Math Co-Processor	$185.00
NEC V20 — 8 MHz. Processor: Direct Replacement for the INTEL 8088 but 10-30% Faster	$ 19.00

DISKETTES

Nashua Diskettes: DS/DD, 48 TPI, Soft Sector or Hard Sector (Specify), Box of 10 with lifetime guarantee. Special! ..... $ 9.50

Nashua Diskettes: DS/DD, 96 TPI, Soft Sector or Hard Sector (Specify), Box of 10 with lifetime guarantee Special! $16.50

Nashua Diskettes: DS/ High Density, Soft Sector, 600 Oersted Media for IBM AT and Compatible 1.2 MB Disk Drives. Box of 10 with lifetime guarantee. Special! $34.50

DYNAMIC RAM MEMORY

4164	64K 150 NS. DRAM	QTY: 1-50 $1.40
41256	256K 150 NS. DRAM	QTY: 1-50 $3.40

Buy any Memory Upgrade Accessory for your Heath/Zenith 100/150/160/200 Series Computer and receive a 10% discount on Dynamic RAM with your order.

HARD DISK DRIVES AND ACCESSORIES

PC COMPATIBLE HARD DISK DRIVES

SHUGART 10 MB. Hard Disk With Xebec Controller (Zenith Compatible)	$475.00
SEAGATE 10 MB Hard Disk With Xebec Controller (Zenith Compatible)	$499.00
SEAGATE 20 MB. Hard Disk With: Xebec Controller (Zenith Compatible)	$595.00
TULIN 30 MB. Hard Disk With: DTC Controller	$925.00
SYQUEST 10 MB. Removeable Media Hard Disk Drive with Xebec Controller, Mounting Hardware, Cables and One Cartridge	$1065.00
Additional Cartridges	$125.00

AT COMPATIBLE HARD DISK DRIVES

SEAGATE ST4026 (20 MB. Formatted), 40 MS. AVG.	$ 775.00
SEAGATE ST4038 (30 MB. Formatted), 40 MS. AVG.	$ 900.00
SEAGATE ST4051 (40 MB. Formatted), 40 MS. AVG.	$1095.00

Z100 COMPATIBLE HARD DISK DRIVES

10 MB. Hard Disk With Zenith Controllers and Installation Hardware, 85 MS. AVG.	$ 999.00
20 MB. Hard Disk Drive With Zenith Controllers and Installation Hardware, 85 MS. AVG.	$1350.00
30 MB. Hard Disk Drive With Zenith Controllers and Installation Hardware, 40 MS. AVG.	$1800.00

MITSUBISHI DISK DRIVES

M4851:	5¼" Half Height, 48 TPI, DS/DD	$105.00
M4853:	5¼" Half Height, 96 TPI, DS/DD	$135.00
M4854:	5¼" Half Height, 96 TPI, 1.2 MB, Z200/IBM AT Compatible	$235.00
M2896:	8" Half Height, 48 TPI, DS/DD, Shugart 850/860 Compatible	$375.00

MODEMS

PROMETHEUS PROMODEM 1200	$299.00
PROMETHEUS PROMODEM 1200B W/ Mite	$255.00
PROMETHEUS 1200 OPTIONS PROCESSOR	$ 89.00
PROMETHEUS 1200 ALPHA-NUMERIC DISPLAY	$ 89.00
HAYES SMARTMODEM 300	$165.00
HAYES SMARTMODEM 1200	$439.00
HAYES SMARTMODEM 1200B W/O Software	$369.00
HAYES SMARTMODEM 1200B W/Smartcom II	$415.00
HAYES SMARTMODEM 2400	$685.00
US ROBOTICS PASSWORD 300/1200	$215.00
US ROBOTICS MICROLINK 1200 W/Telpac	$250.00
US ROBOTICS MICROLINK 2400 W/Telpac	$420.00
US ROBOTICS COURIER 2400	$420.00

Z100 UPGRADE ACCESSORIES

C.D.R. Z-100 Speed Module	$39.95
SOFTWARE WIZARDRY'S RAMPAL 100	$69.95
FBE RESEARCH ZMF-100: Install Up To 768K On Z-100 Mothersboards #181-4917 Or Lower	$59.95

Z-150/160 UPGRADE ACCESSORIES

HYPERCLOCK, Clock/Calendar and Software	$ 44.95
INTEL ABOVE BOARD PC, 64K Expandable to 2 MB	$316.00
RAM TECHNOLOGY'S **Z-MAX 150**: Install up to 704K on H/Z 150/160 Computers using 256K DRAM	$ 29.95
RAM TECHNOLOGY'S **Z-MEG 150**: Install 5 banks of 256K DRAM (1280K) on H/Z 150/160 Computers. 704K Memory is Directly Addressable, Remaining 512K is Available As RAM Disk	$ 59.95

Z-200 UPGRADE ACCESSORIES

INTEL ABOVE BOARD AT, 128K Expandable to 2 MB	$470.00
INTEL ABOVE BOARD AT PIGGYBACK, 128K Expandable to 2 MB. Expands Above Board AT to 4 MB	$233.00

FOR YOUR INFORMATION

Prices and specifications subject to change without notice. All hardware carries a 90-day warranty. Personal checks are held 10 working days for bank clearance; money orders are accepted as cash. Please add 2% to your order to cover shipping and handling; any surplus will be refunded. Orders may be sent COD on request, cash or certified check only. Prices reflect a 4.5% cash discount; please add 4.5% to VISA/MC orders. Purchase orders accepted from businesses and educational institutions with prior credit approval. User group inquiries invited. Please call us Tuesday through Saturday, 10:00 A.M. to 7:00 P.M. EST, or write to RAM Technology, 427-3 Amherst Street, Suite 265, Nashua, New Hampshire 03063. We appreciate your business and guarantee satisfaction.

RAM Technology

427-3 AMHERST STREET, SUITE 265
NASHUA, NEW HAMPSHIRE 03063

1-800-662-0070

In Alaska and New Hampshire: (603) 889-0633

HOURS OF OPERATION: TUESDAY - SATURDAY 10:00 - 7:00 P.M. EST

MasterCard

VISA

SEEK.COM

A CP/M Assembly Language Learning Adventure

Part 4

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

Introduction

In the last article, we developed the 3 text file counting routines for SEEK.ASM. Now, the program accomplishes the main goal: count the lines, words, and sentences in a text file. However, this information is buried in RAM in binary form. So the next objective seems clear: extract the stored values and convert them to decimal. To this end, this article will show how this task is performed. In addition, we'll add some more messages, and a routine that determines the actual size of the text file.

Size Routine

Calculating the text file size was an afterthought on my part. While developing SEEK.ASM, I noticed that after each counting routine finished, the EOF address was left in the HL pair. Couple this with the knowledge that TXTB contains the file's starting address, and you have the elements needed to perform the size calculation. To start with, add the line CALL SIZE immediately after CALL SNTCS in the ASM file. Then insert the following routine:

```
SIZE    LXI     B,TXTB-1        ; Text start --> BC pair
        MOV     A,L             ; LSB of HL to A
        SUB     C               ; Subtract LSBs
        MOV     L,A             ; Store diff  --> L
        MOV     A,H             ; MSB of HL to A
        SBB     B               , Subtract MSBs (with borrow)
        MOV     H,A             ; Store diff. --> H
        SHLD    SAV1            ; Store HL -->SAV1
        RET
```

Here's how SIZE works. Upon exit from SNTCS, the HL pair holds the address of the EOF marker. So SIZE loads the BC pair with a value that is 1 less than the actual text file start address. This is done to offset subtraction error. For example, if you have a 10 byte file stored in RAM and you subtract the first and last addresses (e.g. 0100H and 0109H) the result is 9. This answer is off by 1. To correct for this, we load BC with TXTB–1 rather than TXTB. Now with the HL and BC pairs loaded, a 16 bit subtraction is performed and the answer stored in the SAV1 location. Although this DW area was previously used for another purpose, it is now free to serve double duty. No need to waste 2 more bytes on an extra storage area. With the completion of this routine, we can now focus our attention on the binary to decimal conversion routine.

Step 3 — Module 6 (RESLT)

The RESLT routine is placed between the CALL SIZE line and the EPGM label, and organizes 2 activities: message displays, and the binary to decimal conversion routine. The sequence of these tasks may also be placed in flow chart form. However, because of their repetitive nature, I will list them:

1. Print the text file name.
2. Print a message and then the LINES count.
3. Print a message and then the SNTCS count.
4. Print a message and then the WORDS count
5. Print a message and then the File Size.
6. Print a program end message.

These steps translate to the following ASM instructions:

```
RESLT   LXI     H,MSG8    ;Point to MSG8  ¦ LXI   H,MSG11
        CALL    PRNSEQ    ;Print MSG      ¦ CALL  PRNSEQ
        LXI     H,FCB+1   ;Point To Fname ¦ LHLD  CTR+2
        CALL    PRNSEQ    ;Print Fname    ¦ CALL  ASCII
        LXI     H,MSG9    ;Point to MSG9  ¦ LXI   H,MSG12
        CALL    PRNSEQ    ;Print MSG      ¦ CALL  PRNSEQ
        LHLD    CTR       ;Count ->HL     ¦ LHLD  SAV1
        CALL    ASCII     ;Conv  to dec   ¦ CALL  ASCII
        LXI     H,MSG1Ø   ,               ¦ LXI   H,ENMSG
        CALL    PRNSEQ    ;               ¦ CALL  PRNSEQ
        LHLD    CTR+4     ;               ¦
        CALL    ASCII     ;and so on..    ¦
```

With the exception of locating the file name (Fname) and the ASCII routine, these instructions do not represent any new information. As such, I don't believe it's necessary to, again, describe operations such as message printing, etc. The text of these messages can be found in Listing #1 (the final form of SEEK.ASM).

```
;                    * * *  Listing #1  * * *
;
; 9/19/85                      SEEK.ASM
; Ver. 1.2 - Modified SIZE routine (11/16/85).
; SEEK loads a text file into memory, counts the # of
; words, lines, and sentences in the file, then displays
; the results
;          ---- By: M.D. Zapolski, Sr  ----
;
BDOS    EQU     05H
FCB     EQU     05CH
OPEN    EQU     0FH
STDMA   EQU     01AH
RDSEQ   EQU     014H
CNOUT   EQU     02H
CR      EQU     0DH
LF      EQU     0AH
TOP     EQU     0E7H
;
;
        ORG     0100H
;
BEGIN   LXI     H,MSG1          ; Clr scrn, hm cursor,
                                  Intro msg
        CALL    PRNSEQ
        CALL    OPNRD           ; Open file and read
                                  into memory
                                ; Determine Text File
                                  Statistics
        LXI     H,MSG7
        CALL    PRNSEQ
        LXI     H,TXTB
        CALL    LINES
        LXI     H,TXTB
        CALL    WORDS
        LXI     H,TXTB
        CALL    SNTCS
        CALL    SIZE
                                ; Print results on CRT
RESLT   LXI     H,MSG8
        CALL    PRNSEQ
        LXI     H,FCB+1         ; point to filename
        CALL    PRNSEQ
        LXI     H,MSG9
        CALL    PRNSEQ
        LHLD    CTR
        CALL    ASCII
        LXI     H,MSG10
        CALL    PRNSEQ
        LHLD    CTR+4
        CALL    ASCII
        LXI     H,MSG11
        CALL    PRNSEQ
        LHLD    CTR+2
        CALL    ASCII
        LXI     H,MSG12
        CALL    PRNSEQ
        LHLD    SAV1
        CALL    ASCII
        LXI     H,ENMSG
        CALL    PRNSEQ
EPGM    JMP     0000H
;
NXLN    INX     H               ; next char
LINES   MOV     A,M             ; get character
        CPI     1AH             ; @ EOF ?
        RZ                      ; return if @ EOF
        CPI     0DH             ; find a CR ?
        JNZ     NXLN            ; jmp if not found
FNLN    PUSH    H               ; increment counter
        LHLD    CTR
        INX     H
        SHLD    CTR
        POP     H
        JMP     NXLN
NXWD    INX     H               ; next char.
WORDS   MOV     A,M
NXPN    CPI     1AH             ; @ EOF ?
        RZ                      ; return if @ EOF
        ADI     0DFH            ; set CY if >20h
        JNC     NXWD            ; jmp if <=20h
        MOV     A,M             ; restore char
        ADI     0C6H            ; set CY if >39h
        JC      NXCR            ; jmp if >39h
NXNM    ADI      0AH
        JNC     NXWD
        INX     H
        MOV     A,M
        ADI     0DFH
        JC      NXNM
        JMP     FDWD
NXCR    INX     H
        MOV     A,M
        CPI     02DH
        JZ      NXCR
        ADI     0BFH
        JC      NXCR
FDWD    PUSH    H               ; increment counter
        LHLD    CTR+2
        INX     H
        SHLD    CTR+2
        POP     H
        JMP     NXWD
NXSN    INX     H
SNTCS   MOV     A,M
        CPI     1AH             ; @ EOF ?
        RZ                      ; return if @ EOF
        ADI     0DFH            ; ck if <=20h
        JNC     NXSN            ; pass if >20h
        MOV     A,M             ; restore (A)
        ADI     0C0H            ; ck if >3Fh
        JC      NXSN            ; pass if <=3Fh
        MOV     A,M
        CPI     21H             ; ck for !
        JZ      FND
        CPI     22H             ; ck for "
        JZ      FND
        CPI     2EH             ; ck for .
        JZ      FND
        CPI     3FH             ; ck for ?
        JNZ     NXSN
FND     INX     H
        INX     H
        MOV     A,M
        CPI     20h
        JZ      FDSN
        CPI     0DH
        JZ      FDSN
        CPI     0AH
        JNZ     NXSN
FDSN    PUSH    H               ; increment counter
        LHLD    CTR+4
        INX     H
        SHLD    CTR+4
        POP     H
        JMP     NXSN
SIZE    LXI     B,TXTB-1        ; Point to EOF address
        MOV     A,L
        SUB     C
        MOV     L,A
        MOV     A,H
        SBB     B
        MOV     H,A
        SHLD    SAV1
        RET
OPNRD   LXI     H,5DH           ; Look for 1st  char
                                  of Fname
        MOV     A,M
        CPI     20H
        JZ      NFILE
        LXI     D,FCB
        MVI     C,OPEN          ; Open file- CP/M func.
                                  # 15
        CALL    BDOS
        CPI     0FFH
        JNZ     RDFILE
        LXI     H,MSG3
        CALL    PRNSEQ
        JMP     EPGM
NFILE   LXI     H,MSG2
```

H/Z89 and H8 High Performance Products

Discover why hundreds of H/Z89's still outperform PC clones!

Hard Disk Systems

- Latest Technology 3.5 and 5.25 Inch Drives and Controller
- Includes Comprehensive Support Software for CP/M and HDOS
- Fully Compatible with H17, H37, H47, 4 MHz, CDR, Magnolia, etc.
- Includes 2 Centronics Parallel Printer Ports and Power Supply
- 6 Month Limited Warranty with 1 Year Warranty on Drive Unit
- Fast and Easy to Install for Either an H/Z89 or H8

Includes Z37 Compatible Floppy Disk Controller!

10 MB for Internal H/Z89 Mounting (65 ms)	$795
20 MB for Internal H/Z89 Mounting (65 ms)	$895
10 MB with External Cabinet (65 ms)	$895
20 MB with External Cabinet (65 ms)	$995
30 MB with External Cabinet (40 ms)	$1325
40 MB with External Cabinet (40 ms)	$1435
67 MB with External Cabinet (40 ms)	$2295
Optional 40 Track Floppy Drive (as Shown)	$125
Optional 80 Track Floppy Drive	$175
Users' Manual Only (Deductable)	$10

Computer Aided Drafting

- Turn Your H/Z89 or H8/H19 into a True CAD Graphics Workstation!
- High Resolution 640 by 768 Pixels with 640 by 250 Display Window
- Optical Mouse Control Drafts Complex Designs in Seconds!
- For all Types of Drafting, Electrical, Mechanical, Architectural, etc.
- Compares to Similar CAD Software Costing 10 Times as Much!
- Supports Hardcopy Output to Most Dot Matrix Printer Types
- Includes 256k RAM Drive and Print Spooler for CP/M and HDOS

Interactive Graphics Controller Hardware	$395
CAD—89 Software Package for CP/M or HDOS	$175
Optical Mouse Drafting Input Device (Shown)	$145
Users' Manual Only (Deductable)	$10

MasterCard VISA

Call NOW so we can rush our complete information package to you!

SigmaSoft AND SYSTEMS

"Support is the most important feature."

17000 Dallas Parkway, Suite #207, Dallas, TX 75248

(214) 380-6187

Please include $10 for UPS shipping and specify H/Z89 or H8 with your order. Texas residents please add 6% sales tax.

```
        CALL    PRNSEQ
        JMP     EPGM
RDFILE  LXI     H,MSG4
        CALL    PRNSEQ
        LXI     H,TXTB
        SHLD    SAV1
RDBLK   XCHG                    ; DMA addr  --> (DE)
        MVI     C,STDMA         ; Set DMA-CP/M func.#26
        CALL    BDOS
        LXI     D,FCB
        MVI     C,RDSEQ         ; Read seq.-CP/M func.#20
        CALL    BDOS
        CPI     0H              , Read ok ?
        JZ      NXBLK           ; Yes, if (A)=0
        CPI     1H              ; 9 EOF ?
        RZ                      ; Yes, return-file in
                                  memory
        LXI     H,MSG5
        CALL    PRNSEQ
        JMP     EPGM
NXBLK   LHLD    SAV1
        LXI     D,80H
        DAD     D
        SHLD    SAV1
CKSIZ   MVI     A,TOP
        CMP     H
        JNZ     RDBLK
        LXI     H,MSG6
        CALL    PRNSEQ
        JMP     EPGM
ASCII   LXI     B,10000D        ; load 10k's place
        CALL    CONV
        LXI     B,1000D         ; load 1k's place
        CALL    CONV
        LXI     B,100D          ; load 100's place
        CALL    CONV
        LXI     B,10D           ; load 10's place
        CALL    CONV
        MOV     A,L         ; move 1's to (A)
        ADI     30H         ; conv to ASCII in HEX
        CALL    CHROUT
        RET
CONV    MVI     D,0FFH      ; sub  routine (value in HL)
LP      MOV     A,L         ; LSB --> (A)
        SUB     C           , sub LSB's
        MOV     L,A         ; store diff  --> (L)
        MOV     A,H         ; MSB --> (A)
        SBB     B           ; sub MSB's
        MOV     H,A         ; store diff  --> (H)
        INR     D           ; count # of subtractions
        JNC     LP          ; cont  until MSB borrow
                              occurs
        DAD     B           ; 1 too many- restore BC
        MOV     A,D         ; char  partial value --> (A)
        ADI     30H         , conv to ASCII
        CALL    CHROUT      , print it
        RET
PRNSEQ  XRA     A           ; Zero (A)
        ADD     M           , (A)+<M> -->(A)
        RZ                  , Return if (A)=0 (seq.
                              complete)
        CALL    CHROUT      ; Send character to CRT
                              (console)
        INX     H           , Point (HL) to next character
        JMP     PRNSEQ      ; loop until done
CHROUT  PUSH B!PUSH D!PUSH H
        MVI     C,CNOUT
        MOV     E,A
        CALL    BDOS
        POP H!POP D!POP B
        RET
                            , DB/DW, Message Area
MSG1    DB   1BH,'E',CR,LF,1BH,'pSEEK.COM',1BH,'q
               Ver 1.2 (11/16/85)'
        DB   CR,LF,'Coded by: M.D  Zapolski, Sr ',
               CR,LF,LF,LF,0
MSG2    DB   'No filename specified... Syntax:
               SEEK Fname.ext',CR,LF,0
MSG3    DB   1BH,'pERROR:',1BH,'q File not found.  ',
               CR,LF,0
MSG4    DB   'Loading file ...',CR,LF,0
MSG5    DB   1BH,'pERROR:',1BH,'q Can''t read file   ',
               CR,LF,0
MSG6    DB   'File size exceeds memory availability.. ',
               CR,LF,0
MSG7    DB   LF,'Searching File  . ',CR,LF,0
MSG8    DB   LF,LF,'Statistics for file: ',0
MSG9    DB   CR,LF,09,09,'1) # of physical lines. . ',0
MSG10   DB   CR,LF,09,09,'2) # of sentances.   ',0
MSG11   DB   CR,LF,09,09,'3) # of words.   ',0
MSG12   DB   CR,LF,09,09,'4) File size (bytes)    ',0
ENMSG   DB   CR,LF,LF,LF,LF,09,09,09,
               '---- Program End ----',0
SAV1    DW   0
CTR     DW   0,0,0
TXTB    EQU  $
        END  BEGIN
```

Listing 1

As you can see, the first new item in RESLT is locating the file name. If you'll recall from Part 2 of this series, we stated that CP/M's default FCB started at 05CH. The structure of this FCB is:

Item	Address	# Bytes
1. Drive code	5CH	1
2. File Name	5DH-64H	8
3. Extension	65H-67H	3
4. Current extent #	68H	1
5. Misc. information	69H-7CH	10

So, if the HL pair is loaded with 5DH (5CH+1), then PRNSEQ will output the file name to the CRT. This only works because SEEK .COM does not modify the FCB, and the current extent # is normally 00H (PRNSEQ's delimiter). During file I/O, this byte is not ordinarily 00H. In another program, it may be necessary to store the file name in a separate area of RAM to preserve it.

ASCII — The Binary To Decimal Conversion Routine

For humans, converting numbers from one base to another is more direct than the computer's method. We can change 25 (base 10) to 11001 (base 2) by successive division of 25 by the radix (base) of conversion. While performing this task, we keep track of the remainders of each division. When finished, we then read the remainders in reverse order to obtain the answer:

	Remainder		
25/2	1	^	
12/2	0	R |	Result
6/2	0	e |	------
3/2	1	a |	11001
1/2	1	d |	

With practice this exercise involves minimal effort. But the computer can't perform these steps. First of all, it can't divide. It can only add and subtract. Secondly, the computer deals exclusively with binary values. Suddenly this marvelous machine doesn't seem too bright! As a result of these restrictions, the human must contrive a method to perform the mechanics of base conversion within the operational limits of the computer. Before continuing, I should note that the conversion method I have used is not of my own invention, and other routines do exist. But in its favor, ASCII is easier to understand than most of the other conversion routines.

Let's look at the task at hand. The 16 bit number in a 2 byte storage area can represent a decimal value no greater than 65,535. This means there can be no more than 6 ten-thousands, 5 thousands, and so on in our count value. So, if we can determine how many of each "positional" value (i.e. hundreds, tens, etc.) are in each stored number in RAM, we then may convert the binary value to ASCII and then print the decimal equivalent. To simulate division, we'll use repetitive subtraction and count the number of

subtractions until a "borrow" occurs. Essentially, the routine ASCII does the following:

1. Convert each decimal "positional" value to binary.
2. Subtract the "positional" value from the stored count value noting the number of subtractions.
3. When a "borrow" occurs, add one "positional" value back to the remaining difference.
4. Convert the number of subtractions to ASCII and print the decimal equivalent on the CRT.
5. Repeat steps 1–4 for each "positional" value.

These steps now form the routine ASCII:

```
ASCII   LXI     B,10000D        |       CONV    MVI     D,0FFH
        CALL    CONV            |       LP      MOV     A,L
        LXI     B,1000D         |               SUB     C
        CALL    CONV            |               MOV     L,A
        LXI     B,100D          |               MOV     A,H
        CALL    CONV            |               SBB     B
        LXI     B,10D           |               MOV     H,A
        CALL    CONV            |               INR     D
        MOV     A,L             |               JNC     LP
        ADI     30H             |               DAD     B
        CALL    CHROUT          |               MOV     A,D
        RET                     |               ADI     30H
                                |               CALL    CHROUT
                                |               RET
```

Prior to entry into ASCII, the HL pair is loaded with the value to be converted, e.g. LHLD CTR. Then the binary equivalent of the highest "positional" value is loaded into the BC pair (LXI B,10000D). Next, a call is made to the conversion routine CONV. It is here that the repetitive subtractions and ASCII conversion takes place.

First, we'll use the D register to keep track of the # of subtractions. So load it with a -1 (0FFH). The reason for this will be explained later. Then a 16 bit subtraction is performed (similar to the process used in the SIZE routine). After each subtraction, the D register is incremented, and a check is made to determine if a "borrow" occurred. If not, the routine loops back to the LP label and continues subtracting until a "borrow" takes place. It is important to note that when the "borrow" occurs it does so 1 subtraction after the true count in register D. For example, if 10,000 is repetitively subtracted from 30,000, a "borrow" takes place on the 4th subtraction. This situation is corrected by initially loading the D register with a -1 (0FFH). Since we've subtracted one too many 10,000s, we must add this "positional" value to the count left in the HL pair. The DAD B instruction accomplishes this task. Now, the D register contains the true # of 10,000s in our count value. Converting this number to ASCII is done easily. Just add 30H to it.

This is also known as adding the "ASCII bias". If you look at an ASCII table, you'll notice that the numbers 0–9 correspond to ASCII values 30H–39H. Hence, the term ASCII bias (or constant offset). Finally, the CHROUT routine is used to print the "positional" count value on the CRT. This process continues with each "positional" value until only the "units" remain. They are then converted to ASCII and printed via CHROUT. The conversion routine is subsequently invoked for each stored value as it is needed. Then, at the end of the program, SEEK displays a program end message.

Now, take a few minutes to review Listing #1 and see that it compares to your version of SEEK.ASM. If not, you may want to make the necessary changes to your version before proceeding.

Steps 4 And 5 — Review And Test

Having completed each module and incorporating them into a single program, it will be necessary to analyze the overall program to resolve potential conflicts between its modules. Generally, this is accomplished by assembling the program and testing its operation. With this purpose in mind, create a few test files with known lines, words, etc. Then run SEEK.COM and check the results. Wherever the answers don't correspond to the correct values, you must try to determine the cause of the problem and correct it. For me, this process resulted in many minor, and two major revisions to the original version of SEEK.ASM. Listing #1 shows the current end product (version 1.2). One of the more interesting aspects of this review and testing process is trying to compress the program's overall size making it more efficient. To this end, I'll propose some additional changes to SEEK.ASM for your consideration:

1. Remove duplicate instructions — In the OPNRD routine there are 4 error messages that print via a CALL PRNSEQ, JMP EPGM sequence. If a label (e.g. ENDX) were added to the CALL PRNSEQ instruction prior to the EPGM label, then each of the 2 previous instructions could be replaced with one — JMP ENDX. Thus saving 12 bytes of RAM.

2. In the ASCII routine there are 2 occurrences of the sequence CALL CHROUT, RET. These instructions may be replaced by JMP CHROUT. The reason for this is that the initial CALL ASCII instruction places the address of the next instruction onto the stack (SP reg.). Then, the JMP CHROUT starts the CHROUT routine. At its end, the RET instruction POPs the previously stored address off of the stack and into the PC register (program counter). The net effect of this change is to save 2 more bytes of ASM code space.

3. If you want a challenge, try incorporating another method of binary-to-decimal conversion for the ASCII routine. Listing #2 is one more variation of this process. As you can see, this method, although more compact, is not easily understood. In any event, my thanks to Pat Swayne for providing me with this useful routine.

These examples should give you some idea of the kind of items you may find as you continue to study your ASM program. Some of these changes may not be apparent to you at first, but the more time spent programming, the better you'll become at this.

```
; This is a tricky routine that makes use of the Stack
; to store decoded digits
; Entry reqmts: the # to be decoded is in the HL pair
; 10/8/85 - From P  Swayne (HUG)
;
DECOUT  PUSH B!PUSH D!PUSH H  ; Save registers
        LXI     B,-10         ; Radix for conversion
        LXI     D,-1
DX      DAD     B             ; Divide by 10
        INX     D
        JC      DX
        LXI     B,10
        DAD     B
        XCHG                  ; DE= digit, HL= number/10
        MOV     A,H
        ORA     L             ; Done ?
        CNZ     DECOUT        ; Call recursively until done
        MOV     A,E           ; Get character to print
        ADI     30H           ; Add ASCII bias
        CALL    SCOUT         ; Print it
        POP H!POP D!POP B     ; Restore registers
        RET

SCOUT   MOV     E,A
        MVI     C,2
        CALL    BDOS
```

Listing 2

Conclusion

Over the past few months, we've covered a lot of information. We took an idea and transformed it into a working ASM program. To do this, we needed to understand how CP/M operates (memory mapping, FCB structure, OS functions, etc.). Fortunately for Heath computer owners, this information is readily available. During each step of the transformation process, I have tried to provide the reader with an understanding of why the ASM routines were developed in a particular way. Further, we've looked at examples of error trapping, message handling, and code compression techniques.

Now that you've got SEEK.COM, I hope you'll use it on your term papers, book reports, etc. And most importantly, you may find yourself wanting more information from SEEK.COM. In this case, this series should offer you a foundation for these additional ASM programming needs. Say . . . what about adding a module that computes the average # of words per sentence? Good luck, and have fun.

*

Two Parallel Interfaces For The H-89 (One For The H-8?)

Pierre D. Olivier, III
2019 Marengo Street
New Orleans, LA 70115

I purchased one of the last H-89 computers sold in the Jacksonville area, when it was on the Heath-advertised closeout. At the time I ordered it (there were none in stock), I also ordered the Z-89-11 multiport interface card. The lack of a CP/M-supported parallel interface for my old H-8 always irritated me. About a week later, I was informed that the multiport card was no longer available, but that a parallel interface could be purchased from FBE Research. The store manager said he would order it for me and deduct the cost from the prepaid amount on the multiport card.

Just about the time I finished assembling my H-89, the H89PIP interface arrived and I picked it up. It is a small card (it is not physically held by the accessory mounting bracket, but rather just "hangs on" by the friction of the connector springs), and is designed to install on the right-hand expansion bus. Instructions and materials were provided to place it on the left-hand bus if one did not mind doing some soldering on the CPU board. I hesitate to do this, because "permanent" or irreversible modifications might interfere with some other addition I may wish to make "down the road". There are also complications if one has the Heath memory expansion installed. This is because the memory expansion has a ribbon jumper to the CPU board which is too short to allow the board to be moved over to the next slot. This COULD be overcome by using a longer ribbon cable, but that increases the chances of erratic operation induced (literally) by unshielded data lines. At any rate, I didn't need serial capabilities on a ship (we don't have a telephone line in my office), so modem use is not possible even when in port. I removed the serial board and installed the H89PIP in the newly-vacant right-hand slot.

The H89PIP comes with a dual-format disk containing HDOS and CP/M driver software. PPATCH.COM and PPATCH.ASM are provided for CP/M, and two HDOS drivers (GP.DVD and GP2 .DVD). I was using CP/M exclusively, and cannot comment on the HDOS software. PPATCH is intended to be run upon boot-up, and modifies the BIOS to recognize the 8255 PPI chip which is the heart of the board. The only way to insure that it doesn't get forgotten (and forgetting to run it results in a complete lock-up when trying to print) is to make it the automatic command line using CONFIGUR. My practice was to run the public-domain program SD on boot-up, but the loss of that facility was a small price to pay for parallel interface capabilities. CP/M 2.2.04 will not properly initialize the 8255, since the H89PIP is using a different port on the 8255, and the handshaking lines are not the same as those used in the Z-89-11. An attempted incorporation of PPATCH.ASM into the BIOS of 2.2.03 failed, but I'm sure that a better assembly language programmer (are you reading this, Pat Swayne?) could achieve success where I blew it. Everything worked "as advertised", but I missed SD.COM, and I expected to need the serial port for a modem after my approaching discharge from the U.S. Navy.

Sometime later, I noticed an advertisement in REMark for a parallel interface designed to mount on the serial board. (I could operate a modem after all!) McGaffey Engineering sells the Quick-P interface for the H-89. It is somewhat less in cost than the H89PIP, and requires NO software modification other than "setting" the baud rate (actually, avoiding 9600 baud, which causes a form-feed on boot-up). Great! I get SD back as a bonus.

The Quick-P is two boards connected by a hard-wired ribbon cable. One board is the adapter board which plugs into the socket vacated by any one of the three 8250 UARTs. The other (main) board mounts on a right-angle bracket (included), which in turn is bolted to the CPU board along with the accessory mounting bracket. Also provided are the necessary nuts, bolts, and washers. On some older serial I/O boards (lacking U610) a jumper is required, and solder, jumper wire, and instructions for the procedure are included. The main board has a second ribbon cable terminating in a female DB-25 connector, which replaces the serial port DB-25 on the back plate. The pin-out of the new DB-25 is compatible with the newer IBM-style parallel connector implementation.

I installed the Quick-P parallel interface in my H-89 and was satisfied that it met the advertised claims, but I encountered a

problem in installation that I feel might be solved in future production. I had to space the main board out almost 3/8" from the right-angle mounting bracket using standoffs and longer 6-32 screws than those provided, since there was not sufficient clearance from the ribbon cable connector coming off the adapter board.

I can see two ways of handling this problem in the future. One would be to include longer (3/4") 6-32 screws and 3/8" spacers in the package. A simpler "fix" from a production standpoint (and cheaper in the long run) would be to make the short leg of the right-angle mounting bracket 3/8" longer. If the long leg is shortened by that amount, there is still enough clearance from the backplane to accommodate the main board, so the unbent aluminum stock could be the same length. The only manufacturing change required would be in the positioning of the flat bar on the forming machine.

If you refer back to the title of this article, you'll see that I mentioned a parallel interface for the H-8. The Quick-P is it. The H-8 serial board is functionally identical to the one in the H-89, except that it has more extensive on-board address decoding and an extra port for the terminal. There is ample space to the left of the cards in which to mount the right-angle bracket, and the ribbon cable is long enough to reach one of the left-most ports on the serial board. I didn't have the H-8 available to actually test this out (I sold it in order to pay for the H-89, which is easier to secure for rough weather than the H-8 octopus would be), but the only problem I can see would be determining the H-8 equivalent to U610 in order to tell if the jumper needs to be added.

I was satisfied with the Quick-P modification in all respects excluding the clearance problem, which took a matter of minutes and junk-box parts to correct. I am sending a copy of this review to McGaffey Engineering for their consideration in future production, and maybe they'll change the mounting bracket.

The Quick-P was a little more time-consuming to install, due to the extra hardware involved, but this same hardware provided a degree of vibration-security not available with the H89PIP (a big consideration on a ship that "rocks and rolls" like mine does. In considering the relative merits of the two interfaces, I find the Quick-P to be more "user-friendly", since no extra software is required. Oh, yes, one negative aspect to the Quick-P interface which I forgot to mention — printer interrupts are not supported and must be jumpered "off" on the serial board. This means you can't use the REACH modem program. MODEM7 doesn't require interrupts, so I would have no problems in that regard, but if you use REACH, be forewarned.

I can't quote prices here, since all the invoices are in New Orleans at my permanent residence, and this is being written on board ship in Jacksonville, Florida. The prices should be available in advertisements in back issues of REMark and Sextant magazines, since both companies advertise in those two magazines. I do know that either interface will cost less than $100.00, with the Quick-P the less expensive of the two.

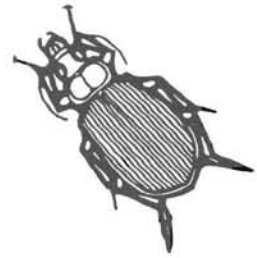
I recommend the Quick-P interface overall. The price makes it more attractive to the consumer, a plus when trying to convince the purchasing manager (spouse) that you "really need" the item. It requires no driver software or bios modifications under CP/M (I didn't have or desire HDOS). The FBE Research product does. Best of all, the Quick-P does not require the elimination of two other ports. Documentation on both is clearly written, and a novice should have no trouble performing either installation, unless you have a problem with soldering. If you are a Heath-Zenith user, even if you didn't build the machine yourself, you should know someone in your local H.U.G. who can help. The HUGgies I've known in Virginia and Florida are generally open to rendering assistance in this sort of problem.

By the way, if you've noticed that I refer to my H-89 in the past tense, it is because I sold it in order to pay for my new pride and joy — an H-151 with 640K RAM and RGB monitor. I enjoyed using my 8-bit machines, but felt it was time to move up to bigger and better things. Unfortunately, I can only afford to support one computer at a time. I didn't see any reason to withhold this information which may help some of the 8-bit enthusiasts just because I wasn't using an H-8 or H-89 anymore.

*

DEBUG

Jerry Furst
17049 Vista Bluff
San Antonio, TX 78247



This article is intended to acquaint you with some of the powerful features of DEBUG, and some simple 'tricks' to quickly and easily patch programs for use with ZPC (the PC emulator from HUG). This will be by no means a complete summary of all the commands of DEBUG, rather it is intended to assist with patching programs.

I have had over a 90% success rate with patching PC programs for use with ZPC. I refer specifically to the Public Domain, and 'Share-ware' type programs typically found on BBS's. The professional software packages I would leave up to experts like Pat Swayne, with one notable exception, I was able to patch TURBO Pascal for the PC (version 3.01A) (with a little help from Pat).

DEBUG in itself is a very handy utility. Typically, we will be able to tell if a file can be patched for use by ZPC in a matter of a couple of minutes using some of the functions of DEBUG.

First some rules. ALWAYS change programs with an .EXE extension to a .BIN (or other) extension before using DEBUG to patch the program, the reason is that DEBUG will strip the program header from an .EXE file and discard it. This makes for unusable files if you attempt to save the program. Also, DEBUG needs HEX numbers for commands, so do your conversions PRIOR to entering DEBUG. Example: doing a dir you may see the following:

```
TESTFILE.TST 40997 12-13-85 04:01:00...
```

The 40997 is the file size in DECIMAL bytes, we convert this to the hex value A025 for our use in patching the file. Keep a pad and a pencil handy when attempting to do these patches, you need to keep notes! Genie, with a notepad and calculator ready in memory, are also good substitutes for a pad and pencil, plus calculator. We have some beginning rules, so now we enter DEBUG.

```
A> DEBUG <CR> ***We will assume a <cr> from now on after
                an end of line ***
-  <--This is the debug prompt; next we (n)ame the file
-ntestfile.tst
-nb:testfile.tst  <--We could also specify a drive
-l <--Then we load the file into memory
- <--And we get the debug prompt again
```

Now for some more rules while in DEBUG. If you use the (t)race command, do not attempt to make your patches to the traced program. They have a habit of growing and modifying things and this will make for nasty results. Instead, use the (q)uit command, come back into DEBUG and make your patches to an un-traced program, then save it.

One of the handiest commands in DEBUG is the (s)earch command. You can search for a string of bytes within a given range of memory (that's what makes patches so easy, when we know what we are looking for) using this command. The correct syntax for a search command is 's' followed by a start address, space, end address, space, search bytes, each separated by a comma, or space . . . like this:

```
s100 a025 ec a8 01
```

This command told DEBUG to start searching at the current Segment as defined by the DS register at offset 100 and search for the bytes ec,a8,01 until offset address a025. This brings up another point, and a rule while using DEBUG, that being that we are examining computer memory, and computers store the MSB (most significant byte) first, and the LSB (least significant byte) second.

To illustrate, we are going to be looking for some PC video ports that play havoc with a Z-100, they are 16-bit ports and are in the range of 03D4 to 03DA, but if we did an s100 a025 03 DA, we would never find them. Why, because 03 is the MSB and DA is the LSB, and their order in memory is reversed for search purposes. The command s100 A025 DA 03 would find our port. See how that works?!

One more point about the search command. I had said earlier that the search command starts at the segment defined by the DS register, but it doesn't have to. If we had a program that was greater than 64K in length and we needed to patch it, we would have to have a way to search the memory above the 1st 64K. We can do that one of two ways, either we can modify the DS register, or we can modify the search segment. I prefer the former.

This also gets us into a bug in DEBUG (excuse the pun), that is that DEBUG expects programs to start at memory address 100 Hex, the command s000 ffff ec a8 01 will generate an error from DEBUG. There is a simple way around it, do two searches. First, do an s000 100 ec a8 01, then do an s100 ffff ec a8 01 and everything will work fine.

To see how we can check the next 64K of memory, I will need to introduce another command, the (r)egister command. If we just type an 'r', then all the registers and their values will be displayed, but if we elect to view just a single register 'rds', then we go into a modify mode. The 'rds' command might come back with a -1a90 for the value of ds. If we add 1000 to that number, then we will be into the next 64K of memory! "Wait a minute!," you say, "how can you add 1000 and get back 64K?!" Let's review how we address memory for a moment.

```
  1A90  = Segment Address
+  0100 = Offset Address
  1AA00 = Absolute memory address
```

So by using the above example, if we add 1000 to 1A90 (DS contents) we get 2A90 + 0100 = 2AA00 - 1AA00 = 10000 hex or 65536 (64K)!

The other way is to add the segment address increase to the 's' command.

```
-s2a90:100 ffff ec a8 01
```

The problem with this is that the DS register still contains 1A90. So, if we would accidentiy forget to add the segment address increase to the commands that we use to alter memory, we would be patching the wrong section of memory. If this boo-boo happens, then let me introduce you to the (q)uit command.

The quit command will do just that, not make any changes, and give you the chance to start over. For this reason alone, I recommend using the rds command. One more point to bring up about modifying the ds and cs registers, before you save your patched file, be sure to return these addresses to their original values, to insure that you save the correct file size, etc.

Let's move into the memory modification commands, there are two that we will discuss (though there are more), they are the (e)nter and the (f)ill command. The enter or examine command will allow you to examine a single byte at a time and modify it if you so desire.

Example:

```
-e1f4 <--we wish to examine this byte
 EE.  <--we want to change this byte so we type the
        change
 EE.90 CD   <--the space bar shows the next byte
 EE.90 CD.  61.  <--and another space bar
- <--RETURN exits the command
```

The fill command is similar to the enter command except that we can fill multiple bytes with the same character;

```
-f1f4,1fe,90 <--would change those memory locations
                (DS holds SEG) to the new value 90
-f1f4,1f4,90 <--this would change only one byte
-f1f4,1f5,90,91 <--this would change two bytes to
                  90 & 91 and so on
```

I usually prefer to use the 'e' command over the 'f' command as I have more control over what I change. Also, if I suddenly found the wrong code, I would have a chance to correct my mistake before I made the blunder.

The next command we want to talk about is the (w)rite command. The write command will write the file from memory back to disk, with any changes that we may have made to it in memory. It writes back to the same file that we had originally called up (no back-up is made) so to protect myself I do this:

```
A> copy testfile.tst testfile.old
```

That way I always have an original copy of the program before I start making any changes. The write command will start writing back to disk the file that starts at 100H (CS contains Segment) the number of bytes in BX and CX. If any of those registers were altered prior to write, you wouldn't get back what you started with. Best to start with a fresh copy of the program (that hasn't been traced) to make your patches and save your new program.

The next two commands I want to go over are the (d)ump and the (u)nassemble commands. Both of these commands require some expertise in assembly language programming, which we won't get into now, but they can be useful.

The 'd' command or dump will give you a hex dump of the memory address you specify. You will get 128 bytes of hexadecimal whatsamacallit. This command is useful only if you are looking for ASCII characters, because everything else will be just garbage to you. The syntax is -DXXXX where the starting address is XXXX (DS contains segment address). You can also continue hitting the 'd' command and see the next 128 bytes if you wish.

The 'u' or unassemble command is much more powerful in that you will be shown 20 hex (32) bytes of assembly language code. Again, this requires some expertise in assembly language, but this can also be useful to the novice patcher as we will need to verify that what we found in our search is what we really wanted.

In a previous example we talked about searching for some video ports in a PC program that were in the range of 03D4 to 03DA, and suppose for a moment, that we had searched a program and found that there were several of these bytes, you don't think for a moment that with all that hex lying around that those characters would only be used for video do you?! Of course not, that would be too easy. Those found bytes could be anything from a memory address to two separate pieces of code! We will use the 'u' command to verify what we have found is correct.

One note of caution here, NEVER start your disassembly of code exactly where the search says it was, otherwise you'll be fooled. The unassemble command jumps into the middle of a program and tries to disassemble the code in that particular memory location to it's correct syntax, and it can take as much as 12 bytes to get it straightened out, always start 10 to 15 bytes ahead of the address you wish to unassemble. If the program says it found ec a8 01 at memory location 5bf5, start your disassembly at 5be0, just to be safe. The correct syntax here would be -u5be0.

This leads us up to the last two commands that we are going to talk about. These are the (T)race command and the (P) special trace command. These commands are very powerful as they allow you to single step through a program, but they also require a fairly thorough knowledge of assembly language. The 't' command has several options in the trace mode. You can

```
-t <--single trace or
-t10 <--trace multiple (10 hex or 16 times or
       whatever) or even
-t=2a0 <--specify the memory address to start the trace
```

The trace command will display the flag and register contents, the memory locations and even the next command to be processed. Tracing a program through execution though can be very tedious. Consider this little bit of code

```
1f0   mov cx,5f00
1f6   mov al,00
1fa   loop 1f6
```

If we were to trace this section of program, we would have to trace it 24320 times! This is a little timing loop, basically what happens is that we move 5f00 (24320) into CX (this is our count register), next we move 00 to the AL register. The loop command decrements CX, then jumps back to the memory address specified. Oh, you could type t5f00 and come back next tuesday to see if it was done, but there is an easier way, the (p) command. This special trace command can be used on LOOPs, CALLs, INT's, and decrementing type instructions. When we encountered the first loop command we could hit the -p instead of the -t and have the machine run it at machine speed. Consider another helpful example of (p) vs. (t).

```
1f0   mov dl,35
1f4   mov ah,02
int   21
```

This is an interrupt sequence. All that it is intended to do is print the number "5" on the screen. However, the interrupt handler (21 in this case) is several hundred bytes of code, and ultimately ends up in the machine ROM to execute this simple instruction. You don't want to be tracing through your interrupt handlers to debug a program, the choice here is to use the 'p' command and just execute the handler vs. tracing it.

This concludes the discussion on DEBUG. Next, we will move into how to use what we've just learned with ZPC. This will be kind of anti-climatic as now that we know what to do, patching programs for ZPC can take as little as 2 minutes and involves just a few steps.

First of all, what are we looking for. We will be looking for video port calls, keyboard interrupts, and to a lesser extent as I have found few PC public domain programs that contain them, the Graphics Character table, and the Sound Generating calls from compiled BASIC programs. These ports and addresses have been thoroughly documented in the documentation accompanying ZPC, in the section on patching programs.

When searching for the video ports, use the port address (like: s100 a025 da 03) rather than the OUT DX,AL (single byte code EE), as there are alot more EE's in the program than there are DA 03s. Searching for three bytes of code is better still, and four bytes can almost guarantee success of finding the right code. It boils down to these 9 basic steps.

1. Make a backup of the original file (just in case).
2. Rename all .EXE programs to .BIN (or something).
3. Have a list ready of the byte code sequences you'll be searching for.
4. Convert all decimal values to HEX values first.
5. Search for 2, 3, or 4 byte sequences if possible.
6. Confirm (using 'u'nassemble) that addresses found are what need to be patched (see ZPC documentation for examples).
7. Patch the program using ('e' or 'f' commands) un(t)raced programs.
8. 'W'rite your patch to disk.
9. Rename your program to original (if changed).

That's all there is to it. I have been successful at patching over 20 PC Public Domain programs using these steps. These patches are available on the SAHUG (San Antonio Heath Users' Group) BBS (512-341-0586) and can serve as examples if you wish. It really isn't difficult once you get into it and try it. Enjoy!

*

What's Better Than Getting Into The IBM/PC Environment?

Getting Into The IBM/PC Environment At The Right Price.

And For Z100 Users, The Price Is Right

New IBM PC compatible computer systems can cost thousands of dollars. And that's just the beginning! There are those other "hidden" costs, like the probable write-off of your existing machine, operating system, related software, and add-ons.

But for Z100 users, this pain can be avoided. For around $599.00, the GEMINI EMULATOR BOARD secures your initial investments and permits a 3-mode operation for your Z100: CPM, Z-DOS, and PC-DOS.

Think it over. We think you'll agree ...for Z100 users requiring IBM compatibility *the price is right.*

Get full details at your nearest Heath/Zenith Electronics Centre, or telephone our head office at (604) 274-1283.

GEMINI technology corp.

13227 N.E. 20th Street, Bellevue, Washington 98005

A Non-Blinking Cursor For Your PC

Pat Swayne
HUG Software Engineer

It has been reported (I even said it myself) in this magazine and in other magazines that you cannot have a non-blinking cursor on an IBM PC computer or a hardware compatible clone of one, such as the Heath/Zenith Z-100 PC series. It might be possible via a hardware modification to your video board, but it is not really possible to stop the cursor from blinking via any software commands. However, on a Heath/Zenith machine at least, it is possible to "fake" it.

One of the ways in which Heath/Zenith PCs really are superior to the "real thing" and other clones is that you can write to video memory at any time without harming the display. On a "real" PC, you must test for horizontal or vertical retrace time and write to video memory only during that time, or else there will be "snow" on the screen.

You can fake a non-blinking cursor on a PC by turning the real cursor off, and then writing a block of white video to the cursor position on the screen. Some word processing programs actually do this, but in order for it to be done outside a program, so that it would work with any program you ran, you would have to update the "cursor" constantly. If you had to test for retrace time before you updated the cursor, it would degrade the performance of the machine.

I have written a small program that can provide a non-blinking block cursor on a Heath/Zenith PC computer. It will not work on an IBM PC, not only because it causes considerable snow on the screen, but because of a scrolling problem that I plan to discuss in a future article. The program is called NOBLINK.COM, and you can make it by typing in and running the following BASIC program.

```
10 REM THIS PROGRAM CREATES NOBLINK.COM
20 DEFINT A-I:OPEN "O",1,"NOBLINK.COM
30 S=0:S1 = 39141 'FOR I=1 TO 410
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 233,215,0,0,0,0,0,0,0,0
80 DATA 0,0,0,0,0,232,5,0,46,255
90 DATA 46,7,1,46,128,62,11,1,0,116
100 DATA 1,195,80,83,81,82,30,184,64,0
110 DATA 142,216,138,46,73,0,186,180,3,128
120 DATA 253,7,116,7,178,212,128,253,4,115
130 DATA 106,176,10,238,66,176,32,238,138,14
140 DATA 97,0,160,98,0,152,187,80,0,3
150 DATA 216,139,7,138,208,134,196,182,80,246
160 DATA 230,50,246,3,208,209,226,66,184,0
170 DATA 184,128,253,7,117,3,184,0,176,142
180 DATA 216,46,139,30,12,1,176,120,58,7
190 DATA 116,18,129,235,160,0,114,4,58,7
200 DATA 116,8,129,195,64,1,58,7,117,6
210 DATA 46,160,14,1,136,7,246,193,32,117
220 DATA 16,139,218,138,7,46,162,14,1,46
230 DATA 137,30,12,1,198,7,120,31,90,89
240 DATA 91,88,195,78,66,128,252,110,116,12
250 DATA 156,46,255,30,3,1,250,232,89,255
260 DATA 251,207,185,64,0,142,217,139,14,96
270 DATA 0,199,6,96,0,7,32,232,69,255
280 DATA 46,162,11,1,180,1,235,218,161,93
290 DATA 0,60,32,117,67,51,192,142,216,190
300 DATA 64,0,196,60,38,129,125,254,78,66
310 DATA 116,72,14,7,86,191,3,1,252,165
320 DATA 165,94,199,4,175,1,140,76,2,190
330 DATA 112,0,86,191,7,1,165,165,94,250
340 DATA 199,4,15,1,140,76,2,251,14,31
350 DATA 186,69,2,180,9,205,33,186,218,1
360 DATA 205,39,50,219,61,79,78,116,7,61
370 DATA 79,70,117,8,254,195,138,195,180,110
380 DATA 205,16,205,32,14,31,186,122,2,180
390 DATA 9,205,33,205,32,13,10,78,79,66
400 DATA 76,73,78,75,32,115,111,108,105,100
410 DATA 32,99,117,114,115,111,114,32,103,101
420 DATA 110,101,114,97,116,111,114,32,105,115
430 DATA 32,110,111,119,32,105,110,115,116,97
440 DATA 108,108,101,100,46,13,10,36,13,10
450 DATA 78,79,66,76,73,78,75,32,105,115
460 DATA 32,97,108,114,101,97,100,121,32,105
470 DATA 110,115,116,97,108,108,101,100,46,36
```

The assembly source code for NOBLINK follows this article. To use NOBLINK, copy it to your system disk, and enter

```
NOBLINK
```

at the system prompt, and hit Return. NOBLINK will load itself into memory and remain there until you reset and reboot. Once you have loaded NOBLINK, you can restore your normal cursor by entering

```
NOBLINK OFF
```

and you can turn the non-blinking cursor back on by entering

```
NOBLINK ON
```

Most programs will work properly with the non-blinking cursor, but Lotus 1-2-3 and Symphony do not, so you should restore your normal cursor before running those programs.

The screen attribute within the cursor block is set to gray on a white background. If you move the cursor over a character on the screen, the character will appear as gray inside the white block. You may have to adjust the black level or contrast control on a monochrome monitor to see the character inside the block. You could change the cursor colors in the assembly source file if you wish, but you should use a combination that is not likely to be used by your programs.

Assembly Source Code For NOBLINK.COM

```
        PAGE    ,132
;       THIS PROGRAM PRODUCES A NON-BLINKING BLOCK
;       CURSOR ON A H/Z PC COMPUTER
;       (AND THEY SAID IT COULDN'T BE DONE!)
;
;       BY P. SWAYNE, HUG SOFTWARE ENGINEER  08-AUG-86
;       COPYRIGHT (C) 1986 BY HEATH/ZENITH USERS' GROUP

CODE    SEGMENT
        ASSUME  CS:CODE,DS:CODE,ES:CODE,SS:CODE

BLACK   EQU     0               ;DEFINE BLACK
DKGRY   EQU     8               ;AND DARK GRAY
CHCOL   EQU     DKGRY           ;CHARACTER COLOR IN CURSOR

;       STORAGE LOCATIONS IN RAM (MOST IN BIOS RAM)

        ORG     49H
MODE    LABEL   BYTE            ;VIDEO MODE
        ORG     50H
CURPOS  LABEL   WORD            ;CURSOR POSITION
        ORG     5CH
FCB     LABEL   WORD            ;FCB LOCATION
        ORG     60H
CURTYP  LABEL   WORD            ;CURSOR TYPE
        ORG     62H
VPAGE   LABEL   BYTE            ;VIDEO PAGE

        ORG     100H

START:  JMP     SETUP           ;SET UP THE PROGRAM

INT10V  DD      0               ;SCREEN INTERRUPT VECTOR
INT1CV  DD      0               ;TIMER INTERRUPT VECTOR
BLFLG   DB      0               ;BLINK OFF/ON FLAG
OLDCP   DW      0               ;OLD CURSOR POSITION
OLDAT   DB      0               ;OLD ATTRIBUTE

;       TIMER INTERRUPT PROCESSOR   DURING TIMER
;       INTERRUPTS, WE TURN OF THE REAL CURSOR, AND
;       SET THE ATTRIBUTE OF THE CURSOR POSITION TO
;       GRAY ON WHITE, SO THAT A WHITE BLOCK APPEARS
;       THERE

TIMER:  CALL    CURSOR          ;MAKE CURSOR
        JMP     CS:DWORD PTR INT1CV ;TO OTHER PROCESSES

;       CURSOR UPDATE PROCESSOR

CURSOR: CMP     CS:BLFLG,0      ;BLINK OFF?
        JZ      BLNKOFF         ;YES
        RET                     ;ELSE, EXIT
BLNKOFF:PUSH    AX              ;SAVE SOME REGISTERS
        PUSH    BX
        PUSH    CX
        PUSH    DX
        PUSH    DS
        MOV     AX,40H
        MOV     DS,AX           ;POINT TO BIOS RAM
        MOV     CH,MODE         ;GET VIDEO MODE
        MOV     DX,3B4H         ;MONO  CRT ADDRESS PORT
        CMP     CH,7            ;MONOCHROME CARD MODE?
        JZ      MODEOK          ;THAT'S OK
        MOV     DL,0D4H         ;CRT ADDRESS PORT (3D4)
        CMP     CH,4            ;IN A GRAPHIC MODE?
        JNB     CUREX           ;IF SO, EXIT
MODEOK: MOV     AL,10           ;CURSOR START REGISTER
        OUT     DX,AL           ;SELECT IT
        INC     DX              ;CRT DATA PORT
        MOV     AL,20H
        OUT     DX,AL           ;TURN OFF REAL CURSOR
        MOV     CL,BYTE PTR CURTYP+1 ;GET CURSOR TYPE HIGH
        MOV     AL,VPAGE        ;GET VIDEO PAGE
        CBW                     ;MAKE IT A WORD
        MOV     BX,OFFSET CURPOS
        ADD     BX,AX           ;POINT TO CURSOR POS.
        MOV     AX,[BX]         ;GET IT
        MOV     DL,AL           ;SAVE COLUMN
        XCHG    AL,AH           ;ROW TO AL
        MOV     DH,80           ;80 COLUMNS/ROW
        MUL     DH              ;GET ROW ADDRESS
        XOR     DH,DH           ;DX = COLUMN
        ADD     DX,AX           ;DX = ABS  CURSOR POS.
        SHL     DX,1            ;DX = CURSOR ADDRESS
        INC     DX              ;MOVE TO ATTRIBUTE BYTE
        MOV     AX,0B800H       ;ASSUME CGA
        CMP     CH,7            ;MONOCHROME CARD?
        JNZ     CGA             ;NO, CGA
        MOV     AX,0B000H       ;ELSE USE MONOCHROME ADDRESS
CGA:    MOV     DS,AX           ;POINT TO VIDEO RAM
        MOV     BX,CS:OLDCP     ;GET OLD CURSOR POSITION
        MOV     AL,70H+CHCOL    ;GET CURSOR ATTRIBUTE
        CMP     AL,[BX]         ;CURSOR ON?
        JZ      TCOFF           ;YES, TURN OFF
        SUB     BX,160          ;TRY LINE ABOVE
        JB      LINE1           ;ON LINE 1
        CMP     AL,[BX]         ;CHECK CHARACTER
        JZ      TCOFF
LINE1:  ADD     BX,160*2        ;MOVE TO LINE BELOW
        CMP     AL,[BX]         ;CURSOR ON?
        JNZ     NOCUR           ;NO
TCOFF:  MOV     AL,CS:OLDAT     ;GET OLD ATTRIBUTE
        MOV     [BX],AL         ;TURN OFF CURSOR
NOCUR:  TEST    CL,20H          ;CURSOR OFF?
        JNZ     CUREX           ;IF SO, EXIT
        MOV     BX,DX           ;GET NEW CURSOR POS
        MOV     AL,[BX]         ;GET ATTRIBUTE
        MOV     CS:OLDAT,AL     ;SAVE IT
        MOV     CS:OLDCP,BX     ;AND OLD POSITION
        MOV     BYTE PTR [BX],70H+CHCOL ;TURN CURSOR ON
CUREX:  POP     DS              ;RESTORE REGISTERS
        POP     DX
        POP     CX
        POP     BX
        POP     AX
        RET

;       SCREEN INTERRUPT PROCESSOR   UPDATE CURSOR AFTER
;       ANY SCREEN CALL

        DB      'NB'            ;IDENTIFIER
SCREEN: CMP     AH,110          ;BLINK CONTROL?
        JZ      SETBLNK         ;IF SO, DO IT
SETCUR: PUSHF                   ;PREPARE FOR IRET
        CALL    CS:DWORD PTR INT10V ;PERFORM SCREEN FUNC
        CLI
        CALL    CURSOR          ;UPDATE CURSOR
```

```
        STI
SCREX:  IRET
SETBLNK:MOV     CX,40H
        MOV     DS,CX           ;ELSE POINT TO BIOS RAM
        MOV     CX,CURTYP       ;GET CURSOR TYPE
        MOV     CURTYP,2007H    ;ENSURE BLOCK CURSOR IS OFF
        CALL    CURSOR
        MOV     CS:BLFLG,AL     ;SET CONDITION
        MOV     AH,1
        JMP     SETCUR          ;TURN SELECTED CURSOR ON

ENDRES:                         ;END OF RESIDENT CODE

;       SET UP TIMER VECTOR AND SCREEN VECTOR, AND LEAVE
;       PROGRAM RESIDENT

SETUP   MOV     AX,FCB+1        ;GET COMMAND ARGUMENT
        CMP     AL,' '          ;ANY GIVEN?
        JNZ     CHKARG          ;IF SO, CHECK IT
        XOR     AX,AX
        MOV     DS,AX           ;POINT TO INTERRUPT PAGE
        MOV     SI,0FFSET 10H*4 ;POINT TO SCREEN VECTOR
        LES     DI,DWORD PTR [SI] ;GET OLD VECTOR
        CMP     ES:WORD PTR -2[DI],'BN' ;NOBLINK INSTALLED?
        JZ      ITSIN           ;YES
        PUSH    CS
        POP     ES              ;ELSE, FIX ES
        PUSH    SI
        MOV     DI,OFFSET INT10V ;SAVE VECTOR HERE
        CLD
        MOVSW
        MOVSW
        POP     SI
        MOV     WORD PTR [SI],OFFSET SCREEN ;VECTOR TO US
        MOV     2[SI],CS
        MOV     SI,OFFSET 1CH*4 ;POINT TO TIMER VECTOR
        PUSH    SI
        MOV     DI,OFFSET INT1CV ;SAVE VECTOR HERE
        MOVSW                   ;SAVE VECTOR
        MOVSW
        POP     SI
        CLI
        MOV     WORD PTR [SI],OFFSET TIMER ;VECTOR TO US
        MOV     2[SI],CS
        STI
        PUSH    CS
        POP     DS              ;PUT DS HERE
        MOV     DX,OFFSET SIGNON
        MOV     AH,9
        INT     21H
        MOV     DX,OFFSET ENDRES
        INT     27H             ;LEAVE PROGRAM RESIDENT
CHKARG: XOR     BL,BL           ;ASSUME ON
        CMP     AX,'NO'         ;TURN NOBLINK ON?
        JZ      SETCND          ;SET CONDITION
        CMP     AX,'FO'         ;TURN NOBLINK OFF?
        JNZ     EXIT            ;BAD ENTRY
        INC     BL              ;NOBLINK OFF CODE
SETCND: MOV     AL,BL
        MOV     AH,110          ;SET NOBLINK CODE
        INT     10H             ;SET IT
EXIT:   INT     20H             ;EXIT TO DOS
ITSIN:  PUSH    CS
        POP     DS              ;PUT DS HERE
        MOV     DX,OFFSET INMSG
        MOV     AH,9
        INT     21H             ;SAY NOBLINK IS IN
        INT     20H

SIGNON  DB      13,10,'NOBLINK solid cursor generator is'
        DB      ' now installed.',13,10,'$'
INMSG   DB      13,10,'NOBLINK is already installed.$'
CODE    ENDS
        END     START
```

*

Z-37 Magnolia CP/M 4 MHz

Loren R. Porter, Jr.
4545 S. Mission Road #190
Tucson, AZ 85714

Those of us old timers who still have their H-89s usually have them upgraded. Mine is no exception. Recently, I decided to add a hard disk drive to it. These items have come down to the price that we had to pay for two Tandon 80 track disk drives.

The best interface and software for it seemed to be from Magnolia Microsystems. They have a board which replaces the three port serial board with the three ports and another port for the hard disk drive. The software is specifically for Corvus. However, it seems to work fine with other drives also.

I set up the interface and software according to Magnolia's instructions. The big moment arrived. I started it up. Everything went fine until I toggled my Kres 2/4 mhz board to run at 4 mhz. It wouldn't access the Z-37 or Z-17 drives. Booting from the winchester I don't need the Z-17, but I needed the Z-37 for back-up.

By paying another $400 you can get a controller from Magnolia which replaces the Z-37. There is a software 4 mhz patch for it from Quickdata. Having already spent a small fortune for my system I was hurting for money, so I looked for another way.

Kres has a KMR ROM and PMM patch which is supposed to solve the problem. However, Kres didn't respond to my order. This meant that I had to do something myself.

I don't like working at the machine level of programming, but this time it was necessary. My first surprise came when DDT would not access the Z-37 HEX module.

I decided to link the Z37.HEX and BZ37.HEX modules to BASECPM following with the CORVUS.HEX and CBOOT.HEX modules. You do this first by renaming the BASECPM to MOVCPM15, according to Magnolia's instructions. I made one change here. I renamed it MOVCPM4M, standing for a 4 MHz version of MOVCPM15.

Loading MOVCPM4M.COM with DDT I found the Z-37 portion to be at the end, address around 2880 HEX to 2B75 HEX. I first tried to put NOPS between all of the OUT PORT and IN PORT commands. I ran into two problems. Magnolia runs the module with interrupt instructions. It seems that you can't put delays between OUT PORT and IN PORT after a disable interrupt. There also needs to be something else done which requires the listing of the Z-37 module. I couldn't get it.

At this point, I was about to give up. However, after thinking about it, Kres allows you to toggle the processor into 2 MHz or 4 MHz by software. Kres supplies the listing on their software. So why not toggle the 4 MHz processor to 2 MHz just before it accesses the Z-37 drive and toggle the processor back to 4 MHz when it is finished. With a little work this was the answer.

First, I needed a toggle routine. Second, where in memory to put it. Third, where to call it from the Z-37 module.

I wrote a toggle routine after checking the Kres listing. It is included in this article. The routine simply puts the correct instructions into unused portions of memory below 0100 HEX, which is the user starting point for CP/M. I first tried putting it behind the BIOS, but the CP/M wouldn't accept it.

I found that it works by patching address 288A to CALL 0010. 288A CD, 288B 10, 288C 00. Also by patching address 2B72 to CALL 0026. 2B72 CD, 2B73 26, 2B74 00. When finished with the patch you go out of DDT with Control-C and write SAVE 52 MOVCPM4M .COM.

Following this, you assemble the TGLDRIVE.MAC file and add the additional modules to the new CP/M that you want.

Last, you place TGLDRIVE.COM on the boot section of the hard disk drive. You must boot from the hard disk drive. Then, call SETAUTO and set it to call TGLDRIVE on cold boot. Now you are in business.

If you folks are more brave and decide to include the H-17, the above procedure will cause trouble even for the H-37. The reason is that the Magnolia CP/M doesn't link modules in the same place. Therefore, yours will likely be different than mine. The following is what to do.

First, assemble the TGLDRIVE.MAC program using the MAC Assembler. If you don't have that assembler you will first need to rewrite it so you can assemble it with ASM. Place TGLDRIVE .COM on the Hard Disk Drive section that you are going to boot from.

Copy your MOVCPM15.COM file to a floppy or to another section of the Hard Disk Drive. Rename the file to MOVCPM4M .COM.

Now call up DDT.COM and the MOVCPM4M.COM file by writing,

```
DDT MOVCPM4M.COM
```

DDT will respond with:

```
DDT VERS 2.2
NEXT PC
4200 0100
```

The 4200 Hex number will likely be different with your CP/M. You must now get a number which you can save the patched MOVCPM4M.COM. To do this you subtract 100 Hex from your left number.

```
  4200 Hex
- 100 Hex
----------
  4100 Hex    Note: Your answer may be different.
```

Next, you convert this number to decimal. 4200 Hex = 16,640 Decimal. Now you divide this number by 256. 16,640/256 = 65. So, when finished with the patch I had to type Control-C and then type

```
SAVE 65 MOVCPM4M.COM
```

The 65 number will likely be different with your CP/M.

To find the H-37 Module when you have MOVCPM4M.COM up on DDT, you press the "D" button. Then look at the right of the screen. You keep looking at the right and pressing the "D" button until you see,

```
  .°..k.Z89-37
Double Density C
ontroller  2.245
```

Look to the left of the first line and you will see the starting address of the Z-37 Module. Add 388 Hex to this address or count another 904 bytes. The address you find should be the front Z-37 address to be patched. Press L_____. (The four blanks are the address you found.) The DDT should read, "_____ LDA 000D". Press S_____. The display will be as follows:

```
        Change to
S ____
  ____ 3A CD
  ____ 0D 10
  ____ 00 00
  ____ --
```

Add 672 Hex or count 1650 bytes from the address you found while pressing the "D" button. The address you find should be the back address to be patched. Press L _____ . The DDT should read:

```
____ OUT 78
____ XRA A
```

Press S_____. The display should be as follows:

```
        Change to
S ____
  ____ D3 CD
  ____ 78 26
  ____ AF 00
  ____ --
```

What you have done is to patch the Z-37 module to call for 2 MHz when going into a drive access and to call for 4 MHz when going back to the system.

To find the Z-17 address, do the same with the "D" button as described above. You start by typing, "D 0100". When you see:

```
  ..      .Z17.Ha
rd Sector contro
ller  2.240$.
```

Note the address on the left. This is the starting address of the H-17 Module. Add 121 Hex or count down 289 Bytes. This should be the front address to be patched. Press L _____ . You should see:

```
        ____ MOV A,M
        ____ ANI 03
             Change to
Press S ____
        ____ 7E  CD
        ____ E6  40
        ____ 03  00
```

Start with the address you found with the "D" button and add 5E8 Hex or count down 1512 bytes. This should be the back address to be patched. Press L_____. You should see:

```
        ____ MVI M,00
        ____ RET
             Change to
Press S ____
        ____ 36  C3
        ____ 00  4C
        ____ C9  00
```

The H-17 Module will now do the same thing as the H-37 Module does. Last press Control-C and save the new MOVCPM4M .COM.

To use the new CP/M, you call a MOVCPM4M as stated in the Magnolia Manual followed by SYSGEN. You put it onto the drive you are going to boot from. This must become drive "A" if it is not so already. Then you press "SHIFT RESET" and BOOT. You must at this point write, "TGLDRIVE" in order to access a floppy disk drive. It is a good idea to use the SETAUTO program so you don't have to write, "TGLDRIVE" every time you do a future cold boot. If you are not operating at 4 MHz as soon as you access a disk drive, you will be. Save your MOVCPM15.COM in case you want to operate at 2 MHz.

When you are up on the new CP/M if you call up DDT and type, "L 0010", you can see what TGLDRIVE has done. The display will be:

```
0010 ?? = D9
0011 ?? = 08
0012 MVI A,00
0014 OUT 74
0016 ?? = 08
0017 ?? = D9
0018 LDA 00D0
001B RET
```

Address 0010 says to exchange registers BC, DE, and HL with registers BC', DE', and HL', a very powerful Z-80 instruction. Address 0011 says to exchange registers A and Condition Code with A' and Condition Code'. Address 0012 places the CPU speed toggle code into register A. It is 00 for 2 MHz and 04 for 4 MHz. 74 is the CPU speed change port. Address 0018 puts the instruction back into what you patched out of the front Z-37 module patch. The other three addresses start at 0026, 0040, and 004C if you would like to examine them.

If you would like this done with your Magnolia CP/M and don't feel up to doing it yourself please feel free to contact me.

```
;*****************************************************
;*                                                   *
;* TGLDRIVE -- PROGRAM TO SET 2 MHZ & 4 MHZ CPU SPEED *
;*             CHANGE INTO LOWER MEMORY              *
;*              FOR H-89 WITH H-37 USING             *
;*              MAGNOLIA CPM                         *
;*              BY. LORAN R. PORTER JR. CET          *
;*                                                   *
;*****************************************************
;
BOOT    EQU     000H            ; SYSTEM CALL

        ASEG
        ORG     0100H

START:  MVI     A,0D3H          ; OUT PORT
        STA     0014H
        STA     002AH
        STA     002EH
        STA     0044H
        STA     0050H
        MVI     A,078H          ; ALSO PORT 78
        STA     002FH

        MVI     A,0C9H          ; RETURN
        STA     001BH
        STA     0031H
        STA     004BH
        STA     0056H

        MVI     A,074H          ; PORT 74
        STA     0015H
        STA     002BH
        STA     0045H
        STA     0051H

        MVI     A,000H          ; ZERO -- BIT FOR 2 MHZ
                                  TOGGLE
        STA     0013H
        STA     0043H

        MVI     A,004H          ; FOUR -- BIT FOR 4 MHZ
                                  TOGGLE
        STA     0029H
        STA     004FH

        MVI     A,0D9H          ; EXCHANGE BC,DE,HL
                                  WITH BC',DE',HL'
        STA     0010H
        STA     0017H
        STA     0026H
        STA     002DH
        STA     0040H
        STA     0047H
        STA     004CH
        STA     0053H

        MVI     A,008H          ; EXCHANGE "A" AND "F"
                                  WITH "A" AND "F" PRIME
;                                 -- Z80 ONLY --
        STA     0011H
        STA     0016H
        STA     0027H
        STA     002CH
        STA     0041H
        STA     0046H
        STA     004DH
        STA     0052H

        MVI     A,03EH          ; MOVE DATA INTO
;                                 REGISTER "A"
        STA     0012H
        STA     0028H
        STA     0042H
        STA     004EH

        MVI     A,03AH          ; LDA 00D0
        STA     0018H
        MVI     A,0DH
        STA     0019H
        MVI     A,00
        STA     001AH

        MVI     A,0AFH          ; CLEAR REGISTER "A"
        STA     0030H

        MVI     A,07EH          ; MOV A,M
        STA     0048H
        MVI     A,0E6H          ; ANI
        STA     0049H
        MVI     A,03H           ; 03
        STA     004AH

        MVI     A,036H          ; MVI M,00
        STA     0054H
        MVI     A,00
        STA     0055H

        JMP     BOOT            ; BACK TO THE SYSTEM

        END
```

*

BASIC Thoughts

John H. Mosow
2238 Southwood Place
Lincoln, NE 68512

It's my turn now! Owning 3 Z-110's and a Z-90, and having read every issue of REMark, it's now my turn to contribute some of my thoughts, experiences, knowledge, and programs to others. I have been programming on the Zenith systems for over three years and have written 6 separate systems which are being used in an executive search firm. Through many trials, tribulations, and with the help of the monthly REMark, I have been able to create some fairly sophisticated systems using BASIC.

My first suggestion to all is to remain consistent. In my case, all of my programs have 'PRINT P6;:' defined to clear the screen. While the variable has no meaning to anyone else, I always know that it will clear the screen. Using this type of consistency makes programming easier and will play an important role as you will see later.

Next, I can never remember the Data Type Declaration Characters. Is the % used for integers or double-precision numbers? Try using the 'DEF' statements in your programs. I almost always start out with 'DEFINT A-L:DEFSTR M-Z' in my programs. Then, I know quickly and easily what type a variable I am working with. In addition, it saves quite a few keystrokes when I am coding my programs.

All of my programs are written in what I call 'generic' BASIC. With minor modifications, my programs should run on any type of BASIC system. I use no PEEKS, POKES, or assembler routines. While my programs might be faster by having some of these features, I feel that the transportability of my programs outweighs my need for these features.

My first problem when programming was to get a good, readable listing of my program. I did not want to put line feeds or spaces in my programs. In some cases, I couldn't, for a line of code was 250 characters long. I ended up writing the program 'UTL002' which gives me a very readable listing. All the listings shown here were done using this program. This program includes the following features:

1. Prints the page number, date and program name at the top of each program listing.
2. Indents the remaining portions of lines greater than 80 characters past the line number. Lines up to 255 characters in length can be printed on multiple lines without the need of line feeds, tabs, or spaces. The line number remains easily identified at the left of the page.
3. Programs to be listed can be entered individually or a file with multiple files to be listed can be used. Using the multiple file option allows you to start printing and walk away while the system does the work.

When using this program, only two rules must be remembered. First, all files must be ASCII files. This means that the programs must be saved with the ',A' option, and that the file used for a multiple file listing must also be an ASCII file. Secondly, no error checking is done. For example, if a file is not found, an error occurs and the program terminates.

The following is the program listing for 'UTL002'.

```
PAGE 1                    10/21/85                    UTL002

100 '"UTL002.BAS" PROGRAM - 03/03/84
200 'USE TO PRINT BASIC PROGRAM LISTINGS
300 DEFINT A-L:DEFSTR M-Z:J = 0:I = 1:APAGE = 0:B = 0
400 P6 = CHR$(27) + "E" + CHR$(13)
500 PRINT P6;:LINE INPUT;"ENTER DATE - (MM/DD/YY) - ";MDATE
600 IF LEN(MDATE) > 8 GOTO 500
700 IF MDATE = "END" THEN PROGFILE = "END":GOTO 1400
800 PRINT P6;:PRINT "'M' FOR MULTIPLE FILES OR 'I' FOR
    INDIVIDUAL FILES OR 'E' F OR END OF JOB":PRINT:PRINT
900 X = INPUT$(1)
1000 IF X = "E" THEN PROGFILE = "END":GOTO 1400
1100 IF X = "M" OR X = "I" GOTO 1200 ELSE GOTO 900
1200 PRINT P6:LINE INPUT;"ENTER FILE TO BE READ - ";
     PROGFILE
1300 IF LEN(PROGFILE) > 14 GOTO 1200
1400 IF PROGFILE = "END" THEN PRINT P6;"END OF JOB";
     :PRINT:PRINT:END
1500 PRINT P6;:
1600 IF X = "I" THEN PROGRAM = PROGFILE:GOTO 2000
```

```
1700 OPEN "I", #2, PROGFILE
1800 IF EOF(2) THEN CLOSE:PROGFILE = "END":GOTO 1400
1900 INPUT#2,PROGRAM
2000 PRINT:PRINT "LISTING - ";PROGRAM
2100 PROGRAM1 = PROGRAM: C = INSTR(PROGRAM,":"):
     IF C = 0 GOTO 2300
2200 PROGRAM = MID$(PROGRAM,C+1,LEN(PROGRAM))
2300 C = INSTR(PROGRAM," "):IF C = 0 GOTO 2500
2400 PROGRAM = LEFT$(PROGRAM,C-1)
2500 OPEN "I", #1, PROGRAM1:IF EOF(1) THEN PRINT:
     PRINT "NO RECORDS IN FILE - PRESS RETURN TO CONTINUE":
     Y = INPUT$(1):CLOSE:GOTO 2600 ELSE GOTO 2700
2600 IF X = "I" GOTO 1200 ELSE GOTO 1800
2700 GOTO 4600
2800 IF EOF(1) THEN CLOSE#1:GOTO 4700
2900 LINE INPUT#1,PLINE
3000 B = INSTR(PLINE," ")
3100 IF B = 0 GOTO 2800
3200 LENGTH = LEN(PLINE):IF LENGTH <= 80 THEN J = 0:
     GOTO 3600
3300 IF LENGTH > 80 AND LENGTH <= 160 - (B) THEN J = 1.
     GOTO 3600
3400 IF LENGTH => 161 - (B) AND LENGTH <= 240 - (B+B)
     THEN J = 2:GOTO 3600
3500 IF LENGTH => 241 - (B+B) THEN J = 3
3600 IF J = 0 THEN LPRINT PLINE
3700 IF J = 1 THEN LPRINT LEFT$(PLINE,80):
     LPRINT SPACE$(B)+MID$(PLINE,81,LENGTH)
3800 IF J = 2 THEN LPRINT LEFT$(PLINE,80).
     LPRINT SPACE$(B)+MID$(PLINE,81,(80-B))
     :LPRINT SPACE$(B)+MID$(PLINE,161-B,LENGTH)
3900 IF J = 3 THEN LPRINT LEFT$(PLINE,80)
     LPRINT SPACE$(B)+MID$(PLINE,81,80-B)
     LPRINT SPACE$(B)+MID$(PLINE,161-B,80-B):
     LPRINT SPACE$(B)+MID$(PLINE,241-(2*B),LENGTH)
4000 I = I + 1 + J:J = 0
4100 IF I > 58 GOTO 4200 ELSE GOTO 2800
4200 B = I - 58:B = 6 - B
4300 FOR A = 1 TO B
4400 LPRINT
4500 NEXT A
4600 APAGE = APAGE + 1:LPRINT "PAGE";APAGE;TAB(36);MDATE;
     TAB(65);PROGRAM:LPRINT:LPRINT:I = 1:J = 0:GOTO 2800
4700 FOR B = I TO 63
4800 LPRINT
4900 NEXT B
5000 J = 0:I = 1:APAGE = 0:B = 0
5100 IF X = "I" THEN GOTO 1200 ELSE GOTO 1800
```

My next problem, how do I get to use libraries with my programs? Libraries are a standard set of statements which can be used in any program. In my case, I use them for defining my standard variables, for a date input routine, for a sort routine, etc. In addition, I also compile all my programs.

At first, I thought the 'INCLUDE' statement in the compiler would take care of this. However, after careful reading, the include statement does not resequence the library. Therefore, an 'INCLUDE' statement must always have the same sequence number in all programs. Plus, there was no way for me to test my programs using the interpreter, a serious loss for me. I thought there had to be a better way.

The program I wrote is called 'MERGE'. It does the library maintenance that I was looking for. The following are the features of the program:

1. Merges library files into a source file and resequences it so that the library can be placed anywhere in a program.
2. Programs can be merged individually or a file with multiple files to be merged can be used. Using the multiple file option allows you to start a merge and walk away while the system does the work.
3. The program checks for sequencing errors while merging libraries and prints where the sequence error occurs.
4. The '/C' option after the program names allows the program to be merged but a new file not written to disk. This is for checking the merge for sequencing errors.
5. The '/L' option lists the program while it is being listed. Using this option gives a complete program listing, including the libraries as they were merged into the program.
6. The new merged file is written to a '.TMP' file. It can then be compiled or executed using the interpreter.
7. I keep a compiled date display in my programs. If the word 'COMPILED' occurs in any line, the date is entered into the '.TMP' file.

There are a few rules to remember when using this program:

1. Programs and libraries must be saved in the ASCII format. Use the ',A' option when saving these files.
2. Enough of a gap must exist between line numbers in the main program for the library to be merged into it. For example, a library with 50 lines will not fit in a program sequenced by 10. I always sequence all my files by 100 using 'RENUM 100,,100'.
3. To enter a library into the source file, the following statement is used: 'REM INCLUDE=B:FILE.EXT'. This must be the only statement on the line.
4. Only 'GOTO' statements can be resequenced in library files using this program. The 'GOTO' line number must also end in a colon, 'GOTO 200:' and can only reference a line number in the library.
5. In the main program, libraries are best referenced as a subroutine. Use the 'GOSUB' statement using the library line number to access a library and be sure that the library contains a 'RETURN' statement.

The following is a small sample program showing how to code a program, what a library would look like, and the merged file.

```
        S A M P L E   B A S E   P R O G R A M

PAGE 1                  10/21/85                    SAMPRG

100 '"SAMPRG.BAS" PROGRAM
200 DEFINT A-L:DEFSTR M-Z
300 GOTO 500
400 REM INCLUDE=B:DATEINP.BAS
500 LINE INPUT "ENTER TODAY'S DATE (MMDDYY) - ";XTEMP
600 X = "":I = 0:GOSUB 400
700 IF X = "X" GOTO 500 ELSE MDATE = XTEMP
800 END

          S A M P L E   L I B R A R Y   F I L E

PAGE 1                  10/21/85                    DATEINP

100 ' DATEINP.BAS LIBRARY
200 IF LEN(XTEMP) <> 6 GOTO 700.
300 IF LEFT$(XTEMP,2) < "01" OR LEFT$(XTEMP,2) > "12"
    GOTO 700:
400 IF MID$(XTEMP,3,2) < "01" OR MID$(XTEMP,3,2) > "31"
    GOTO 700:
500 IF RIGHT$(XTEMP,2) < "00" OR RIGHT$(XTEMP,2) > "99"
    GOTO 700:
600 GOTO 800.
700 X = "X":PRINT XBEEP;:
800 RETURN
900 ' END OF DATEINP.BAS LIBRARY

          S A M P L E   M E R G E D   F I L E

PAGE 1                  10/21/85                SAMPRG.TMP

100 '"SAMPRG.BAS" PROGRAM
200 DEFINT A-L:DEFSTR M-Z
300 GOTO 500
400 REM INCLUDE=B:DATEINP.BAS
```

```
401 ' DATEINP.BAS LIBRARY
402 IF LEN(XTEMP) <> 6 GOTO 407:
403 IF LEFT$(XTEMP,2) < "01" OR LEFT$(XTEMP,2) > "12"
    GOTO 407:
404 IF MID$(XTEMP,3,2) < "01" OR MID$(XTEMP,3,2) > "31"
    GOTO 407:
405 IF RIGHT$(XTEMP,2) < "00" OR RIGHT$(XTEMP,2) > "99"
    GOTO 407
406 GOTO 408
407 X = "X":PRINT XBEEP;.
408 RETURN
409 ' END OF DATEINP.BAS LIBRARY
500 LINE INPUT "ENTER TODAY'S DATE (MMDDYY) - ";XTEMP
600 X = "":I = 0:GOSUB 400
700 IF X = "X" GOTO 500 ELSE MDATE = XTEMP
800 END
```

The following is the listing for the program 'MERGE'.

```
PAGE 1                    10/21/85                    MERGE
100 'MERGE.BAS
200 '
300 DEFSNG A:DEFINT J-K:DEFSTR M-Z:COMMON XINFILE
400 OPTION BASE 1:DIM XARR(300),ANUM(300)
500 P6 = CHR$(27) + "E" + CHR$(13)
600 PRINT P6;:
700 IF LEN(XINFILE) > 0 THEN PRINT P6;:XKBD = "Y":
    PRINT "MERGING ";XINFILE;:GOTO 2300
800 LINE INPUT "ENTER TODAY'S DATE (MM/DD/YY) - ";MDATE
900 PRINT P6;:PRINT "'M' FOR MULTIPLE FILES -
    'I' FOR INDIVIDUAL FILES - 'E' FOR END OF JOB":
    PRINT:Y = INPUT$(1)
1000 IF Y = "E" GOTO 8000
1100 IF Y <> "M" AND Y <> "I" GOTO 600
1200 IF Y = "I" OR XCHK = "Y" THEN PRINT P6;:GOTO 1900
1300 PRINT P6;:PRINT "INSERT SOURCE DISK IN DRIVE 'B'
     AND DESTINATION DISK IN DRIVE 'A'":PRINT:PRINT
     "PRESS RETURN TO CONTINUE":X = INPUT$(1).
     IF X <> CHR$(13) GOTO 1300
1400 RESET:PRINT P6;
1500 PRINT P6;:PRINT CHR$(13);:LINE INPUT
     "ENTER MULTIPLE FILE LIST NAME TO READ - ";XLIST
     XLIST = "B:" + XLIST
1600 OPEN "I", #4, XLIST
1700 IF EOF(4) THEN CLOSE:GOTO 8000
1800 LINE INPUT#4,XINFILE:PRINT P6;:PRINT XINFILE
1900 ACOMP1 = 0:J = 0:K = 0:JLINES = 0:JLIB = 0:J1 = 0
     B = 0:I1 = 0:APAGE = 0:XLIST = "":XCHK = ""
2000 IF Y = "M" GOTO 2300
2100 LINE INPUT "ENTER INDIVIDUAL SOURCE FILE NAME - ";
     XINFILE
2200 PRINT:PRINT
2300 IF XINFILE = "END" THEN GOTO 8000
2400 XLIST = "":XCHK = "":IF INSTR(XINFILE,"/") = 0
     GOTO 2900
2500 IF INSTR(XINFILE,"/C") > 0 THEN XCHK = "Y"
2600 IF INSTR(XINFILE,"/L") > 0 THEN XLIST = "Y"
2700 XINFILE = LEFT$(XINFILE,INSTR(XINFILE,"/")-1)
2800 PROGRAM = XINFILE
2900 IF Y = "M" THEN XOUTFILE = "A:" + XINFILE + ".TMP"
     ELSE XOUTFILE = "B:" + XINFILE + ".TMP"
3000 XINFILE = "B:" + XINFILE + ".BAS"
3100 IF Y = "M" THEN Z = "Z" ELSE Z = ""
3200 OPEN "I", #1, XINFILE:IF XCHK <> "Y"
     THEN OPEN "O", #2, XOUTFILE
3300 IF EOF(1) GOTO 3400 ELSE GOTO 3700
3400 CLOSE#1:IF XCHK <> "Y" THEN CLOSE#2
3500 IF XLIST = "Y" THEN XLIST = "":GOSUB 10100
3600 GOTO 7700
3700 LINE INPUT#1,XINPUT
3800 IF MDATE = "" OR ACOMP1 > 0 GOTO 4000
3900 ACOMP1 = INSTR(XINPUT,"COMPILED"):IF ACOMP1 > 0
     THEN XINPUT = LEFT$(XINPUT,ACOMP1+7)
     + " - " + MDATE + CHR$(34)
4000 ALINE = VAL(LEFT$(XINPUT,INSTR(XINPUT," ")-1))
4100 JLINES = JLINES + 1
4200 IF XCHK <> "Y" THEN PRINT#2, XINPUT
4300 IF XLIST = "Y" THEN PLINE = XINPUT:GOSUB 8600
4400 A = INSTR(XINPUT,"REM INCLUDE="):IF A = 0
     OR XINFILE = "B:MERGE.BAS" GOTO 3300
4500 JLIB = JLIB + 1:
     XLIB = RIGHT$(XINPUT,LEN(XINPUT)-INSTR(XINPUT,"="))
4600 LINE INPUT#1,XINPUT:
     ASAVE = VAL(LEFT$(XINPUT,INSTR(XINPUT," ")-1))
4700 J = 0:OPEN "I", #3, XLIB
4800 IF EOF(3) THEN CLOSE#3:GOTO 5100
4900 J = J + 1:LINE INPUT#3,XARR(J)
5000 GOTO 4800
5100 FOR I = 1 TO J.
     ANUM(I) = VAL(LEFT$(XARR(I),INSTR(XARR(I)," ")-1))
     NEXT I
5200 FOR I = 1 TO J
5300 FOR K = 1 TO J
5400 A = INSTR(XARR(I),"GOTO "):IF A = 0 THEN K = J
     GOTO 5700
5500 B = INSTR(A,XARR(I),":"):C = VAL(MID$(XARR(I),A+5,B-1))
5600 IF C = ANUM(K) THEN X = STR$(K + ALINE)
     XARR(I) = LEFT$(XARR(I),A+4) + RIGHT$(X,LEN(X)-1) +
     RIGHT$(XARR(I),LEN(XARR(I))-B+1)
5700 NEXT K
5800 FOR K = 1 TO J
5900 A = INSTR(XARR(I),"GOSUB "):IF A = 0 THEN K = J.
     GOTO 6200
6000 C = VAL(RIGHT$(XARR(I),LEN(XARR(I)) - (A+5)))
6100 IF C = ANUM(K) THEN X = STR$(K + ALINE)
     XARR(I) = LEFT$(XARR(I),A+5) + RIGHT$(X,LEN(X)-1)
6200 NEXT K
6300 FOR K = 1 TO J
6400 A = INSTR(XARR(I),"RETURN**"):IF A = 0 THEN K = J.
     GOTO 6700
6500 C = VAL(RIGHT$(XARR(I),LEN(XARR(I)) - (A+7)))
6600 IF C = ANUM(K) THEN X = STR$(K + ALINE).
     XARR(I) = LEFT$(XARR(I),A+5) + X
6700 NEXT K
6800 NEXT I
6900 FOR I = 1 TO J
7000 IF ASAVE < I + ALINE THEN PRINT
     "SEQUENCE NUMBER OUT OF RANGE AT SOURCE LINE - ",
     ALINE + I:PRINT:PRINT:PRINT "END OF JOB":PRINT:
     PRINT:CLOSE:END
7100 X = STR$(I + ALINE):XARR(I) = RIGHT$(X,LEN(X)-1) +
     RIGHT$(XARR(I),LEN(XARR(I))-(INSTR(XARR(I)," ")-1))
7200 JLINES = JLINES + 1
7300 IF XCHK <> "Y" THEN PRINT#2,XARR(I)
7400 IF XLIST = "Y" THEN PLINE = XARR(I):GOSUB 8600
7500 NEXT I
7600 GOTO 4000
7700 PRINT:PRINT:PRINT "NUMBER OF LIBRARIES - ";JLIB:
     PRINT:PRINT "TOTAL NUMBER OF LINES - ";JLINES:
     PRINT:PRINT:PRINT
7800 IF XKBD = "Y" GOTO 8200
7900 IF Y = "I" GOTO 1900 ELSE GOTO 1700
8000 IF Z <> "Z" OR XCHK = "Y" GOTO 8200
8100 PRINT P6;:PRINT
     "INSERT SYSTEM DISK IN DRIVE 'A' AND PRESS RETURN"
     PRINT:PRINT:X = INPUT$(1).
     IF X <> CHR$(13) GOTO 8100 ELSE RESET
8200 PRINT P6;:PRINT "END OF JOB":PRINT:PRINT:PRINT:END
8300 '
8400 ' PRINT ROUTINE
8500 '
8600 B = INSTR(PLINE," ")
8700 Z = INKEY$:IF Z = CHR$(13) THEN XLIST = "".
     GOSUB 10100 ELSE GOTO 8900
8800 RETURN
8900 IF APAGE = 0 THEN GOSUB 10600
9000 LENGTH = LEN(PLINE):IF LENGTH <= 80 THEN J1 = 0:
     GOTO 9400
9100 IF LENGTH > 80 AND LENGTH <= 160 - (B) THEN J1 = 1:
     GOTO 9400
9200 IF LENGTH => 161 - (B) AND LENGTH <= 240 - (B+B)
     THEN J1 = 2:GOTO 9400
9300 IF LENGTH => 241 - (B+B) THEN J1 = 3
9400 IF J1 = 0 THEN LPRINT PLINE
9500 IF J1 = 1 THEN LPRINT LEFT$(PLINE,80)
     LPRINT SPACE$(B)+MID$(PLINE,81,LENGTH)
9600 IF J1 = 2 THEN LPRINT LEFT$(PLINE,80):
     LPRINT SPACE$(B)+MID$(PLINE,81,(80-B)):
```

```
     LPRINT SPACE$(B)+MID$(PLINE,161-B,LENGTH)
9700 IF J1 = 3 THEN LPRINT LEFT$(PLINE,80):
     LPRINT SPACE$(B)+MID$(PLINE,81,80-B):
     LPRINT SPACE$(B)+MID$(PLINE,161-B,80-B):
     LPRINT SPACE$(B)+MID$(PLINE,241-(2*B),LENGTH)
9800 I1 = I1 + 1 + J1:J1 = 0
9900 IF I1 > 58 THEN GOSUB 10100
10000 RETURN
10100 B = I1 - 58:B = 6 - B
10200 FOR A = 1 TO B
10300 LPRINT
10400 NEXT A
10500 IF XLIST = "" THEN RETURN
10600 APAGE = APAGE + 1:LPRINT "PAGE";APAGE;TAB(36);MDATE;
      TAB(65);PROGRAM:LPRINT:LPRINT:I1 = 1:J1 = 0:B = 0
10700 RETURN
```

My next challenge, convert all these CP/M programs to MS-DOS. Thanks to the consistency of my screen manipulation variables and my use of the 'generic' BASIC, this proved to be an easy task. The screen manipulation variables, which I have defined in a library file, are as follows:

1. **'PRINT FNXFUN(BX,BY);:'** — This is used for direct cursor addressing. The MS-DOS version replaces my 'PRINT FNXFUN(JX,JY)' statements with the proper 'LOCATE' statements.
2. **'PRINT P6;:'** — This statement is used to clear the screen. It is replaced with the 'CLS' statement for MS-DOS.
3. **'PRINT PM;:'** — This is used to enable and clear the 25th line and move the cursor to it. Because MS-DOS requires that the 25th line be enabled before moving the cursor to it, this is replaced with 'PRINT PM;:LOCATE 25,1' in the MS-DOS version of the program.
4. **'PRINT PN;:'** — This statement was used in CP/M to disable the 25th line and locate the cursor in the HOME position. It is replaced with the 'LOCATE 1,1' statement.
5. **'PRINT P1;:'** — This statement is used to turn the cursor off. It is replaced with the appropriate 'LOCATE' statement.
6. **'PRINT P2;:'** — This statement is used to turn the cursor on. It is replaced with the appropriate 'LOCATE' statement.
7. **'PRINT XBEEP;:'** — This is used to sound the bell on the system. For MS-DOS, it is replaced with the 'BEEP' statement.
8. **'PRINT XKEYS;:'** — I have this defined to display a standard display for the function keys on the 24th line. This statement is replaced with a 'LOCATE 24,1' statement plus a print of this standard description line.

As you can see, I have only 8 variables defined. By remaining consistent with how these variables are used, my conversion to the MS-DOS version of BASIC is an automatic and easy process.

Features of the program, 'TRANSFER', include:

1. New source code is produced with an extension of '.TMP'. Therefore, original code is not lost and the new version can be removed as need be.
2. A 'FIND' option exists for proofing the transfer before actually creating the new file.

The rules for using the program, 'TRANSFER', are quite simple:

1. Programs must be saved in the ASCII format. Use the ',A' option when saving these files.
2. Only one 'PRINT PM;:' statement can occur in a line. Because the MS-DOS version still uses this statement, it is never removed and will, therefore, go into a loop if more than one statement occurs in the line.
3. Be careful how long a line of code is. While most of the MS-DOS version of the statements are shorter than the CP/M version, care should be taken so that the new line will not exceed 255 characters.

Note: I have since combined both the 'TRANSFER' and the 'MERGE' programs to do both functions in a single program. This has proven to be a time saver when proven CP/M programs are brought across to MS-DOS.

The following is the listing for the program 'TRANSFER'.

```
100 'TRANSFER.BAS PROGRAM - 11/30/83
200 DEFSTR P,X,Y,Z:DEFINT A,B,C,O
300 P6 = CHR$(27) + "E" + CHR$(13):PRINT P6;:
400 B = 0
500 PRINT "'F' FOR FIND OR 'C' TO CHANGE OR 'E' TO END"
600 PRINT:PRINT
700 Z = INPUT$(1):IF Z = "E" THEN PRINT P6;:END
800 IF Z = "F" OR Z = "C" GOTO 900 ELSE GOTO 500
900 LINE INPUT;"ENTER FILE NAME - ";XINPUT
1000 PRINT
1100 IF Z = "F" GOTO 1400
1200 LINE INPUT;"ENTER NEW FILE NAME - ";XOUTPUT:
     OPEN "O", #2, XOUTPUT
1300 PRINT
1400 OPEN "I", #1, XINPUT
1500 IF EOF(1) THEN CLOSE:PRINT:PRINT:GOTO 400
1600 B = 0:LINE INPUT#1, XLINE
1700 XOLD = XLINE
1800 A = INSTR(XLINE,";FNXFUN("):IF A = 0 GOTO 2200
1900 A = A + 7:B = INSTR(A+2,XLINE,","):
     C = INSTR(B,XLINE,";")
2000 XLINE = LEFT$(XLINE,A-7) + "·
     LOCATE " + MID$(XLINE,A+1,B-A) +
     MID$(XLINE,B+1,C-B-2) + ".
     PRINT " + MID$(XLINE,C+1,(LEN(XLINE)-C+1))
2100 GOTO 1800
2200 A = INSTR(XLINE,"PRINT FNXFUN(")·IF A = 0 GOTO 2600
2300 A = A + 11:B = INSTR(A+2,XLINE,",")
     C = INSTR(B,XLINE,";")
2400 XLINE = LEFT$(XLINE,A-12) + "LOCATE "
     + MID$(XLINE,A+2,B-A) + MID$(XLINE,B+2,C-B-3) + "·
     PRINT " + MID$(XLINE,C+1,(LEN(XLINE)-C+1))
2500 GOTO 2200
2600 A = INSTR(XLINE,"PRINT PM;:"):IF A = 0 GOTO 3000
2700 A = A + 9:B = LEN(XLINE) - 9
2800 XLINE = LEFT$(XLINE,A-10) + "LOCATE 25,1:
     PRINT SPC(79);XCR;:" + RIGHT$(XLINE,LEN(XLINE)-A)
2900 B = 1:GOTO 2600
3000 A = INSTR(XLINE,"PRINT PN;:"):IF A = 0 GOTO 3400
3100 A = A + 9:B = LEN(XLINE) - 9
3200 XLINE = LEFT$(XLINE,A-10) + "LOCATE 1,1:
     " + RIGHT$(XLINE,LEN(XLINE)-A)
3300 B = 1:GOTO 3000
3400 A = INSTR(XLINE,"PRINT P6;:"):IF A = 0 GOTO 3800
3500 A = A + 9:B = LEN(XLINE) - 9
3600 XLINE = LEFT$(XLINE,A-10) + "CLS:
     " + RIGHT$(XLINE,LEN(XLINE)-A)
3700 B = 1:GOTO 3400
3800 A = INSTR(XLINE,"PRINT XBEEP,.");IF A = 0 GOTO 4200
3900 A = A + 12:B = LEN(XLINE) - A
4000 XLINE = LEFT$(XLINE,A - 13) + "BEEP:"
     + RIGHT$(XLINE,LEN(XLINE)-A)
4100 B = 1:GOTO 3800
4200 A = INSTR(XLINE,"PRINT P1;:"):IF A = 0 GOTO 4600
4300 A = A + 9:B = LEN(XLINE) - A
4400 XLINE = LEFT$(XLINE,A - 10) + "LOCATE,,0:"
     + RIGHT$(XLINE,LEN(XLINE)-A)
4500 B = 1:GOTO 4200
4600 A = INSTR(XLINE,"PRINT P2;:"):IF A = 0 GOTO 5000
4700 A = A + 9:B = LEN(XLINE) - A
4800 XLINE = LEFT$(XLINE,A - 10) + "LOCATE,,1:"
     + RIGHT$(XLINE,LEN(XLINE)-A)
4900 B = 1:GOTO 4600
5000 A = INSTR(XLINE,"PRINT XKEYS;:"):IF A = 0 GOTO 5400
5100 A = A + 12:B = LEN(XLINE) - A
5200 XLINE = LEFT$(XLINE,A-13) + "LOCATE 24,1:PRINT XKEYS,
     ." + RIGHT$(XLINE,LEN(XLINE)-A)
5300 B = 1
5400 A = INSTR(XLINE,":PRINT  "):IF A = 0 GOTO 5800
5500 A = A + 7
5600 XLINE = LEFT$(XLINE,A - 8) + "·"
```

```
    + RIGHT$(XLINE,LEN(XLINE)-A)
5700 GOTO 5400
5800 IF B > 0 THEN PRINT XOLD:PRINT "NEW",XLINE:PRINT
5900 IF Z = "F" GOTO 1500 ELSE PRINT#2, XLINE
6000 B = 0:GOTO 1500
```

My final project, write a program that would build a file for merging and compiling programs, and still allow for some of the compiler options. This particular project required the help of my REMark library.

Please Note: From this point on, all the programs must be in a compiled CP/M form in order for them to work with the next 2 programs!

The first problem, how do I pass a program name into the 'MERGE' program? Well, I diverted from my 'generic' philosophy a bit and recalled an article on command line input for compiled programs. It contained a 'POKE' statement. I conceded and used it.

My second problem quickly came up. If I tried to put this routine into the 'MERGE' program, it did not work. Why, I don't know? So I wrote 'MRGINP' which will pass the program name onto the 'MERGE' program. That is why there is a 'COMMON' statement in the program.

There is only one rule when running the program. The source must reside on the 'B' drive and have an extention of '.BAS'. To start the merge, merely enter, 'MRGINP NAME', where name is the program to be merged.

The following is the listing for the program 'MRGINP'.

```
PAGE 1                  10/21/85                        MRGINP

100 DEFINT B-L:DEFSTR M-Z:COMMON XINFILE
200 IN = PEEK(128):IF IN > 16 THEN IN = 16
300 FOR I = 1 TO IN
400 J = PEEK(129+I):IF J < 32 THEN J = 32
500 XFL = XFL + CHR$(J)
600 NEXT I
700 POKE 128,0
800 IF J > 32 THEN XFL = XFL + CHR$(J)
900 XINFILE = XFL:CHAIN "MERGE"
1000 END
```

Finally, there was the program to tie the merging and compiling together. The program is 'UTL001' and it creates a SUBMIT file which takes care of all my manual compiling tasks of merging the source and libraries, compiling the programs, entering compile options, and removing the temporary combined source and the '.REL' files from the disks. In addition, for a little flexibility, it will also allow for the creation of any type of ASCII file. I use this for creating other submit files and for creating my files for multiple merges or listings.

Again, REMark came in quite handy. One of the issues showed what format a '.SUB' file had to be.

The features of this program are:

1. The 'SUB' option allows the creation of an ASCII file. The file name is specified and then the lines of the file are entered. When 'END' is entered, input is terminated.
2. A file called 'COMPILE.SUB' is created. This file contains the names of the files to be compiled and the options to be used.
3. Before the program name is entered, the options are specified by using the appropriate function key. The options can be changed for each program entered.
4. When 'COMP' is entered, a '$$$.SUB' file is created and the compiles are started automatically. If 'END' is entered, the program merely goes to an end of job.

There is basically only one rule to remember with this program. When compiling programs which are to be merged first, enter the program as 'NAME.TMP'. The program name you enter here is the name which will be used as the source name in the compile statement.

Note: I have the 'SUBMIT.COM' program copied on my disks as 'RUN.COM'. That is why in line 2400 I have the statement 'RUN COMPILE' entered in the '$$$.SUB' file.

The following is the listing for the program 'UTL001'. Notice that it does use a library.

```
PAGE 1                  10/21/85                        UTL001

100 ' UTL001.BAS - USED TO BUILD COMPILE DECK
200 '
300 DEFINT A-L:DEFSTR M-Z
400 '
500 REM INCLUDE=B:DEFINE.BAS
501 ' DEFINE.BAS LIBRARY
502 XCR = CHR$(13):XTAB = CHR$(9):XBEEP = CHR$(7):
    XBLNK = "":XSPC = " "503 PM = CHR$(27) + "x" + "1"
    + CHR$(27) + CHR$(89) + CHR$(56) + CHR$(32) + XCR
    + SPACE$(80) + XCR:PN = CHR$(27) + "y" + "1" + XCR
504 XKEYS = PN + CHR$(27) + "Y" + CHR$(55) + CHR$(32) +
    XCR + "<F1>      <F2>      <F3>      <F4>      <F5>
      <F6>      <F7>      <F8>" + XCR
505 PSTOP = CHR$(127):P6 = PN + CHR$(27) + "Y" + CHR$(32)
    + CHR$(32) + CHR$(27) + "E" + XCR.
    P1 = CHR$(27) + "x" + "5" + XCR:
    P2 = CHR$(27) + "y" + "5" + XCR
506 ' END OF DEFINE.BAS LIBRARY
600 '
700 DEF FNXFUN(BX,BY) = CHR$(27) + "Y" + CHR$(31 + BX)
    + CHR$(31 + BY)
800 '
900 XCOM = "A:":XREL = "A:":XERR = "":XSTR = "":
    XERA = "Y":JX = 22:JY = 1:XMRG = "Y":XTRACE = ""
1000 '
1100 '
1200 PRINT P6;:PRINT "UTL001 - COMPILED - 06/16/85"
1300 FOR I = 1 TO 1000:NEXT I
1400 '
1500 OPEN "O", #1, "COMPILE.SUB"
1600 IF JX > 21 THEN PRINT PN;:PRINT P6;:JX = 5:
     PRINT FNXFUN(1,1);:PRINT XCR;"ENTER FILE NAME
     OR 'SUB' OR 'COMP' OR 'END'";: ELSE GOTO 1800
1700 GOSUB 5800
1800 J = 79:GOSUB 6500
1900 IF K > 0 THEN GOSUB 3700 ELSE GOTO 2100
2000 GOTO 1600
2100 XFILE = XTEMP
2200 IF XTEMP = "" THEN PRINT XBEEP;:GOTO 1800
2300 IF XFILE <> "COMP" GOTO 2500
2400 OPEN "O", #2, "$$$.SUB":PRINT#2,CHR$(11);
     "RUN COMPILE";CHR$(0);"$":XFILE = "END"
2500 IF XFILE = "END" THEN CLOSE:PRINT P6;
     PRINT "END OF PROGRAM - UTL001":PRINT:PRINT:END
2600 IF XFILE = "SUB" GOTO 9000
2700 IF INSTR(XFILE,":") = 0 THEN XFILE = "B:" + XFILE
2800 IF INSTR(XFILE,".") = 0 THEN XFILE = XFILE + ".BAS"
2900 IF LEN(XFILE) > 14 THEN PRINT XCR;:PRINT FNXFUN(JX,JY);
     :PRINT XCR;SPC(LEN(XFILE));XCR,:PRINT FNXFUN(JX,JY);
     :PRINT XCR;:PRINT XBEEP;:GOTO 1600
3000 A = INSTR(XFILE,"-") + 1:B = INSTR(XFILE,".");
     XBASE = MID$(XFILE,A,B-A).
     XEXT = RIGHT$(XFILE,LEN(XFILE)-B)
3100 IF XMRG = "Y" THEN PRINT#1,"MRGINP " + XBASE
3200 PRINT#1,"BASCOM " + XREL  + XBASE + ".REL,="
     + XFILE + XERR + XSTR + XTRACE
3300 IF XEXT = "TMP" AND XERA = "Y"
     THEN PRINT#1, "ERA " + XFILE
3400 PRINT#1, "L80 " + XREL + XBASE + ".REL," + XCOM
     + XBASE + " COM/N/E"
3500 PRINT#1, "ERA " + XREL + XBASE + ".REL"
3600 JX = JX + 1:GOTO 1600
3700 ON K GOTO 3900,4100,4300,4500,4700,4900,5100
3800 PRINT XBEEP;:GOTO 5300
```

```
3900 IF XCOM = "A:" THEN XCOM = "B:" ELSE XCOM = "A:"
4000 GOTO 5300
4100 IF XREL = "A:" THEN XREL = "B:" ELSE XREL = "A:"
4200 GOTO 5300
4300 IF XERR = "" THEN XERR = "/E" ELSE XERR = ""
4400 GOTO 5300
4500 IF XSTR = "" THEN XSTR = "/S" ELSE XSTR = ""
4600 GOTO 5300
4700 IF XERA = "Y" THEN XERA = "N" ELSE XERA = "Y"
4800 GOTO 5300
4900 IF XMRG = "Y" THEN XMRG = "N" ELSE XMRG = "Y"
5000 GOTO 5300
5100 IF XTRACE = "" THEN XTRACE = "/D" ELSE XTRACE = ""
5200 GOTO 5300
5300 GOSUB 5800
5400 RETURN
5500 '
5600 ' PRINT OPTIONS ROUTINE
5700 '
5800 PRINT FNXFUN(24,1);:PRINT XCR;"COM DRV";TAB(11),
     "REL DRV";TAB(21);"ERROR";TAB(31);"STRINGS";TAB(41);
     "ERASE TMP";TAB(51);"MERGE";TAB(61);"TRACE";XCR;
5900 PRINT PM;:PRINT TAB(3);XCOM;TAB(13);XREL;TAB(22);
     XERR;TAB(33);XSTR;TAB(45);XERA;TAB(53);XMRG;
     TAB(63);XTRACE;XCR;
6000 PRINT FNXFUN(1,1),,PRINT XCR,
6100 RETURN
6200 '
6300 ' KEYBOARD INPUT ROUTINE
6400 '
6500 PRINT XCR;:PRINT FNXFUN(JX,JY);:K = 0:XTEMP = "":
     FOR I = 1 TO J
6600 X = INPUT$(1):Y = INKEY$
6700 IF X = CHR$(27) AND I = 1 THEN GOSUB 7800
6800 IF K > 0 GOTO 7600
6900 IF X = XCR THEN I = J + 1:GOTO 7600
7000 IF I = 1 GOTO 7400
7100 IF X = CHR$(8) THEN XTEMP = LEFT$(XTEMP,LEN(XTEMP)-1)
     PRINT XCR,:PRINT FNXFUN(JX,JY-2+I),:PRINT " ";XCR;
     :PRINT FNXFUN(JX,JY-2+I);:I = I - 2 ELSE GOTO 7400
7200 IF I < 0 THEN I = 1
7300 GOTO 7600
7400 IF X < CHR$(32) OR X > CHR$(127) THEN PRINT XBEEP;
     :GOTO 6600
7500 XTEMP = XTEMP + X:PRINT X;:
7600 NEXT I
7700 RETURN
7800 IF Y = CHR$(83) THEN K = 1
7900 IF Y = CHR$(84) THEN K = 2
8000 IF Y = CHR$(85) THEN K = 3
8100 IF Y = CHR$(86) THEN K = 4
8200 IF Y = CHR$(87) THEN K = 5
8300 IF Y = CHR$(80) THEN K = 6
8400 IF Y = CHR$(81) THEN K = 7
8500 IF Y = CHR$(82) THEN K = 8
8600 I = J + 1:RETURN
8700 '
8800 ' CREATE FREE-FORM .SUB FILES
8900 '
9000 JX = 22:PRINT PN,:PRINT P6;,
     LINE INPUT "ENTER FILE NAME - ";XFILE1
9100 IF LEN(XFILE) > 14 THEN PRINT XBEEP,:GOTO 9000
9200 OPEN "O", #2, XFILE1
9300 IF JX > 21 THEN PRINT P6;:PRINT "ENTER LINE":JX = 5
9400 J = 70:GOSUB 6500
9500 IF XTEMP = "END" THEN JX = 22:CLOSE#2:GOTO 1600
9600 IF XTEMP = "" GOTO 9400
9700 PRINT#2,XTEMP:JX = JX + 1:GOTO 9300
```

I hope that these thoughts and programs will make someone else's programming life as easy as it has made mine. I will continue reading REMark for other useful programs that I can apply to the systems and utilities I write.

*

On The Leading Edge

The Z-241, MS-DOS 3.1, And Hard Disk Support

William M. Adney
P.O. Box 531655
Grand Prairie, TX 75053-1655

Like many useful tools, the use of a computer system is a double-edged sword. On the one hand, it provides the capability to significantly improve one's productivity through the use of word processors, spreadsheets, databases, and other application software. That's the good news.

The bad news is that you can quickly, easily, and quite effectively commit computerized hara-kiri. By falling on your computerized sword, you can rather painlessly (until you realize what has happened) initiate computer suicide by performing the disembowelment of data on a floppy or a hard disk. In other words, you totally erased all programs and data on a disk by using ERASE *.* (you were logged onto the wrong disk) or you entered the wrong drive letter for the FORMAT command. It takes only a few seconds to wipe out MEGABYTES of data!

That's one of the reasons that I have often suggested that you make frequent backups of data, even for a floppy disk. It is simply too easy to make a mistake, particularly when you get in a hurry. Don't take the approach that you will never make a mistake. The only thing that I can promise you is that you WILL do it and it WILL happen to you. It's not a question of "IF"; it's a question of WHEN.

On Committing Computerized Hara-kiri

Those of us who have used the old version 1 Z-DOS were particularly vulnerable to this problem. If a drive letter was not specified in the FORMAT command, the program assumed that you wanted to FORMAT the default drive and merrily did so. I lost more than one system disk because of that particular problem. That's why I specifically noted that you should ALWAYS enter the drive letter for the FORMAT command on page 118 of the MS-DOS FlipFast book. And that is still a good idea, incidentally.

I've had a hard disk in my H-100 for three years. During that time, I have had a total of five complete data losses on my 26 MB hard disk. The first three were due to a simple media failure. Another one was due to a power failure which did something strange to the hard disk and caused a complete data loss. The last one was due to a problem (that has been fixed) in the version 1.5 ROM on the Easy PC that wiped out my hard disk as I reported in June. But those failures only caused me to lose about two hour's worth of data since I had backups.

Zenith has always provided a nice set of hard disk utilities, but I believe that their most significant contribution has been programs like PREP, PART, ASSIGN/ASGNPART, DETECT, and SHIP. As far as I know, Zenith is the only manufacturer that has provided a utility like PREP as part of the operating system software. That is particularly unique since IBM does not provide anything in PC-DOS that allows a user to test and "initialize" a hard disk. All IBM hard disks are "factory prepared".

Regardless of the manufacturer, however, the use of a hard disk carries a large responsibility for the vast amounts of data that it can contain. But there is more to it than that.

Computer Problems

As reported last month, one user has blamed Zenith for the loss of business data on his '241 with a 38 megabyte Winchester hard disk. But that is not too unusual. Large companies like Zenith and IBM are convenient scapegoats for problems whether they were caused by user error or not.

Even IBM has had its share of problems. If you haven't seen it, I recommend you look at the cover of the April 29, 1986 issue of PC Magazine. Articles included in that issue are based on the idea that "You Can't Go Wrong Buying IBM . . . Or Can You"? The articles talk about the IBM AT hard disk failures that have even occurred at Ziff-Davis, publishers of PC Magazine. In the cover story "Courting Disaster: The IBM PC AT", they report that their five ATs have trashed a total of 12 hard disks in six months. Even given that they probably use their computers more than the average individual, it's still a pretty poor record.

Although there appears to be no way to isolate the cause of the problem, the fact is that a number of AT hard disks have failed in normal use. IBM has consistently maintained that there are no abnormal failures associated with the AT hard disks, but a lot of

people don't believe that. Even worse, there is absolutely no agreement among the experts as to what is causing the failures. Some blame the manufacturer of the hard disk; some blame the disk controller; and everybody seems to blame IBM. Ah well, I guess that is one of the hazards of being a large company.

Although this might be a problem with "new" technology, none of the technology used is really new. Even the well-known Peter Norton has had several hard disk problems with the IBM AT. His column in the PC Magazine issue previously mentioned will give you some idea about his view of the IBM AT: "Down with the PC AT, Up with the Compaq 286". Too bad he doesn't appear to know about the Z-200.

Most of the industry experts, like Peter Norton, seem to agree that there was some kind of hardware problem (not DOS) with the AT, even though they don't agree what it is. But enough of IBM's problem with the AT. Let's review a little of the history behind the hard disk support provided by MS-DOS and its derivatives, such as PC-DOS.

The History Of PC Hard Disk Support

Version 2 of PC-DOS was required to support any hard disk on an IBM system. And since that time, MS-DOS and PC-DOS have been stuck with the 32 megabyte limit on the size of the hard disk that could be used with a DOS. That, in my opinion, was a design flaw by Microsoft like the 640K memory limit for the IBM PC. Users have complained for a couple of years about that limit, but it has only been recently that anything has been done about it.

But there is some rationale behind the 32 megabyte limit. As it turns out, MS-DOS has historically been able to address a maximum of 64K (FFFFh - 65,536 decimal) physical sectors on a disk. That is the absolute limit as far as DOS is concerned. Those of you who have worked with assembly language will recognize the significance of that number since it is the limit for addressing with 16-bit arithmetic.

One thing about discussing disk sectors is that you must be specific as to whether you are talking about physical sector numbers or logical sector numbers. PHYSICAL SECTOR NUMBERS are the actual numbers of the sectors on the disk beginning with sector 1. By convention, all disks begin with physical sector 1.

Consider a standard 9 sector, DS/DD floppy disk. The first track on the top (called side 0) of the disk (called track 0) contains sectors 1 through 9. Sectors 10 to 18 are located on the first track on the bottom side (called side 1) of the disk. Sector 19 begins on track 1 of side 0 of the disk, and so on. Great, you say, but who cares? Well, this technique (called cylinder mapping) is important since it allows the DOS to access any kind of disk — hard disk or floppy. In particular, this is the reason that version 1 of PC-DOS could not be used with a hard disk. Version 1 used track mapping.

LOGICAL SECTOR NUMBERS are specifically used by the DOS to address any kind of disk. In particular, it is important to recognize that the MS-DOS kernel (i.e. MSDOS.SYS or equivalent) "thinks" in terms of logical sectors. More importantly, it is the Microsoft MS-DOS kernel which has the 16-bit arithmetic limit for calculating logical sector numbers (i.e. 0-65,535 decimal). The most important point of this is to recognize that the capability to address a maximum of 64K disk sectors is a Microsoft limitation, not a Zenith one.

Microsoft supplies DOS code to Original Equipment Manufacturers (OEMs) like Zenith; however, some parts of the code are not usually modified by OEMs. This normally includes both the MS-DOS kernel, as well as the command interpreter COMMAND .COM. OEMs are required to provide a Basic Input\Output System (BIOS) for their hardware. You may recall that the BIOS provides the interface between the DOS and the computer hardware.

The BIOS must use physical sector numbers to access a disk since that is part of the hardware interface to the DOS. But the MS-DOS kernel uses logical sector numbers during processing. The end result is that the BIOS must perform all of the logical to physical sector translation routines from the DOS to the disk.

Aside from this you may be wondering if it is important for most computer users to understand physical and logical sectors. In general, the answer is an emphatic no. The major exception to that is that many programmers MUST understand how this process works. For example, DEBUG uses logical sectors for the Load and Write subcommands because DOS thinks in terms of logical sectors. If you are writing a disk sector editor, you need to understand how that translation works in detail.

Which reminds me of a letter that I received recently. One Huggie was trying to write a disk sector editor in BASIC and wanted to know what was wrong with his program. The biggest single problem is that BASIC is not an appropriate language to write that kind of program. Some of the disk directory information and all of the File Allocation Table (FAT) information was set up to be "easily decoded" in assembly language. BASIC was simply not intended for easy use of bit-level fiddling that's required.

Now I won't say that it can't be done in BASIC, but I certainly don't know how to do it even though I've written a couple of low-level disk access programs for my own use in C and assembler. I learned to program in COBOL and FORTRAN about 20 years ago, and I simply have no use for a language that does not use labels and has very primitive I/O functions. Yeah, I know that some BASIC's now allow labels, but many of the programming constructs are still VERY clumsy. And although I have even taught a class in BASIC, I always recommend Pascal (especially Turbo Pascal) as the best programming language for beginners. More on programming languages in a future article, but back to sectors and such.

If you have followed this so far, you have seen that since the Microsoft supplied kernel cannot access more than 64K logical sectors, the OEM supplied BIOS may also be limited to 64K physical sectors in the translation routine. But what does that mean in terms of the maximum disk size?

Calculating The Disk Size Limit

Every PC oriented magazine that I have seen has discussed the IBM PC 32 megabyte hard disk limit at one time or another even though the reason for the limit has usually not been explained. However, it is extremely easy to calculate the 32 megabyte limit given a very limited number of facts.

We already know that DOS can address a maximum of 64K sectors or 65,536 decimal. If you also know that DOS uses a standard sector size of 512 bytes per sector, then you can do the following calculation:

```
65,536 sectors x 512 bytes/sector = 33,554,432 bytes
```

So far, so good. Now let's calculate the storage capacity of a 32 megabyte disk as follows:

```
32 MB x 1024 bytes/KB x 1024 KB/MB = 33,554,432 bytes
```

Although I could have simply divided the result of the first calculation by 1,048,576 bytes/megabyte, I wanted to make the point

that "KB" or "MB" values are in terms of 1024, not 1000 as many people think. And since we are already on the subject of calculating kilobytes, let's calculate the actual capacity of 640K memory as follows:

```
640 KB x 1024 bytes/KB = 655,360 bytes
```

Regardless of whether you understand the technical parts of this or not, I hope it is clear that the physical 32 megabyte limit is a Microsoft limitation, not a Zenith one. But the implications of this limit are more significant than that.

I have seen a criticism that the '241 cannot "use" a hard disk larger than 32 megabytes with early versions of MS-DOS 3.1 (i.e. IO.SYS version 3.03 and below). The facts are that all current versions of PC-DOS (including version 3.2) STILL cannot "use" all of the capacity of a hard disk with a physical capacity of more than 32 megabytes. If you have purchased a hard disk for the IBM AT with, say a 43 megabyte capacity, you can only define 32 megabytes of it for use as a DOS partition. The balance of the capacity, 11 megabytes, can ONLY be defined as a non-DOS partition for use with any operating system that supports that capability.

That leads to the logical question: "Why are vendors selling hard disks larger than 32 megabytes for PCs and ATs when PC-DOS cannot use that disk space?" Part of the answer is that many PC-DOS users are finding out the hard way that they cannot use that excess capacity. The other part of the answer is that most network software DOES allow the use of that capacity when the disk is used as a file server. Aside from that, we'll take a look at the IBM approach to hard disks later in this article. I think you'll find that enlightening.

The Original Zenith (Microsoft) MS-DOS Limitations

All Zenith version 2 releases of the PC series MS-DOS and early version 3.1 releases (IO.SYS 3.03 and below) provided the same hard disk support as the IBM PC (and AT) machines and PC-DOS. That is, the Microsoft imposed 32 megabyte physical hard disk limit for DOS existed and Zenith did not change it.

And if you take a look at any of the IBM advertisements, you will see that they sell ATs with a 30 megabyte drive. That should give you a clue that they still have not broken the 32 megabyte limit, even though you can easily buy larger drives from other vendors. But regardless of the drive size, you will still be limited to the use of 32 megabytes per physical drive because of the standard DOS limitation.

The point is that both version 2 and early version 3.1 releases of Zenith MS-DOS were EXACTLY compatible with PC-DOS even down to the 32 megabyte physical hard disk limit. Later releases (IO.SYS 3.04 and above) were enhanced to allow a 32 megabyte PARTITION size limit so that a 128 megabyte hard disk could be used in the "Zenith mode". It seems to me that a safe assumption would be that the Zenith MS-DOS is exactly PC compatible. Therefore, one should "know" the PC-related restrictions and assume that they also apply to the Zenith MS-DOS in the absence of any information to the contrary. And even though Zenith provides us with the most extensive set of DOS documentation in the industry, that doesn't seem to be enough for some users. You can't please everyone I guess.

Avoid The DTs

Mechanical and media failures aside, you CAN avoid most Disk Trashes with a modest amount of effort and very little technical knowledge. You have already taken the first step by becoming a HUG member and reading REMark. Users of Heath and Zenith equipment will also find that Sextant magazine also provides a lot of good information on their systems. And if you have a PC series computer, I recommend something like PC Magazine or PC Tech Journal since both provide an excellent perspective on those IBM systems. Most of the articles that I write are a direct result of something new that I have learned or a mistake that I have made. In some cases, I have simply written about things that I have known, but didn't think they were important until I noticed that a user was having some kind of difficulty.

And so it is with the 32 megabyte DOS limit. Contrary to some reports, this problem is not a Zenith "bug" at all — it's a known Microsoft DOS restriction. It is unfortunate that this restriction is not quite as well known as Microsoft, IBM, and Zenith may have assumed. I have even looked through all of the IBM PC-DOS manuals including the DOS Technical Reference, but I was unable to find any specific statement about the 32 megabyte DOS limitation even though it's easy to calculate based on the published information.

Twenty-twenty hindsight would seem to indicate that perhaps Zenith should have explicitly included it in their documentation even though IBM and other manufacturers have not. Maybe, maybe not. But it seems to me that if we users are going to insist that Heath and Zenith provide us with PC compatibility, we must also accept some responsibility for understanding what that really means. To that end, I will be including more information about PC compatibility issues in my articles, but I won't neglect the Z-100. It's still the workhorse that I use for most of my books and articles.

But enough of that problem. Let's take a look at how Zenith has "fixed" the Microsoft limitation through the use of some rather clever and sophisticated programming.

The Zenith MS-DOS Enhancement

Even though the Microsoft MS-DOS kernel is still limited to addressing 64K of logical disk sectors, Zenith has found a way to "fix" the 32 megabyte physical limitation. This enhancement has been added to MS-DOS 3.1 with an IO.SYS version of 3.04 and above. You can use the VER command to determine your IO.SYS and MSDOS.SYS versions.

The ZDS BIOS has been enhanced to use 24-bit arithmetic for addressing physical disk sectors. As a result, the Zenith MS-DOS systems can address a total of 256 times the original 64K sectors for a total disk size of 16,777,216 sectors. You can calculate the maximum size of the disk as follows:

```
16,777,216 sectors x 512 bytes/sector = 8,589,934,592 bytes
```

Hmmm . . . a modest 8 billion bytes or so. I wonder what that is in megabytes so I'll divide that by 1,048,576 bytes/megabyte. Rats! Zenith shortchanged me on that since I am NOW limited to an 8,192 MEGABYTE hard disk! Oh well, I guess you can't have everything, so I'll go looking for an eight gigabyte hard disk.

(Much, much later) — Well, the bad news is that I could not find an 8 gigabyte hard disk. The best I could find was a commercially available IBM 3380 mainframe disk drive with about a one gigabyte capacity. Sigh. I was also wondering how I would be able to get that anyway since it would have only cost about $100,000 by the time I bought the unit and provided all of the special power and other requirements.

As it turns out, that was probably all for the best since the disk controller is hardware limited to 128 megabytes anyway. I guess I'll have to wait at least until next year to get my multi-gigabyte hard

disk drive. So much for the DOS limitation, and the clever way that Zenith was able to overcome it.

The MS-DOS Hard Disk Utilities For The PC Series

But there are a few things that are not obvious even though the information can be found in the Zenith MS-DOS manuals. For example, did you know that the MS-DOS 3.1 CONFIGUR command directly influences how your hard disk operates? We'll take a look at that in more detail in a minute.

However, one user report says that the Zenith MS-DOS manuals for both the Z-100 and PC series infer that only two hard disk partitions can be active at one time. Even though this user correctly notes that all versions of Zenith MS-DOS can have four active partitions at one time, it's difficult for me to see how the manuals infer a two partition limit. Both of the Zenith MS-DOS manuals for the version 2 ASSIGN command specifically mention the four partitions for the Z-100 (page 11.8) and the PC series (page 11.7). For version 3.1, the ASGNPART command specifically mentions this on pages 11.10 (Z-100) and 11.18 (PC series). There is no inference on these pages since it is explicitly stated. I guess the moral of this story is to always take a few minutes to review the documentation when you get any software. This user apparently never took the time to read the detailed information about how the ASSIGN/ASGNPART command was used.

This same report also says that MS-DOS version 3 requires that you must update your CONFIG.SYS file with the LASTDRIVE parameter in order to have more than two active hard disk partitions. That turns out not to be the case. I tried that on my own '241 and it does not work. I got the error message: "Invalid drive/drive not available" when I tried to use the ASGNPART command to assign drive E. What happened? To answer that, we need to take a quick look at the IBM philosophy for using a hard disk.

The IBM Approach

Unlike Zenith, IBM clearly does not have any use for multiple DOS partitions. As a matter of fact, this is quite clear when you realize that the IBM FDISK command does not even allow more than one DOS partition for ANY physical hard disk. Even more revealing is a statement from the IBM DOS Version 3.00 Technical Reference (page 8-3) that says: "Any operating system must consider its partition to be an entire disk, and must ensure that its functions and utilities do not access other partitions on the disk." Therefore, if you want to have a second DOS partition under PC-DOS, you MUST have a second hard disk drive. That is why I believe that it is unlikely that IBM will worry about the 32 megabyte hard disk limit in the near future. Their solution is to add another hard disk if you want a second DOS partition. I think that approach is kind of tacky. Why should I buy another hard disk just to have a second partition? This is one of those features that Zenith has specifically provided us with a rather elegant alternative. We have always been able to use four active partitions on a single hard disk.

IBM has another little hook in the latest DOS versions which performs Automatic Partition Assignment. What does THAT do? Well, PC-DOS automatically assigns the DOS partition (only one allowed remember) on each physical hard disk to a drive letter. The first drive is assigned as drive C and a second drive (if it exists) is assigned as drive D. This happens to be required if you are using IBM's Topview or Microsoft Project.

The Zenith CONFIGUR Command

In order to maintain the IBM PC compatibility AND still retain the best hard disk features in the industry, Zenith had to come up with an interesting solution. The CONFIGUR command in version 3.1 allows you to disable (or set) the Automatic Partition Assignment flag. It is particularly important to note that the PC series MS-DOS version 3 system is shipped with the Automatic Partition Assignment flag turned on so that it will perform just like the equivalent PC-DOS system.

That's important because you will get the "Invalid drive/drive not available" error message that I mentioned earlier if you try to ASGNPART a third partition on the hard disk. Changing the LAST-

Listing 1
Hard Disk Characteristics Of The IBM AT And Z-200

	Z-200				IBM AT				
Drive Type	Cyls	Heads	Pre-C	Capacity	Type	Cyls	Heads	Pre-C	Capacity
1	306	4	128	10 MB	Same				
2	615	4	300	20 MB	Same				
3	699	5	256	30 MB	3	615	6	300	31 MB
4	940	8	512	64 MB	Same				
5	940	6	512	48 MB	Same				
6	615	4	FFFFH	20 MB	Same				
7	699	7	256	42 MB	7	462	8	256	31 MB
8	733	5	FFFFH	31 MB	Same				
9	900	15	FFFFH	115 MB	Same				
10	925	5	FFFFH	40 MB	10	820	3	FFFFH	21 MB
11	855	5	FFFFH	36 MB	Same				
12	855	7	FFFFH	51 MB	Same				
13	306	8	128	20 MB	Same				
14	733	7	FFFFH	44 MB	Same				
15	Reserved Same								

Cyls = Number of cylinders
Pre-C = Write pre-compensation cylinder (FFFFH = No Pre-comp required)

DRIVE parameter will not fix the error — you MUST use the CONFIGUR command to allow Manual Partition Assignment for use with the ASGNPART command.

In summary, the Automatic Partition Assignment flag allows MS-DOS to function just like and be compatible with PC-DOS and PC software. The Manual Partition Assignment flag allows MS-DOS to function in the "Zenith mode" that allows up to four active hard disk partitions that may be spread across one or more hard disks. If you intend to use the ASGNPART command, just remember that you must use the CONFIGUR command to set the Manual mode.

Other Zenith Hard Disk Utilities

I suppose that we all get a little spoiled by some of the features that Zenith provides with all of their operating systems. In many cases, it is easy to lose sight of the many advantages of the hardware and software that Heath and Zenith have made available. Perhaps that's because we are simply used to the best and have become too critical of the things that they haven't done. As I said earlier, it's very easy to criticize a large company, but I think that it is also important to keep the criticism in perspective.

Let's compare the Zenith MS-DOS hard disk utilities with those supplied with PC-DOS. For example, PC-DOS does not have a command equivalent to ASGNPART (or ASSIGN in version 2) because it is assumed that a single hard disk will only have one DOS partition as I mentioned earlier. That may be a dynamite assumption if you want to sell a lot of hard disks, but I think it is kind of dumb. In order to have an IBM system that would operate in the "Zenith mode", you would have to buy four hard disks just in order to have four partitions assuming that PC-DOS and the IBM hardware could support that which it can't.

Take a look at the Zenith PREP command. It tests and initializes a hard disk to help ensure reliability. And to help users verify the reliability of their hard disk, the DETECT utility is one of the nicest features yet. It provides an easy way to determine whether or not you have any potential bad sectors that are developing on your disk.

The DETECT Utility

If you own a hard disk, you should run the DETECT utility at least once a month to check the reliability of the media. That should be Standard Operating Procedure for any hard disk system. Why? Consider the following error message:

```
Bad sector error at address 88F
Abort, Retry, Ignore?
```

If you have a hard disk and haven't seen that yet, you're lucky. But the problem is that MS-DOS cannot read that sector. Since there may be a slight glitch in any disk media (usually called a soft error), DOS is instructed to perform a specified number of retries in the BIOS. That number is on the order of 3-5 retries depending on the specific operating system. When the Zenith MS-DOS displays that message, it has already exhausted the retry attempts specified in the BIOS. Although you can continue to retry in the "manual" mode from that message, you now have a hard error which will probably result in the loss of data.

It should be no surprise that a good disk sector can be read (or written) with a single try. Therefore, the DETECT utility simply tries to read each sector on the hard disk, and if it can't read the sector in one try, it updates the Bad Sector Table with that bad sector address. The rationale is that more than one try to read a sector indicates that it is becoming marginal. Since the program only attempts to read a sector, it does not destroy any data on the disk.

You can tell that DETECT has found a problem when it displays the "Bad sectors located. Tables modified" message. My procedure is to run DETECT immediately after I complete my monthly hard disk backup of all data in every partition. If DETECT does find a bad sector, I can then reformat to "lock out" that bad sector and then restore all data to the disk.

It is important to recognize that you must FORMAT the hard disk again when DETECT finds a bad sector in order to prevent future loss of data. DETECT only performs a non-destructive read on each sector and updates the Bad Sector Table when the first attempt to read the sector fails. The FORMAT program reads the Bad Sector Table when it formats a hard disk and specifically updates the File Allocation Table (FAT) with an FF7 hex for the cluster containing the bad sector. DOS recognizes that "reserved" (i.e. bad) cluster and never uses it to store any data or programs. If you are interested in more details about this, you might want to review my article on "Files, FATs, and Directories" that appeared in the March 1986 issue of REMark.

A Talk With Zenith

It appears that the user I have mentioned thinks all of his problems were caused by Zenith "bugs" or a failure to document something in a manual. I have considerable difficulty believing that as I have already said since I have seen nothing that indicates that this user's problems were anything other than user error — either failure to know the system limits or failure to read the Zenith documentation or both.

However, since these criticisms have already become "public knowledge", I thought it would be interesting to get Zenith's perspective on this. Since most of these criticisms seem to center around the issue of a "Zenith MS-DOS problem", I specifically asked Ron Kasik, Director of ZDS Operating Systems Software to comment. In response to a question about these issues, Ron simply told me: "Zenith does not intentionally try to mislead users in technical details. Our documentation is the most comprehensive in the industry and I think that the physical size of the Zenith MS-DOS manuals is sufficient evidence of that."

Although I don't believe that Zenith is always right, particularly with respect to their pricing policies, I find it difficult to believe that one could reasonably criticize Zenith for providing a DOS that is PC compatible even to the 32 megabyte physical hard disk limit. Either you have PC compatibility or you don't. And since there are many issues related to PC compatibility, I have already planned additional information in several articles which will discuss some things that I have briefly mentioned here. The moral of the story is: "Learn your system limitations". That will help prevent your losing data and getting the DTs. But there are still a few other ways to lose data.

10 Ways To Lose Your Data

There are many ways to lose your Winchester data (or any data for that matter), but I thought I would share ten of them with you. Each one of these will give you the DTs. And although there is no way to totally eliminate the DTs, it is useful to understand some of the possibilities. These are not in any particular order by the way.

The first way is to have a power failure at the wrong time. I had this happen to me last year. A five second power loss apparently did something to the Superblock on my hard disk. All my data and programs were gone, but I only lost about two hours' worth of work because I take backups as I write.

The next way, and perhaps the most common, is to have a number of bad sector errors caused by hard disk media failure. All storage media, both floppy and hard disks, deteriorate with time and usage. That's happened to me three times.

Perhaps the most common way to lose data is due to simple mistakes, such as ERASE *.* or using FORMAT on the wrong disk drive. I'll never tell how many times I've done that, but it's more than once. It is interesting to note that, at a recent meeting of the Dallas HUG group, I was assured by each and every member present that NOBODY in that organization had EVER made a mistake like this.

A mechanical hard disk failure, such as a head crash, will also cause loss of data. I haven't heard of too many head crashes on microcomputer hard disks, but they are a common occurrence on mainframe computer systems.

Bugs in hardware can also cause loss of data. I mentioned earlier that the AT hard disk problems may have been caused by the hard disk drive and/or the disk controller although there seems to be no consensus among the experts about this.

Problems with application software, such as word processors, may cause significant data loss. I have never wanted to work again with Multimate since my bad experience last year.

Failure to read and understand information concerning the DOS commands (including the hard disk commands) will almost certainly cause a massive loss of data at some time due to mistakes.

In some cases, you must have the latest DOS version that has new features in order to prevent loss of data. The Zenith DOS enhancement that allows for high-capacity hard disk drives is one example of this.

If you really want to ensure losing all of your hard disk data, the final way to do so is to drop your system from the top of a 50 story building. You have my personal guarantee that all data will be irrecoverably lost.

But there is a way to at least minimize, if not prevent, the loss of data.

Adney's 3 Rules For Using A Computer System

The user that I have mentioned in this article could have avoided the loss of his business data by following three simple rules that apply to the use of any computer system from micro to mainframe. These same rules apply whether you are using a 100 KB hard-sectored floppy disk on an H-89 or a one gigabyte hard disk on a mainframe system.

First, back up all data periodically. Define a specific schedule and follow it religiously. The schedule should be determined by your estimate of the importance of the data and how often it is updated. Don't skip a periodic backup because you "don't have time". Write a batch file to perform the backup on your lunch hour.

Next, back up data frequently. That means that you should back up data daily (or even hourly) if necessary. Business computer users should be especially careful to take backups every time a major file update is made to minimize data loss. My frequent data backups are made EVERY time I spend more than three or four hours writing. Sometimes I make hourly backups, particularly when I have been writing a difficult part of an article or book. I usually keep a floppy disk in a drive specifically for backup purposes and use the COPY command to make a continuous, updated backup.

Last, but not least, back up data ALWAYS. That includes all data and programs even though you don't have a hard disk. It's easy to

Listing 2
Extended Disk Characteristics Of The 8 Mhz IBM AT And Z-200

	Z-200				IBM AT (8 Mhz)				
Drive Type	Cyls	Heads	Pre-C	Capacity	Type	Cyls	Heads	Pre-C	Capacity
16	612	4	0	20 MB	Same				
17	977	5	300	42 MB	Same				
18	977	7	FFFFH	58 MB	Same				
19	1024	7	512	61 MB	Same				
20	733	5	300	31 MB	Same				
21	733	7	300	44 MB	Same				
22	733	7	300	44 MB	22	733	5	300	31 MB
23	306	4	0	10 MB	Same				
24	612	2	FFFFH	Syquest	Reserved				
25	615	6	300	31 MB	n/a				
26	462	8	256	31 MB	n/a				
27	820	3	FFFFH	21 MB	n/a				
28	754	11	FFFFH	71 MB	n/a				
29	918	15	FFFFH	117 MB	n/a				
30	987	5	FFFFH	42 MB	n/a				
31	830	6	400	42 MB	n/a				
32	Reserved				n/a				

Cyls = Number of cylinders
Pre-C = Write pre-compensation cylinder (FFFFH = No Pre-comp required)
n/a = not available

use the COPY, DISKCOPY or BACKUP command to make a back-up of a single file or a complete disk. Then you will always have at least some capability to recover data when you do the ERASE *.* trick or FORMAT the wrong disk. Business users should also consider the use of a streaming tape to back up a hard disk on a regular basis.

Comparing The Z-200 And The IBM AT

One thing that many users simply don't have time to do is a direct comparison of computer systems. And since this article is about the hard disks, I thought it might be a useful idea to compare the hard disk support provided by the IBM AT and the Z-200. Although this might seem to be a trivial exercise, it is important to understand that all PC hard disk characteristics are contained in the system ROM tables. For the Z-200 and other Zenith compatibles such as the AT, you must define the hard disk with a ROM-based SETUP command. The results are interesting.

Listing 1 shows the ROM-based hard disk characteristics of the Z-241 and the standard IBM AT. Although other characteristics are listed in the drive tables, I have only shown the most commonly known ones. Notice that the Z-200 has 4 different entries (in both Listing 1 and Listing 2) compared to the AT. When I asked Zenith about that, their answer was that they believe that those drives are most reliable. The differences between the AT and the Z-241 don't mean that the AT isn't Zenith compatible, just different.

And for the first time anywhere, I also managed to get some information on the updated Zenith ROM and 8 megahertz IBM AT tables as shown in Listing 2. The extended Zenith hard disk definitions will work in both the '241 and the new 248. The AT definitions are only available with the 8 megahertz AT; the 6 megahertz IBM AT cannot be retrofitted with the new ROM according to my sources. I suppose that's because IBM, in their infinite wisdom, decided to put a hook in the 6 Mhz AT ROM that prevents that system from running at speeds greater than 6 Mhz.

A New Zenith Product

Zenith has announced a dynamite little accessory, called the Real-Time Clock, which works on both the Z-100 and the Z-150 series computers. It is just slightly thicker than a regular IC socket and piggybacks into the system ROM socket in both computers. Since the top of the Z-100 ROM needs a little more clearance from the video board, Zenith has thoughtfully included three spacers and longer screws for reinstalling the video board. The spacers and screws are not needed on the Z-150 systems.

The Real-Time Clock assembly (the IC "socket") consists of an IC and a completely sealed lithium battery to ensure that it won't leak on a printed circuit board. Aside from the Real-Time Clock, you also get a disk with the software to set and read the unit. After installation, all you need to do is set the current time and date with the CLOCK command, add the RDCLOCK program to your bootable disk and update the AUTOEXEC.BAT file. Source code, consisting of C and assembler, is also included on the disk in case you're interested in the programming details.

I have one in my '100 and I really like it; however, I still don't believe that someone at Zenith does much market research on similar products. The Real-Time Clock lists at $59.00 which is $10.00-20.00 more than a similar product that I can get locally. As I've said before, it's too bad that Zenith feels compelled to price a lot of their accessories significantly above the competition. Perhaps they haven't figured out the Borland (Sidekick, Reflex, and Turbo Pascal) marketing philosophy which is simply to sell LOTS of products at a reasonable price. In any case, I highly recommend the Real-Time Clock.

ANOTHER Strike For Copy-Protected Programs

A friend of mine recently wiped out a disk on an IBM AT at work because of copy protection. Fortunately, it was not his master copy because he had made a backup with COPYIIPC. The program was Microsoft Chart (his personal copy) that he was using to develop a presentation.

All he did was insert the disk into the A drive on the AT and try to start the program. The disk was trashed and was totally useless. The DTs strike again! What happened? The A drive was a high density unit, and it appears that the Microsoft copy protection scheme tried to do some kind of update to the disk. It is (I hope) well known that a high density drive cannot WRITE RELIABLY to a standard 9-sector 360K disk. Regardless of exactly what happened, the end result was a wiped out disk. What should YOU do?

If you have a Z-200 or an AT, the first thing you should do if you have both drive types (i.e. 360K and 1.2 MB) is find a label maker and some black label tape. Label each drive.

If you use any copy-protected software, in most cases you MUST swap the drives so that the 360K drive is always the A drive. The reason is that some copy-protection schemes actually write to the disk like Microsoft Chart. In addition, some copy-protected software MUST also be run in drive A. Don't take any chances. Even if you don't have any copy-protected software, I still recommend that you swap the drives. Since virtually all software is distributed on 360K disks, you won't have any problem with that.

Most of you already know that I won't review any copy-protected software because I don't believe it is in the best user interest. Many vendors, including Microsoft, are eliminating copy-protection from their latest software. Version 3 of Microsoft Word does not have any copy protection for example. A number of vendors are boldly advertising the fact that their software is not copy-protected.

You may not realize that most of this is due to a lot of market pressure from users. In most cases, it is a fact that software that is NOT copy-protected sells better than software which is. The major "hold-outs" at this point include both Lotus and Ashton-Tate (dBase). I don't see how that can last much longer since there are "work-alike" products available that are MUCH cheaper and NOT copy-protected.

If you happen to be in a situation where you can't get away from copy-protected software, you should find a "copy-buster" program that allows you to make backup copies. Don't ever use the master distribution disk to run the program. Always run it from the copy. You could end up with a wiped out disk like my friend.

Too Technical?

At least one user seems to think that I expect everyone to be very technical and be able to understand more about programming and computers than the average person does. That turns out not to be the case.

My background consists of about 20 years' experience in data processing including a wide range of technical, management, and consulting experience. I have managed Systems and Programming groups, as well as user groups. Regardless of the position that I held, I have always found that an informed computer user is nearly always a happy computer user. And the converse is usually true

also: an uninformed computer user is typically quite unhappy with his system and criticizes almost everything about it.

In a mainframe system, the user normally doesn't need a lot of technical knowledge since a qualified data processing staff is available. That is not normally the case in a microcomputer environment. A microcomputer user typically performs ALL of the duties of a data processing staff. That includes system programming, telecommunications, application programming, systems analysis, systems design, and computer operations.

I have also taught introductory computer-related courses on both the graduate and undergraduate level at universities on the West Coast and here in Texas. I'm currently teaching at UTA, and I've found that my students don't have too many problems with DOS because I spend about two classes talking about it and the DOS commands before we ever look at any applications software.

It's kind of like the old saying that goes something like: "Don't give a man a bunch of fish; give him a fishing pole and teach him how to fish." Self-sufficiency is the name of that game. And the same thing is true for computers. An informed user will usually be able to quickly and effectively be able to recover from many errors, and probably won't make too many mistakes in the first place.

That is part of the purpose of the "General Information Series" of articles that I have been writing. However, as I was writing this article, one other point occurred to me.

Perhaps we get so involved with our Heath and Zenith systems that we sometimes forget the advantages that these systems have over others. And since many of you may not have had the chance to do direct comparisons with other Zenith clones, I'll try to point out some of the differences in future articles.

Closing REMarks

With a little luck, there are some additional new things that I'll be able to report on next month. Both of these new things that I have in mind will be of interest to many Huggies. In case you haven't looked closely at the Fall Heathkit catalog (No. 203), you will find that the long-awaited Z-100 Windows has been quietly listed on page 93 under Operating Systems. We'll take a look at that next month. No, I will not neglect the Z-100 series computers because I also have a '241.

For those of you who haven't seen it, Steve Robbins has developed PC WatchWord for the PC series computers and other Zenith clones. We'll be taking a look at it in the next month or two. I have a pre-release copy that I have been working with for a couple of weeks. It is quite similar to the Z-100 WatchWord although many function keys have been changed around for obvious reasons. Based on my preliminary review, it looks like Steve has developed another nice product that will probably be highly recommended.

I'll be glad to answer any questions about information in this article if you enclose a stamped, self-addressed envelope with your letter. And don't forget, IBM is the largest manufacturer of Zenith clones in the world today, so we'll be doing some more direct comparisons in the future.

Products Discussed

Software

MS-DOS Version 3	
PC only (OS-63-31)	$150.00
Z-100 only (OS-63-30)	150.00
Programmer's Utility Pack (CB-3163-30)	150.00

Hardware

Advanced Personal Computer (HS-241)	$2499.00
Monochrome/Color Video Card (Z-409)	239.00
20MB Winchester (ZD-200)	1499.00
40MB Winchester (ZD-400)	2499.00
H-100 Desktop Computer (HS-1108-41)	999.00
Real-Time Clock (CB-5063-30)	59.00

Heath/Zenith Computer Centers
Heath Company Parts Department
Hilltop Road
St. Joseph, MI 49085
(800) 253-7057 (Heath Catalog orders only)

WatchWord (Z-100 Only)	$100.00
Resident Speller (Z-100 MS-DOS)	100.00
PC WatchWord (PC Series)	99.95
PC Resident Speller (Z-150/PC)	99.95

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

*

Charles F. Ballinger
S. 3810 Havana
Spokane, WA 99223

Automating The Configure Feature

Trying to use both a serial and a parallel printer on your Zenith Z-100? Are you currently going thru the following scenario — run CONFIGUR to set your letter quality printer up, use your word processor, then run CONFIGUR again to put things back the way they were? Well, here is a one-time approach to this problem that allows you to do it all via a batch file.

I'm currently using my Z-100 with both a parallel dot matrix printer, and a serial letter quality printer. Needless to say, the hassle of running CONFIGUR every time I needed to switch from one to the other finally was more than I could bear.

This article explains how I automated the CONFIGUR routine and incorporated it into a BATCH file, so now it is automatically configured for my serial printer and then reconfigured to parallel upon exiting the word processor. While I could just reset the computer after CONFIGURing only in memory, I could not see the need of going thru the date/time everytime I wanted to change from dot matrix to letter quality. The nice thing about this procedure is that once set up, you do not have to change it again, unless you change printers.

The key to automating the CONFIGUR program is the ability to create a file that contains just the desired responses without extra carriage returns (CR's), or linefeeds (LF's), as part of the text.

Since my word processor happens to be WordStar, I chose to use the non-document mode to set up the initial portion of the file I needed. Actually, you can use any word processor or text processor, even lowly EDLIN, to complete this first phase.

The first thing to do is to run thru the CONFIGUR program and make a note of the exact entries required. Pay close attention to whether or not you have to press the RETURN key or just enter the letter choice.

In my configuration, I am running a TranStar 140 serial letter quality printer, so my particular configuration was as follows:

- Select option "A" from the first CONFIGUR menu screen, for configuring the PRN device.
- Select option "J" from the next menu, for User defined selection.
- Select option "B" for the device type (selects the serial device).

The next menu will ask you four questions that are responded to with either "Y" or "N", or just a carriage return to accept the default response.

Strip parity on input? (Y/N) <N>	Press return for default
Strip parity on output? (Y/N) <N>	Press return for default
Map lowercase to uppercase on input? (Y/N) <N>	Press return for default
Map lowercase to uppercase or output? (Y/N) <N>	Press return for default

Note: For these four questions you can either press the RETURN key to accept the default answer of no, or you could enter the letter "N". The next menu allows selection of the I/O port, my serial printer is attached to port A (J1) (E8H) so I entered option "A".

Now comes the baud rate selection menu. In my case, my printer runs at 1200 baud, so I entered an option selection of "I".

The next menu is the handshaking menu. Here, my printer required RTS Positive (pin 4), so I selected option "D".

Now you are at the stop bit menu, so you can select the appropriate number of stop bits required. In my case, this was option "C", for two stop bits.

The CONFIGUR program will now ask the following question:

Do you wish to use parity? (Y/N) <N> Press return for default

The next item you must select is the word length. My selection was "D", for an 8 bit word.

Now comes the section on pad characters. Since my printer does not require a pad character of any kind, I simply responded with the RETURN key to the prompt, "Press the key corresponding to your desired pad character". I also responded with a RETURN key to the question for number of pad characters.

At this point, you are now back at the main CONFIGUR menu Now I selected option "G", to make changes to memory only, and then selected option "E" to exit the CONFIGUR program.

As a brief review, I entered the following responses to the various menus, or questions:

A
J

B
C/R
C/R
C/R
C/R
A
I
D
C
C/R
D
C/R
C/R
C/R
G
E

As you can see, I had to enter 18 responses in order to complete my particular setup for the CONFIGUR program. The C/R's listed above are actually carriage returns, (hex 0D).

Now that I knew just what entries were required, I could now generate a temporary file using WordStar in the non-document mode.

I opened a file in non-document mode called, "WSSETUP.DAT", and then entered in the responses shown above. The only difference is that you cannot enter a carriage return in WordStar without entering a linefeed also. To bypass this, just enter a space in place of the C/R that is required.

If you look at Listing 1, you will see that the debug dump of the file WSSETUP.DAT shows the above entries, and that the spaces are now hex 20s (space character).

```
C:\>DEBUG A:WSSETUP.DAT
-D
378D:0100  41 4A 42 20 20 20 20 41-49 44 43 20 44 20 20 20  AJB    AIDC D
378D:0110  47 45 0D 0A 1A 1A 1A 1A-1A 1A 1A 1A 1A 1A 1A 1A  GE.
378D:0120  1A 1A 1A 1A 1A 1A 1A 1A-1A 1A 1A 1A 1A 1A 1A 1A
378D:0130  1A 1A 1A 1A 1A 1A 1A 1A-1A 1A 1A 1A 1A 1A 1A 1A
378D:0140  1A 1A 1A 1A 1A 1A 1A 1A-1A 1A 1A 1A 1A 1A 1A 1A
378D:0150  1A 1A 1A 1A 1A 1A 1A 1A-1A 1A 1A 1A 1A 1A 1A 1A
378D:0160  1A 1A 1A 1A 1A 1A 1A 1A-1A 1A 1A 1A 1A 1A 1A 1A
378D:0170  1A 1A 1A 1A 1A 1A 1A 1A-1A 1A 1A 1A 1A 1A 1A 1A
```

Listing 1

We now need to use debug to change these spaces to actual carriage returns (hex 0D), in order for CONFIGUR to correctly recognize the commands.

We do this with DEBUG, using the FILL command. Immediately following Listing 1 you can see the necessary entries to convert the spaces, (hex 20), to carriage returns (hex 0D). If you look at Listing 1, location 112, you will see hex 0D 0A (this is WordStar's carriage return/linefeed combination). You are going to replace it with hex 1A 1A at those two locations — these are end-of-file indicators.

The FILL command syntax is as follows:

F Addr Length Hex characters

For example, the first entry F 103 L4 0D 0D 0D 0D tells DEBUG to FILL starting at address 103 hex for a length of 4 bytes, and to place the hex characters 0D (carriage returns), at location 103 for 4 characters. Remember, when you do a "D" to display memory on your system, the first four digits shown for the address may be different from mine. Only the last four digits are important, as the first four will be dependent upon how much RAM you have in your system, and what version of MS-DOS/ZDOS you are using.

In Listing 2, you can see the final results of our changes using DEBUG. All the hex 20s are now hex 0Ds, and we removed the last carriage return/linefeed that WordStar placed in the file with the appropriate end-of-file indicators. We can now write this file, using the "W" command of DEBUG and then enter a "Q" to exit the DEBUG program.

We have now completed the first phase, setting up our serial letter quality printer. Now, we need a similiar method to un-set it when we have finished using our word processor to restore the system to the parallel printer.

To do this, we create a file similiar to the one above, although much shorter, that will execute CONFIGUR again, and this time RESET the system back to parallel operation.

In my system, this is accomplished by the following options from CONFIGUR:

- Select the "A" option from the first CONFIGUR menu screen, for configuring the PRN device.
- Now select option "A", again to configur the PRN device as a centronics parallel device.

  The program will now draw a picture of your backpanel and all you do is press the RETURN key, this will require a carriage return in the actual data file.
- You now select option "G" (make changes to memory), or option "H", if you wish to change the hard disk.
- Now select option "E" to exit the CONFIGUR program and you are ready to go.

As before, we now have the entries we need, they are:

A
A
C/R
G
E

I once again opened a non-document file in WordStar, called "WSRETURN.DAT", and entered the responses shown above. Once again, I needed to enter a space, since WordStar cannot enter a carriage return without adding a linefeed character.

Listing 3 shows the file after you have saved it in WordStar and are looking at it using DEBUG. In this file, the only thing we have to change is to replace the first hex 20 with hex 0D (space with a carriage return), and then replace the hex 0D 0A with hex 1A 1A (carriage return/linefeed combination with end-of-file indicators). Again, we use the FILL command in DEBUG to do this. The first fill instruction is F 102 L1 0D, which will change the first hex 20 to 0D. The second FILL instruction changes the carriage return/linefeed combination to two hex 1As (end-of-file indicator). After this is completed, we again use the DEBUG command "W" to write the file to disk. Listing 4 shows what the file looks like after you have made the changes. You can verify your changes were made correctly by entering the DEBUG command D 100, like I did, to redisplay memory starting at location 100 hex. After

```
-F 103 L4 0D 0D 0D 0D
-F 10B L1 0D
-F 10D L3 0D 0D 0D
-F 112 L2 1A 1A

-D 100
378D:0100  41 4A 42 0D 0D 0D 0D 41-49 44 43 0D 44 0D 0D 0D   AJB....AIDC.D. .
378D:0110  47 45 1A 1A 1A 1A 1A 1A-1A 1A 1A 1A 1A 1A 1A 1A   GE              .
378D:0120  1A 1A 1A 1A 1A 1A 1A 1A-1A 1A 1A 1A 1A 1A 1A 1A                   .
378D:0130  1A 1A 1A 1A 1A 1A 1A 1A-1A 1A 1A 1A 1A 1A 1A 1A     .  ...  .
378D:0140  1A 1A 1A 1A 1A 1A 1A 1A-1A 1A 1A 1A 1A 1A 1A 1A   ......  ..  .  .
378D:0150  1A 1A 1A 1A 1A 1A 1A 1A-1A 1A 1A 1A 1A 1A 1A 1A         .
378D:0160  1A 1A 1A 1A 1A 1A 1A 1A-1A 1A 1A 1A 1A 1A 1A 1A               ...
378D:0170  1A 1A 1A 1A 1A 1A 1A 1A-1A 1A 1A 1A 1A 1A 1A 1A        . ..
-W
```

Listing 2

There it is. You now can set up the CONFIGUR program to suit your particular requirements and make the entry or exit for special setups hassle-free. This is a lot easier than trying to explain to your wife or the kids how to set the computer up so they can use the letter quality printer, not to mention the time savings if they have a problem running the CONFIGUR program.

This sequence can be used on any program that requires repetitive entry for which you wish to only do the entry once and then forget about it. Once set up, only the name of the batch file that invokes the procedure need be remembered, or written down, instead of pages of instructions.

I hope this procedure helps you and reduces your frustration level, as it has mine. Now changing back-and-forth between a dot matrix printer and the letter quality printer is only a command away. *

you are satisfied with the changes, you exit DEBUG using the Q command and you are almost home free.

The last step is setting up a batch file to invoke the various commands necessary to load CONFIGUR, load your word processing program, and then load CONFIGUR again, a final time. In my case, I set up a batch file called, "WSLETTER.BAT" that contained the following commands:

```
CLS
.
```

```
C:\>DEBUG A:WSRETURN.DAT
-D
378D:0100  41 41 20 47 45 0D 0A 1A-1A 1A 1A 1A 1A 1A 1A 1A   AA GE.          . .
378D:0110  1A 1A 1A 1A 1A 1A 1A 1A-1A 1A 1A 1A 1A 1A 1A 1A      .
378D:0120  1A 1A 1A 1A 1A 1A 1A 1A-1A 1A 1A 1A 1A 1A 1A 1A        .   .         .
378D:0130  1A 1A 1A 1A 1A 1A 1A 1A-1A 1A 1A 1A 1A 1A 1A 1A        .
378D:0140  1A 1A 1A 1A 1A 1A 1A 1A-1A 1A 1A 1A 1A 1A 1A 1A              .       ..
378D:0150  1A 1A 1A 1A 1A 1A 1A 1A-1A 1A 1A 1A 1A 1A 1A 1A                 ..      .
378D:0160  1A 1A 1A 1A 1A 1A 1A 1A-1A 1A 1A 1A 1A 1A 1A 1A        .    ..      .
378D:0170  1A 1A 1A 1A 1A 1A 1A 1A-1A 1A 1A 1A 1A 1A 1A 1A           .
```

Listing 3

```
-F 102 L1 0D
-F 105 L2 1A 1A

-D 100
378D:0100  41 41 0D 47 45 1A 1A 1A-1A 1A 1A 1A 1A 1A 1A 1A   AA.GE.  .
378D:0110  1A 1A 1A 1A 1A 1A 1A 1A-1A 1A 1A 1A 1A 1A 1A 1A   . . .. ....... .
378D:0120  1A 1A 1A 1A 1A 1A 1A 1A-1A 1A 1A 1A 1A 1A 1A 1A         .  . . .
378D:0130  1A 1A 1A 1A 1A 1A 1A 1A-1A 1A 1A 1A 1A 1A 1A 1A
378D:0140  1A 1A 1A 1A 1A 1A 1A 1A-1A 1A 1A 1A 1A 1A 1A 1A
378D:0150  1A 1A 1A 1A 1A 1A 1A 1A-1A 1A 1A 1A 1A 1A 1A 1A     . ... .      .
378D:0160  1A 1A 1A 1A 1A 1A 1A 1A-1A 1A 1A 1A 1A 1A 1A 1A   .        ...
378D:0170  1A 1A 1A 1A 1A 1A 1A 1A-1A 1A 1A 1A 1A 1A 1A 1A              ..
-W
Writing 0080 bytes
-Q
```

Listing 4

```
. Load WordStar in Letter Quality Mode
  after CONFIGURING Printer
.
CONFIGUR <WSSETUP.DAT
WSL
CONFIGUR <WSRETURN.DAT
CLS

  Computer now returned to PARALLEL
  Printer Operation
.
```

You will notice in the above batch file that I am using the "PIPE" feature of MS-DOS to tell CONFIGUR to get the input for the program from a file called WSSETUP.DAT.

Next I load WSL, which happens to be a version of WordStar that I have configured specifically for a letter quality printer.

When I have exited WordStar, the next line in my batch file is executed, which causes CONFIGUR to be run again with the input being taken from the file called, "WSRETURN.DAT", which now sets my system back to parallel printer operation.

Continued from Page 8

Of the cursor functions, I have not found any that will work although not all have been tried. Among the modes of operation codes, the codes F, G, p, and q used with ESC, DO work.

Concerning the set and reset modes on page B.17, the 1, 5 and semicolon codes DO work. I find some others do not, and I've not tried others. (The manual says nothing about the full ESC sequence needed for these codes, i.e., each must be preceded by ESC and either lower case x [set mode] or lower case y [reset mode].)

Although I have not used ZBASIC, I see that page D.9 of Appendix D lists a number of screen function codes that presumably work with that dialect of BASIC. These codes differ considerably from the H-89 escape codes with which I am most familiar. The H-89 codes originated as the widely-used H-19 terminal codes and, for the most part, were carried forward into the H-100 escape code set listed in Appendix B.

The GW-BASIC manual seems to omit explaining that line 25 must be cleared and enabled before printing to that line is possible. Nor does the Z-100 user's manual explain this — or at least I cannot find this information in either. This is a significant manual error in my view, because I, and I suppose many others, use line 25 extensively in BASIC programs.

Sincerely yours,

Elmer A. Goetsch
LtCol. U.S. Army — Retired
7524 Island View Road
Three Lakes, WI 54562-9216

Slowing The Accelerated Demise Of The Z-100

Dear HUG:

I think it's about time that someone made a lot of noise in praise of the work of Pat Swayne to slow the accelerated demise (obsolescence) of the Z-100, this in contrast to the actions and policies of Heath/Zenith. (Did you know that Heath is now surplusing Z-100 motherboards? The local surplus store has a populated board, sans memory, for $50.) Anyway, I've been using ZPC with considerable success, I'm now in the process of implementing the hardware changes which Pat suggested in the April issue of REMark, and I look forward eagerly to the issuance of ZPC version2. I need particularly to thank Pat for providing a patch which would enable my copy of SuperCalc 3, version 2.1 to run under ZPC.

While I'm at it, let me hammer on another point. I'm a reasonably experienced user of Lotus, Multiplan and SuperCalc3, and despite Heath/Zenith's failure to support the program, I urge any reader who's contemplating purchase of a spreadsheet to take a careful look at SuperCalc3. It's easier to use, by far. It will do almost anything that either of the other two programs will do, graphics are better, it will do a number of things that the other spreadsheets will not do, and it's less expensive. Also, it's not copy protected. Herewith a strong vote for SuperCalc3.

Finally, does anybody know of an MS-DOS version of the CP/M public domain catalog program, CAT? There are some MS-DOS programs named CAT, but so far all that I've examined just catalogs files on one disk, and that clumsily. In contrast, I've a collection dating back to 1978 of more than 4500 CP/M files on more than 200 disks, a mix of 5" and 8", single- and double-sided, single- and double-density. The disks are numbered sequentially, and I can find which disk contains any file in less than 10 seconds. This assumes that I haven't forgotten completely how the file was named. If that occurs, then I have to start searching and it will probably take as much as a minute using suggestive letter/number combinations plus wildcards. I would like to have a similar capability for my MS-DOS files.

Again, a vote of thanks to Pat Swayne.

Clinton J. Chamberlain
2425 Chalet Gardens Court
Madison, WI 53711

CD.ASM In Subdirectories

Dear HUG:

I enjoyed the article by Joseph Katz in the April '86 issue of REMark (Page 69). I punched in CD.ASM, assembled and linked it, and watched it bomb. Well, it works just fine if your WordStar is in the Root Directory, but I have WS.COM, et. al. in a subdirectory. Running CD.EXE this way, WS.COM cannot be found. So, you cannot return to WordStar. I tried adding the subdirectory to my AUTOEXEC.BAT PATH command with no success. Suggestions? By the way, here is the same program in Turbo Pascal:

```
PROGRAM CD;
{Change Directory, S. A. Jacob, Apr 4, 1986}
var NewPath   :      string[49],
begin
  if paramCount=0 then
     begin
       write('New Path: '),
       readln(NewPath)
     end
   Else newPath:= paramStr(1);
   {$i-}ChDir(newPath){$i+};
   if ioresult=0 then writeln ('PATH CHANGED ')
   else writeln('PATH UNCHANGED.')
end
```

Stephen A. Jacob
103 B Fourth Street
Honolulu, HI 96818

CCM Company Has A Solution

Dear HUG:

While the EasyPC and Gemini boards, which let Z-100 computers run IBM-PC software, have been a boon to many users, they have created problems for people who have added other hardware to their machines, such as speed-up kits, non-standard memory expansion and real-time clock calendar boards. The problems have been with both hardware, where it is sometimes physically impossible to fit everything in, and software, where there can be I/O port and memory address conflicts.

The CCM company has a solution to users who want both the IBM-compatibility boards and the real-time clock calendars: Our AUTOCLOK board keeps the date and time when the computer is off so that the DOS will automatically know the time and date when the machine is turned back on. Because it plugs into the S-100 bus, there is no hardware conflict with the EasyPC and Gemini boards. We supply software which works in both the Z-100 modes and the IBM modes for the EasyPC. If you own the Gemini board, the software will work in the Z-100 mode, but due to a design decision by the makers of the Gemini board, it will not work in the IBM mode. I called Gemini Technology and they said that their software in ROM intercepts all I/O port accesses, even

those not meant for IBM devices, and this is what causes the trouble, since AUTOCLOK is port addressed. They did say, however, that they will release a new "Version 2" ROM which will provide a way around this problem soon.

The AUTOCLOK hardware/software package sells for $185 ($55 for software only). It includes the clock/calendar boards, a useful collection of time management utilities (with Turbo Pascal source code) and the SSE text editor. Contact:

Peter Halverson
The CCM Company
1308 East 8th
Tucson, AZ 85719
(602) 622-2796 (weekends, eves)
(602) 622-2058 (days, ask for Peter)

Screen Dump To Epson Printer

Dear HUG:

The April 1986 issue of REMark contained an article entitled "Z-100 Drafting Program for Scientific Graphs" by Michael D. Chafetz.

After a few typographical error corrections in the listing, I found the program to perform exceptionally well for all CRT operations on my Z-100 with color monitor. As the original screen dump protocol was designed to go to a Star Gemini 10X pritner, and I work with an Epson LP-1500, it became necessary to change a few statements in the algorithm. For a screen dump to the Epson LQ-1500, the routine should be:

```
535 DIM P(639):DIM=1
540 LPRINT CHR$(27);CHR$(64)
542 LPRINT CHR$(27);CHR$(51);CHR$(24)
545 FOR X=1 TO 637:V=0:FOR Y=0 TO 7:
548 V=V+-1*(0<>POINT(X,Y+D))*2^(7-Y):NEXT Y
550 IF V=9 THEN LET V=8
553 P(X)=V:NEXT X
555 LPRINT CHR$(27);CHR$(90);CHR$(125);CHR$(2),
560 FOR N=1 TO 637:LPRINT CHR$(P(N));:NEXT N
565 LPRINT CHR$(13)
570 D=D+8:IF D=>212 THEN 40 ELSE 545
```

Comparison of the Gemini printed graph in the article to my printed graph from the Epson shows that the Gemini gives a much denser copy with double-density dot graphics than the Epson with quadruple-density dot graphics. Also, my graph prints out much smaller than what is shown on the monitor. Possibly other users of the Epson LQ-1500 printer may have some input to these observations with regard to this program.

Sincerely,

David A. Chapman, Research Associate
Dairy Breeding Research Center
Penn State University
University Park, PA 16802

Scrolling In GW-BASIC

Dear HUG:

It seems strange that GW-BASIC on my Z-150 PC does not make provision for scrolling through a BASIC program by using the PgUp and PgDn keys. Fortunately, scrolling can be implemented quite easily by defining two function keys.

An undocumented feature of GW-BASIC is that pressing Ctrl-X displays the previous statement of a BASIC program, and pressing Ctrl-Y displays the next statement. By defining two function keys as follows, you can scroll half a screen at a time:

```
10 'List up = cursor up + previous 12 lines + cursor down
20    KEY 8,CHR$(30)+STRING$(12,24)+CHR$(31)
30 'List down = cursor up + next 12 lines + erase to end
40    KEY 10,CHR$(30)+STRING$(12,25)+CHR$(26)
```

To use these keys, you first place the cursor on the line below the desired starting line. Since this is the natural position of the cursor after pressing one of these function keys, you can scroll through a program by pressing the key several times.

Another missing feature is the ability to insert a blank line on the screen before adding a new statement at its logical place in the listing. This can be accomplished as follows:

```
50 'Open line = cursor up + line feed + backspace + return
60    KEY 9,CHR$(30)+CHR$(10)+CHR$(8)+CHR$(13)
```

These definitions can be placed at the top of your application program, or in a separate program to be run before you begin editing.

Sincerely,

Richard A. Harris
1903 Baker Avenue
Utica, NY 13501

85° And No Air Conditioning

Dear HUG:

Has anyone else had this problem — and found a solution? When the weather warms up, my hard disk flops. (Warm is over 85, no air conditioning; the drive is a Miniscribe model 2012, P/N 40002014 in a ZW-111-32; the data separator card is P/N 85-2818-1). I wish I'd gotten the machine in July instead of October, then I'd have found the fault under warranty! At any rate, after about an hour or so of operation, I get a read error on drive E: — sector address 0035 is a common message. Shutting down for 15 to 20 minutes cures the problem (of course, any data not saved is lost, but luckily WordPerfect makes auto saves every 5 minutes the way I've got it configured, and I save often when I'm using the Turbo Pascal editor, so I don't lose much.

My local Heath/Zenith dealer has problems with the data separator card on one of his machines — he has a wad of tissue under one part to keep it flexed up a little, and it seems to work! He gave me a can of freeze spray, and the next time the hard drive flopped, I popped the lid off the low-profile machine and gave the general area of the data-sep card a squirt — and the retry worked! Successive checks on the next few failures seemed to narrow the problem area to the area of U46 and U47 at the center back of the card. These are TI chips, an AM26LS31CN and an AM26LS32ACN (the latter made in Taiwan). I never got any schematics or other data separator board documentation with my machine, so I don't have any idea if these chips, or what looks like a resistor network just behind U47, could be at fault. If anyone has any ideas or has had a similar experience, I'd be glad to hear about it. Also, how warm should a hard drive's case be getting? Mine gets quite warm to the touch. A final note: operating with the lid off greatly extends the time between failures — it has to be up in the 90s outside before it starts to give any problems.

Other than this I've had no complaints. In fact, using various IBM keyboards (PC, PC-AT, 3270, 5191, etc.) has made me really ap-

preciate the Z-100. I'm looking forward to getting my ZPC v.2 upgrade — even though I'll still have to cripple the keyboard to emulate big blue's, at least I won't have to look at that PC character set anymore! Thanks a lot, Pat Swayne.

Steve Caple
1150 Arcade Boulevard
Sacremento, CA 95815

CF: A Configuration Utility For The H/Z-100

Dear Mr. Kalis:

While reading the April 1986 issue of REMark, I began thinking about your statement in the first paragraph that you can't make changes to the Z-100 configuration with a batch file. Below is the batch files I use to change which printer I use. One is a Sears Communicator 3 with computer interface and the other is an Okidata 82a. This is a series of four files. Two batch files and two data files. First, the batch file for letter quality printing:

```
CONFIGUR <+-DATA+-LETTER.DAT >NUL:
```

Next, the data file for letter quality:

```
AA
HEE
```

Now the batch file for the draft printer:

```
CONFIGUR <+-DATA+-DOT.DAT >NUL:
```

And last, the data file for the draft printer:

```
AJBNNNNAOEAND
0

HEE
```

The batch files do not allow you to see the current configuration, but do allow you to make changes by only typing one word or letter. This idea originated by Mr. Pat Swayne at HUG. He used the batch and data files to patch software to run under ZPC. Although I probably won't be using your CF program, I did enjoy reading the article since it does help me understand how assembly language programs work.

Regards,

Tommy J. Creek
3416 Ridglea Court
Del City, OK 73115

ZPC And WordPerfect

Dear HUG:

I am writing this with my new WordPerfect program on a Z-121 using Pat Swayne's ZPC emulator program. Besides my delight with this fine word processor, I have all the advantages of the Z-100 machine, i.e.

- 4 floppy disk drives (since I am using the Z-100 IO.SYS)
- a readable Z-100 font
- a comfortable Z-100 keyboard
- 704K of memory
- compatibility with the P-SST board, including its clock
- close to 8 MHz operating speed
- full use of the PERKS program

I have to face the following disadvantages:

- must use the F0 and HELP keys as CTRL and ALT with all special keys
- must make up my own template at special key functions
- a slight loss of speed over a true 8 MHz (at least as compared to a Z-158 with a hard disk)

Pat Swayne deserves the Software-of-the-Year award. I might add that I also have the Gemini board installed, and so far the only thing I really need it for is to run the Flight Simulator. WordPerfect on the Gemini is slow and worse, it has occasional unexplained crashes.

Although I have a color system, I find that I have to use WordPerfect under the PC 7 (monochrome) mode. The hardware ZPC modification will be necessary, but the program runs faster monochrome, so I think I'll leave it that way.

A note about the PERKS program. It works best if it is loaded after you run ZPC, although upon exiting a few spurious, but temporary, characters remain. If you load it before loading ZPC, you will find an annoying second cursor cluttering up the screen. You should disable the screen-saver clock in PERKS, because it is incompatible with the screen operation under ZPC-PC. Curiously, even with the PC set to monochrome, PERKS comes up in full color.

Sincerely,

William N. Tavolga
5151 Windward Avenue
Sarasota, FL 34242

ZPC Patch Problem

Dear HUG:

A comment on ZPC news in the March issue. The patch described for Turbo Pascal 3.01 has a problem, at least on version 3.01A with ZHS installed. That is, the compiler seems to work, but whenever the Turbo-editor is called, the program exits to hyperspace. This is particularly annoying because the editor is one of Turbo's best features, especially when you come up in the editor after an error, the cursor is pointing at the problem.

Keep up the good work.

Seth Neumann
2712 Katrina Way
Mountain View, CA 94040

Interested In Ada Language Compiler

Dear HUG:

Let me begin by first saying that I thoroughly enjoy REMark magazine and the articles on the Z-100 computer.

I am interested in obtaining an Ada language compiler for the Z-100. I would appreciate any help from ZDS or fellow HUG members in finding a source for this item.

Sincerely,

Donald R. Sentowski
P.O. Box 3124
APO 09057-5370

*

Continued from Page 43

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
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	36
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
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

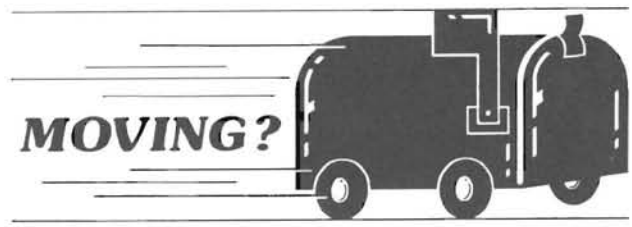
NOTE: The [-37] means the product is available in hard sector or soft sector. Remember, when ordering the soft sectored format, you must include the "-37" after the part number; e.g. 885-1223-37.

Continued from Page 42

- Multiplication
- Output to Printer (with titling)
- Row-Reduction
- Summation and Subtraction
- Transpose
- Disk Reading and Writing (ASCII or binary)

Comments: MATT was written with ease-of-use and speed in mind. The menu items are designed to be mnemonic. For instance, "M" performs multiplication, "S" performs summation, etc. In short order, the new user can be performing matrix operations without needing to look at the menu. "One touch" menu response, spreadsheet type matrix entry and editing, and input "garbage filters" make MATT very friendly. Support for the 8087 numeric coprocessor can be added by simply recompiling the source using Turbo-87 from Borland International, Inc. Z-100 owners having the MOUSEPACK by Paul F. Herman will find the spreadsheet editing compatible with their mouse system.

TABLE C Rating: (10)



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

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

CUT ALONG THIS LINE

HUG MEMBERSHIP RENEWAL FORM

HUG ID Number: _____

Check your ID card for your expiration date.

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

Name _____

Address _____

City-State _____

Zip _____

REMEMBER - ENCLOSE CHECK OR MONEY ORDER

CHECK THE APPROPRIATE BOX AND RETURN TO HUG

	NEW MEMBERSHIP RATES	RENEWAL RATES
U.S. DOMESTIC	$20 ☐	$17 ☐
FPO/APO & ALL OTHERS*	$35 ☐	$30 ☐ U.S. FUNDS

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

Heath/Zenith Users' Group

Hilltop Road
Saint Joseph, Michigan 49085

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

POSTMASTER: If undeliverable, please do not return.

P/N 885-2081