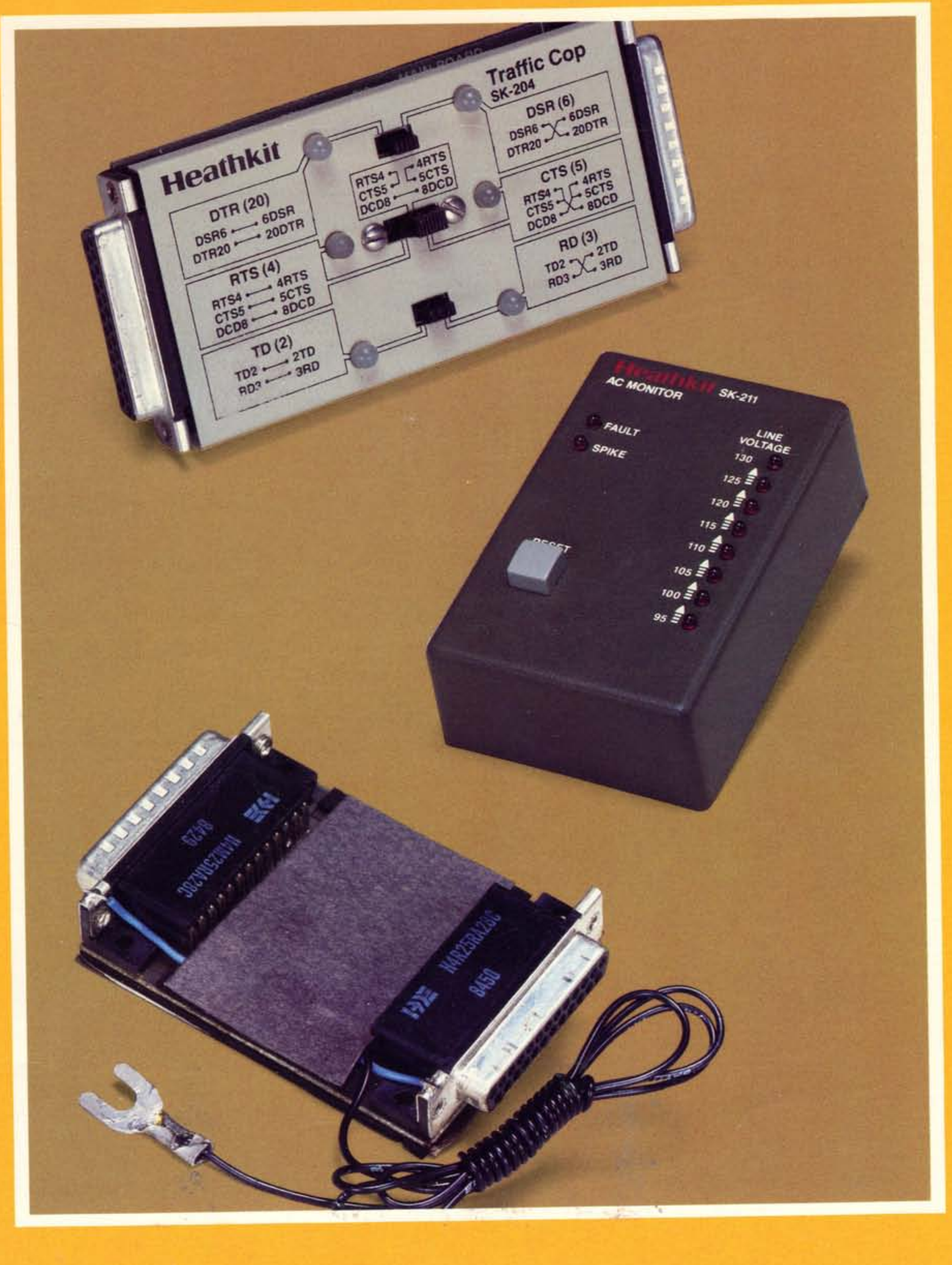
$2.50

REMark®

Volume 8, Issue 2 • February 1987

P/N 885-2085 Issue 85

Official magazine for users of HEATH ZENITH computer equipment.



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.

G·G GEMINI
technology corp.

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

IBM and IBM/PC are registered trademarks of International Business Machines
Zenith and Z100 are registered trademarks of Zenith Electronics Corp.

"Have Your Computer Talk To Ours!"

"HUGPBBS is on line 24 hours a day, with over 20 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!"

NOW MULTI-USER!

Heath/Zenith
Users'
Group

Managing Editor .... Jim Buszkiewicz
(616) 982-3837

Software Engineer .... Pat Swayne
(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	$22.95	$37.95*
Renewal	$19.95	$32.95*

* U.S. Funds

Limited back issues are available at $2.50, plus 10% shipping and handling — minimum $1.00 charge. Check HUG Product List for availability of bound volumes of past issues. Requests for magazines mailed to foreign countries should specify mailing method and appropriate added cost.

Send Payment to: Heath/Zenith Users' Group
Hilltop Road
St. Joseph, MI 49085
(616) 982-3463

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

Articles submitted by users and published in REMark, which describe hardware modifications, are not supported by Heath/Zenith Computers & Electronics Centers or Heath Technical Consultation.

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

REMark®

Official magazine for users of HEATH ZENITH computer equipment.

Volume 8, Issue 2 • February 1987

Index of Advertisers

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

On The Cover: *Pictured are three of Heath Company's new Starter Kits: The RS-232 Line Protector (SK-202), The Traffic Cop (SK-204), and The AC Monitor (SK-211). For a complete review of these kits see Page 75.*

IN MULTIUSER COUNTRY, ALL ROADS LEAD TO SCO

SCO XENIX® SYSTEM V

In the heartland of multiuser country, you're licensed for up to 16 users on IBM® PC AT® or PC XT®, respective compatibles, or AT&T PC 6300 Plus! Discover unprecedented price performance per user. Enjoy Multiscreen™ vistas in breathtaking **setcolor.** Explore DOS cross-development. Protect your investment with 8086/80286 and XENIX 3.0 applications compatibility. And bring along your widest range of peripherals!

SCO XENIX-NET XENIX-DOS LAN

Travel through SCO XENIX-NET and witness the future of office computing: networked DOS *and* XENIX PC workstations. Visit a unified environment that is both "multiuser" *and* "LAN." Watch XENIX and DOS systems serve each other and share resources — each doing what it does best — without sacrificing the strengths of either.

SCO uniPATH™ SNA 3270

Gateway to mainframe territory, SCO uniPATH is a must-see for both VARs and MIS/DP managers. Experience the real power of PCs in corporate distributed processing. Build customized applications with programmatic interfaces, and let your networked multiuser PCs process mainframe data – with session hold and multiple sessions per user, of course.

HOT-LINE SUPPORT DOCUMENTATION USER TRAINING NEXT EXIT

SCO's multiuser traveler's services are unrivaled, and start with SoftCare℠ Support, including a toll-free hotline! You can even enjoy 30 days of Introductory SoftCare with the purchase of each product – on us! SCO's excellent documentation is considered a standard for multiuser road guides. And a wide variety of training courses is available for all SCO products.

SCO applications available for all popular UNIX/XENIX systems.

XENIX and Multiplan are registered trademarks of Microsoft Corporation. • IBM, AT and XT are registered trademarks of International Business Machines Corporation. • Multiscreen, SCO Professional and Lyrix are trademarks and SoftCare is a service mark of The Santa Cruz Operation, Inc. • 1-2-3 and Lotus are registered trademarks of Lotus Development Corporation. • dBASE II is a trademark of Ashton-Tate. • uniPATH is a trademark of Pathway Design. • UNIX is a trademark of AT&T Bell Laboratories. • FoxBASE is a trademark of Fox Software, Inc. • Informix is a registered trademark of Relational Database Systems, Inc. 6/86

The Santa Cruz Operation, Inc., P.O. Box 1900, Santa Cruz, CA 95061

SCO Professional™ 1-2-3 WORKALIKE

Take this exit, and you'll swear you're on Route 1-2-3® – until you notice enhanced features and multiuser access that Lotus® users only dream about! But don't worry about culture shock – if you like 1-2-3, you'll *love* SCO Pro. Read and write to 1-2-3 DOS files, enjoy full 1-2-3 functionality and macro support, and rest easy with file locking data security.

SCO FoxBASE™ dBASE WORKALIKE

As with SCO Professional, another *deja vu* awaits you on this route – this time, dBASE II™ – but once again, it's better, faster, and multiuser! Bring along your dBASE II data files, macros, and expertise – they all work here, too! You'll be sharing dBASE II-compatible data with friends, in record and file locking comfort, right out of the box.

SCO Lyrix™ WORD PROCESSING

Built especially for UNIX™, "what you see is what you get" SCO Lyrix is a favorite of both new and veteran multiuser travelers. Fully user-configurable, it's a natural attraction to foreign language visitors, as well as system administrators. You'll appreciate advanced WP features such as spelling corrector, and automatic footnotes and section numbering, as well as direct menu access to UNIX and other destinations.

SCO SALES - INFO 1-800-626-UNIX

Call us today to order any of these and other exciting SCO products such as Multiplan®, Informix®, languages, and business graphics. VARs, ISVs and other qualified customers can "mix and match" for cumulative discounts across the entire SCO product line. We look forward to booking your next multiuser tour!

SCO
THE SANTA CRUZ OPERATION

(800) 626-8649
(408) 425-7222
TWX: 910-598-4510 SCO SACZ
uucp: ...decvax!microsoft!sco!info

BUGGIN' HUG

Two Parallel Interfaces For The H-89

Dear HUG:

The article "Two Parallel Interfaces For The H-89" by Pierre D. Olivier, III (REMark, Volume 7, Issue 10 — October 1986) discussed the Quick-P parallel printer interface for the H-89 from McGaffey Engineering. The article was generally favorable to the Quick-P. (It should have been — the Quick-P is an excellent product which lets you use nice printers like the Epson FX-85, discussed in another article in the same issue!) However, the article included the statement: ". . . printer interrupts are not supported and must be jumpered 'off' on the serial board. This means you can't use the REACH modem program. . . . if you use REACH, be forewarned."

The first part of that statement is true — the printer interrupt must be jumpered "off". The second part is totally incorrect: REACH runs just fine with the Quick-P, with either HDOS or CP/M, and the printer is still supported! It sounds to me like Mr. Olivier made an assumption, and didn't bother to test it.

For a detailed discussion of the Quick-P and other attractive H-89 upgrades, watch for my article, "Upgrading The H-89 . . ." in a forthcoming issue of "Sextant".

Sincerely,

W. K. Clarkson
Comptographics
1612 Via Arriba
Palos Verdes Estates, CA 90274

MATT Fix

Dear HUG:

I would appreciate your publishing this minor correction to my program, MATT, which was released in the October issue of REMark. Both the Z-100 and PC versions have a flaw which causes the program to crash if you are editing a matrix larger than the screen display, and you attempt to move the element cursor off-screen. Two easy fixes are possible. The simplest, which requires no modification to code, is to simply recompile the source (MATT.PAS or PCMATT.PAS) using your version of Turbo Pascal from Borland. Follow this procedure:

1. Load Turbo
2. Log onto the drive containing the source code and include files using the "L" option in the Turbo menu
3. Select "O", Then "C", then "A" from the menu
4. Specify 1000 at the prompt for maximum stack size
5. Select "Q", then "C" to compile
6. Type MATT or PCMATT for the Z-100 or PC series, respectively for the name of the file to compile
7. Select "Q" to quit the Turbo environment and return to DOS

Problem solved! Originally, the executable files were incorrectly compiled to a stack of 0B00 paragraphs instead of 1000 as above. This is fine if you have limited the maximum size to 20 or less as discussed in the documentation.

The second fix requires a few minor changes to your version of the main program source code, either MATT.PAS or PCMATT.PAS. This fix has the advantage of making MATT a lot less "memory hungry", and may make the element cursor response a little snappier. The changes are as follows.

Line numbers in MATT.PAS, PCMATT.PAS to change:

205, 228:	Procedure Draw(M:Mat2; Rs,Cs,Rf,Cf:Byte);
change to:	Procedure Draw(var M:Mat2; Rs,Cs,Rf,Cf:Byte);
243,264:	Procedure Point(M:Mat2; var Rs,Cs,R,C,I,J :Byte);
change to:	Procedure Point(var M:Mat2; var Rs,Cs,R,C,I,J :Byte);
641,662:	Procedure Disk__Save(M:Mat2; R,C:Byte; L :Char);
change to:	Procedure Disk__Save(var M:Mat2; R,C:Byte; L :Char);
731,752:	Procedure Cursor(M:Mat2; var Rs,Cs,R,C,I,J :Byte);
change to:	Procedure Cursor(var M:Mat2; var Rs,Cs,R,C,I,J :Byte);
811,832:	Procedure Jump(M:Mat2; var Rs,Cs,R,C,I,J :Byte);
change to:	Procedure Jump(var M:Mat2; var Rs,Cs,R,C,I,J :Byte);
844,865:	Procedure Mod__Dim(M:Mat2; var Rs,Cs,R,C,I,J :Byte);
change to:	Procedure Mod__Dim(var M:Mat2; var Rs,Cs,R,C, I,J :Byte);
938,960:	Procedure Prn(M:Mat2; R,C:Byte; L:Char);
change to:	Procedure Prn(var M:Mat2; R,C:Byte; L:Char);

Having accomplished these changes, you may now recompile the source using the procedure discussed in the first fix, except now you can get by with a maximum stack size of 600 paragraphs instead of 1000.

My apologies for any inconvenience this may have caused fellow HUG members.

I believe MATT will prove to be very useful to high school students, teachers, engineers, scientists, or anyone else who has a need to handle arrays of numbers.

Dennis K. Greer
1635 Westbrook Drive
Xenia, OH 45385

Installing A Reset Switch And Patching Disk Parameters

Dear HUG:

Just a note to thank you for two short recent articles which I have found very useful and to point out to your readers some additional information concerning them.

The first was Jim Buszkiewicz's May 1986 (p.77) note on installing a reset switch on the Z-241. One of the many reasons I like the Z-100, as well as the AT&T 6300 computers is the hardware reset capability. So having it now on the Z-241 is really a useful addition to the machine. I had only one real problem and that was in trying to order Heath's 64-685 momentary contact switch. Unfortunately, it is no longer available. So I have substituted an Archer 275-1547 miniature push button switch which can be obtained in packages of 5 from (ugh!) Radio Shack for $2.49. I mounted it on the removable 3" × 4" panel on the back of the machine in a 1/4" hole I drilled there. Although a bit of soldering is required, it is quite easy to do thanks to Jim's clear diagram. Just loop the wire around the leads and tack solder them down with a small iron. (If anyone needs a switch, I have 4 left over!)

The second item about Pat Swayne's article on patching disk parameters in MS-DOS, August 1986, pp. 17-20. For those of you wanting to patch the two bytes starting at location 0:579 on a regular basis, you can use one of Pat's two programs, or you can use input redirection in DEBUG. It does not generally seem to be well-known that DEBUG uses DOS function calls for input and output. Hence, one can execute a series of commands in DEBUG by having them read from a file rather than from the keyboard. To load the values suggested by Pat into my Z-241, I execute the program:

```
DEBUG<patchdsk
```

from AUTOEXEC.BAT. Patchdsk contains the single line:

```
e0:579^M00 02^Mq
```

Here, ^M represents a carriage return. This sequence says: examine the bytes beginning at 0:579, changing the first to 00 and the next one to 02, then quit. If you have trouble in putting a ^M into your editor, then the three lines

```
e:0:579
00 02
q
```

will do the trick, but the display is not as neat because of the extra linefeed. Of course, you can always redirect the output to the NUL device and not see anything during the run with the command:

```
DEBUG<patchdsk>NUL
```

The technique here has wider application than just patching disk parameters during reboot.

Bill Hall
AT&T Information Systems
Court N24A
Pine Tree Boulevard
Old Bridge, NJ 08857

New H-8/89 Newsletter

Dear HUG:

Mr. Hank Lotz is starting a quarterly newsletter entitled "The Staunch 8/89'er". He is asking $5.00 for the four issues per year. He is asking all prospective customers to fill out a questionnaire so he can get to know his audience. You may either write to him and ask for the questionnaire, or send $5.00 and he will send you the questionnaire. The address is:

Hank Lotz,
2024 Sampson Street,
Pillsbury, PA 15221

NAVPROG8 Correction

Dear HUG:

I was going over the NAVPROG8 program the other day and I noticed that I had included an error in the fuel consumption calculations in the NAVPROG7 program. Actually, I had neglected to make a correction that the original author Alan Bose had sent in earlier. The effect of the error is to understate the fuel consumption during the climb phase of the flight. The correction is as follows:

```
              NAVPROG7.BAS
1470          .....K(I)=KU:KC=KU:CK=CK+KU:.....
         should read:.....K(I)=KU:CK=CK+KU:.....
1490          .....K(I)=KU:CK=CK+KU:.....
         should read:.....K(I)=KU:KC=KU:CK=CK+KU:.....
```

I apologize for the oversight and hope it has not caused anyone using the program a problem.

Yours very truly,

Peter T. Ambrose
5119-139 Street
Edmonton, Alta.
CANADA T6H 3Z4

Roots II: Editorial Omissions

Dear HUG:

Thank you for publishing the first installment of my article, Roots II: A Dream Program For Genealogists in the November 1986 Issue of REMark. There are several items I would like to have passed on to your readers.

First, there were some understandable editorial omissions regarding LDS beliefs. Unfortunately, genealogy is deeply indebted to and intertwined with LDS beliefs. A complete explanation of those beliefs would fill several columns of print and probably offend some readers. My original material contained an explanation of the doctrine behind the LDS involvement in genealogy. For those readers that are interested, I will be happy to forward that omitted material to you on an individual basis. It makes for interesting reading and helps explain why the LDS do genealogy. Please make your request to me c/o HUG.

The second item is my mistake. The address of Commsoft changed about a year ago and I inadvertently sent the old address with the articles. I sincerely hope this did not inconvenience anyone that tried to get in touch with the company. The correct address should read:

Commsoft
2257 Old Middlefield Way
Mountain View, CA 94043

Third, I make some rather pointed statements in my article about the Split routine. After using the routine again (I finally reached the 4095 capacity of the program more precisely by temporarily deleting parents where I wanted a break to take place in the family tree. Needless to say, I was much happier with the results than I indicated in my article.

Finally, I have been informed that Commsoft is readying version 2.0 of Roots II. Howard Nurse, the president of the company tells me that it encompasses changes in the way data is handled and incorporates new search abilities and printed forms. I am also told that there are improvements to the old forms, although most of them are great the way they are now.

Sincerely,

Tom Huber
Heath Company
St. Joseph, MI 49085

Thoughts On Softstrips

Dear HUG:

I have just discovered something else I like about REMark columnist Joseph Katz: he is so blatantly, embarrassingly honest. No man with larceny in his heart would ever have made the statement he did in the December 1986 Issue of REMark to the effect that drawing a red line through a Softstrip would copy protect it. True, doing that does put a lockout on using a Xerox machine, but that wouldn't stop any amateur photographer with a camera, enlarger, flat field lens and a red filter. Putting a rainbow of lines through the Softstrip would take the services of a professional photofinisher, but I think that could be copied too. But why bother? There is such a silly way around all of this. Remember that a Softstrip cannot be altered in any way except to destroy it. The simple act of reading it with the Reader does not, cannot, change it at all. Now then, what does the act of reading a Softstrip do? It transfers the contents of the strip onto a floppy without changing the strip. What, then, is to prevent our putative Captain Hook from simply putting in a new floppy and pushing R again? And again, and again, or maybe just using DISKCOPY on the first floppy? No receiver of stolen goods (i.e. user of pirated software) would blink at using a floppy rather than a Softstrip. Rather he would welcome it, as he wouldn't then have to buy a reader.

The only way you can copy-protect a Softstrip, then, is to arrange it so that its contents cannot be transferred onto a floppy. This would mean that to use the program, a user would have to load it from the Reader each time. Maybe I am being naive now, but I simply do not think that any user would put up with that, especially if it is a multiple strip program. Such a copy-protected monster would never sell, even for $6.95. Therefore, I maintain that it is going to take one heck of a genius to arrange a copy-protection scheme for Softstrips.

The rest of the article was fine, and well past due. I have had a Cauzin Softstrip(tm) system for some months now connected to my Z-200. I would like to share some of my thoughts on the system with you. I bought it for the sole purpose of reading in programs published in magazines. All of the other so-called advantages evaporate in the light of practical experience. Mr. Katz's article didn't stress this enough, I feel. Few of us work around cyclotrons, NMR spectrometers, or humongous radar installations and also desire to carry in software, so why worry about stray magnetic fields? Archiving of files takes too long via Stripper(tm) and would never be done. Archiving a 20MB hard disk on paper would take WEEKS. Forget that! Who owns an Apple and a Zenith and still desires to use the Apple? So much for the machine-machine transporting. Marketing a commercial program requires a royalty payment to Cauzin which makes it financially advantageous to stick with floppies, which can be had to $.60 each. Paper can be mechanically damaged just as easy as a floppy — no advantage there. Storing your software in a hot environment would make Softstrips attractive, but where, except in a car, does anyone use a computer in an environment hot enough to melt a floppy? Cauzin does not yet market a portable Reader, although there is nothing to prevent it, so that advantage isn't here yet. Where does that leave us?

It leaves us with one enormous benefit: downloading a public domain program from the printed page of a magazine such as REMark. I cannot tell you the number of times I have seen a marvelous program published in any number of places, but found that it was written in C or Fortran, or some other such foreign language and, thus for me, useless. A few were in some language I do understand, but started on page 16 and continued on without a break until page 124. Inputting and debugging such a monster was out of the question, no matter how useful the program. What an aid a Softstrip would be in such a situation! Alas! Where are the Softstrips? Has Cauzin's dreadful royalty demands scared away all the publishers?

Mr. Katz mentioned a problem that might occur if the reader read in the first of a number of files and balked at the last one, forcing him to start all over. That is a minor problem that I am willing to overlook, since it takes only a few minutes to read in the whole thing again. MUCH worse is the equivalent in printing a program. I had my Epson barf up garbage an HOUR into printing one program! 6 pages of perfect strips, then one bit slips 'twixt computer and printer and the whole shebang heads for the dumpster. Talk about unacceptable! I could find no way to restart printing at strip #50. Therefore, I will never again try to print out any program that takes more than one page to print. That about cooks it for the Stripper program. Unless and until Cauzin fixes this bug.

Well, I am taking up too much of your time. The bottom line here is that I am asking you to begin publishing listings in Stripper format in REMark articles. The advantage of being able to read in any published program without effort (or much of it) and in any language is too much to ignore. We Heath Users' Group members would benefit enormously from this. Please! The Catch-22 of not having published Softstrips until readers have the Cauzin system and readers not buying the system until they see published Softstrips is analogous to the situation that existed with digital disks. That didn't hurt the acceptance of this new audio medium, but that was because the record companies took the risk of publishing Compact Disks before many listeners had players. Magazines will have to do the same for Softstrips, if the system is not to suffer crib death.

Maybe someday the publishing of listings in human (only) readable form will be a thing of the past. Who will mourn its passing?

Yours truly,

William G. Nabor
27172 Huerta Street
Mission Viejo, CA 92692

Clarifying Issue On UCI RAM Board

Dear HUG:

I have read several complaints about using the UCI RAM board with MS-DOS 3.10 on the Z-100 and am writing to clarify the issue.

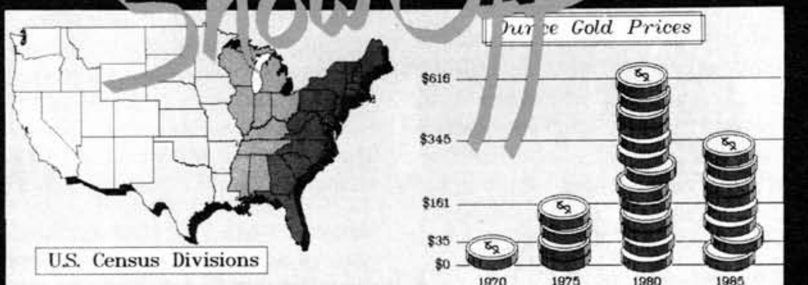
The problem is not with the UCI RAM board, but is caused by a change in the BIOS making it specific to Zenith hardware.

Versions of MS-DOS starting with about 2.19 (I am not sure exactly when it was put in, but 2.22 definitely has it) have code (in two places) to initialize the parity checking hardware on the Z205 RAM boards (either of them, no less! The Z-100 won't even hold that many!)

Since the UCI RAM board uses more bits in the control port than the Z205, the BIOS code actually disables the UCI board completely. UCI has chosen to fix the problem by changing their RAM-

Continued on Page 74

NEW!
Load Sidekick & Smartkey Above DOS


U.S. Census Divisions
Ounce Gold Prices
$616
$345
$161
$35
$0
1970
1975
1980
1985

Of Mice And Men

Jack W. Bazhaw
900 - 13 Street
Bellingham, WA 98225

Recently I added a third drive to my H/Z-150 — a 20 Meg hard disk. Ever since I had gotten my 150 I had been keeping an eye on the price of hard disks, hoping they would keep coming down until they reached what I felt was an affordable price. The bug to get one was really planted when my boss demonstrated his 150, using a hard disk. Shortly thereafter I placed an order. Now, after just a couple days of use I know I would never be happy with a computer without a hard disk. I liked retaining my two floppy drives. The standard 3-1/2 inch half-height hard drive I obtained was also more economical than several of the new drive-on-a-card type.

Looking at the H/Z-150 unit, it does not appear to have room for a third drive. However, apparently it was designed to hold one half-height floppy and a full-height hard drive. With two half-height floppy drives installed there is still room under the lower drive to tuck in a half-height hard disk — invisible inside the machine.

If everything is on hand it will take about 30 minutes to install the drive. Since the drive will be hidden inside, I put the drive indicator light on the front panel of the H/Z-150 and that added another quarter hour. In order to power all three drives, I built a Y-connector using Calectro F3-270 connectors. They require some filing in order to fit. The Y-connectors are also available ready made, but I didn't realize one would be needed and did not want to wait several days again for the postman.

After removing the cover (and power cord), front panel and the cables from the floppy drives, I removed the chassis holding the drives as a unit with the drives still attached. The 3-1/2 inch hard disk slipped in right under the lower drive; it was not necessary to remove either floppy. The mounting holes are even in the right place on the chassis.

Before fastening the drive in place, I unplugged the indicator lamp. Included with the drive was a full height bezel. I removed the indicator light from it including the attached cable and plug. Then I cut the cable in the middle and soldered in a 10 inch extension to each wire. The indicator light in the half-height bezel attached to the drive was not used.

To put the indicator lamp in the panel, I drilled a hole then used a square file on the soft plastic to make it fit the lamp holder.

Reassembly required some juggling. The little plug for the drive indicator light had to be inserted before fastening the drive into the chassis. This left the front panel dangling while fastening the drive in the chassis and putting the chassis back into the main assembly. The front panel was re-attached last.

The half-card controller was inserted in a slot and the flat ribbon cables to the hard disk were connected as were the original ribbon cables to the floppy drives. Only two power connectors for the disk drives were available. I used the one that had powered B drive to run the hard disk. The two floppies were powered from the A drive cable using the Y-cable mentioned earlier.

After putting the cover back on, the disk was ready for formatting. I started the PREP command (DOS 2.1) at 9 PM. The manual said it would take about 45 minutes per 5 megs of disk capacity to make 7 read/writes on each sector. When the train whistle woke me at 4 AM, I took a peek and it had finished with no bad sectors found.

The next few days were spent putting files on the drive and setting up the sub-directories.

It didn't take long before the drive was half filled so the decision for the larger (and more expensive) 20 meg versus 10 meg paid off. The increased convenience (and speed) has sold me on hard drives. The ability of the H/Z-150 to accept the third drive with so little fuss was a bonus.

As soon as I started using the hard disk two things became obvious: a) I liked not having to mess with floppies; and b) there was a lot more typing needed to run programs.

My first attempt to reduce the typing involved setting up batch files. This worked but required the remembering of more file names. My wife felt some of the batch file names were rather cryptic.

Table 1
Mouse Menu Definition File

```
BEGIN      run,run
autol:     MENU "SELECT PROGRAM",5,30
      OPTION     "CANCEL"
      OPTION     "Archive Files to Disk",P9
      OPTION     "Microsoft Word - Vicki",P1
      OPTION     "Microsoft Word - Jack",P7
      OPTION     "Reflex",P2
      OPTION     "Reflex Reports",P8
      OPTION     "Multiplan",P3
      OPTION     "PCTools",P10
      OPTION     "Printmaster",P4
      OPTION     "PCPaintbrush",P5
      OPTION     "END SESSION",P6
      OPTION     "CANCEL"
      MEND
      run: EXECUTE autol
P1:   TYPE       "wp" ENTER      ;start word processor
P2:   TYPE       "db" ENTER      ;start reflex
P3:   TYPE       "mp" ENTER      ;start multiplan
P4:   TYPE       "pm" ENTER      ;start printmaster
P5:   TYPE       "pb" ENTER      ;start pcpaintbrush
P6:   TYPE       "ship" ENTER "y"    ;park disk heads
P7:   TYPE       "wp2" ENTER     ;start word processor
P8:   TYPE       "db2" ENTER     ;start reflex report
P9:   TYPE       "archive" ENTER     ;copy data files
P10:  TYPE       "t" ENTER       ;start pctools
```

The hard disk had included a program (1DIR) that claimed to allow menu selection of programs, including the creation of custom menus. After reading over the manual it became obvious that this program required learning another set of commands to manipulate the menu. The program also included the ability to erase files (!) and I panicked the time I was playing with the keys and saw a cursor moving through all the file names. I had visions of a disk wipe-out.

What I really wanted was a menu system that was easy for both of us to use and could be bypassed at will. Equally as important it had to be something that I, as a programming illiterate, could handle.

Then I recalled something in my Microsoft Mouse manual about menus. There I found a procedure for designing a custom menu that did not require an overlay of another command set to learn.

Although the mouse program allows one to build all the necessary commands right in the menu file, I decided instead to use the menu to activate external batch files. It seemed easier to write the commands in batch files and changes did not require re-compiling as did changes in the mouse menu.

During bootup the mouse menu file is loaded by the autoexec.bat file. Then, after bootup, clicking a mouse button causes the menu to pop up. I would have preferred it to come up automatically but have not yet found a way. As a reminder, I added a screen prompt to click a button. After placing the cursor on the desired program name, another click starts the program running.

Table 1 is a listing of the mouse menu definition file I wrote. This file is later compiled to make the menu program. The lines starting with "OPTION" will display the text between the double quotes in the menu. When the cursor is over that selection and a button is clicked, the label at the end of the line is executed.

The labels are all defined following the end (MEND) of the menu description. Label "P1", for example, types out "wp <CR>" which calls up the batch file wp.bat, listed in Table 2. Since we use separate sub-directories for our document files with Word there is a menu selection for both of us.

Table 2
"WP.BAT" File

```
cd\light
envi d
cd \word\vicki
word letter.doc
cd\
help
```

Wp.bat sets the Turbo Lightning environment for the word processor, then starts Word with the proper sub-directory for the files. Upon exiting the word processor, the file help.bat is called up which reloads the menu. Every batch file called by the menu has "help" as the last entry. Table 3 is a listing of help.bat.

However, not all programs exit in a manner that allows the batch file to continue. PC Paintbrush, for example, will terminate but not return to the batch file that called it. In this case, the menu is reloaded by typing, "help".

The instructions in the Mouse manual for making the menu, although not designed for the beginner were sufficient, along with some samples on disk, to guide me through the project. The menu system met its design goals with flying colors.

The hard disk has really made the computer "user friendly". No more searching for the disk to load DOS, then the RAM resident programs, then the application program. Then all the disks for the next application program. An afternoon with Word, Reflex, Turbo Lightning and PC Tools no longer creates a disaster zone of floppies and jackets.

Table 3
"HELP.BAT" File

```
echo off
cls
cd\
menu auto_mnu
echo
echo CLICK A MOUSE BUTTON
echo
```

*

Still Faster Than Big Blue

A Look At The HS-248 Advanced Personal Computer Kit

Pat Swayne
HUG Software Engineer

In the March 1986 issue of REMark, Jim Buszkiewicz reviewed the HS-241 kit, Heath's version of an IBM PC-AT compatible computer. Even though this machine used the same CPU clock speed (6 MHz) as the IBM machine, it ran about 30% faster, because it accessed its memory without inserting wait states. Now, both IBM and Heath/Zenith have increased the speed of their machines to 8 MHz. Was Heath/Zenith able to maintain a speed advantage?

The new, faster Heath machine is the HS-248, and I had the pleasure of building the first one to arrive here at HUG headquarters. Construction of the computer is identical to the HS-241, and the machine's appearance is also identical. Because of this, I will not dwell on the construction of the machine in this article, and readers can refer to Jim's article for those details and for pictures.

Unlike the HS-158, which was a whole new machine on the inside when compared to the HS-151, the HS-248 is simply a faster HS-241. One noticeable change in the construction of the new machine, however, is that the fan on the power supply has been turned around, so that it blows out. There has been some discussion about H/Z-241 and 248 heating problems and the fan direction in the computer news media in the past months. Users have found on their own that the machine remains cooler if the fan blows out. An analysis of the cooling air path indicates why. If the fan is blowing in, air passes over the top of the boards in the computer, down between them, and out slots in the bottom of the cabinet front. Heated air normally rises, so the air flow is against its normal direction. If the fan is made to blow out, air is pulled in through the slots in the front of the cabinet, up through the boards, and is exhausted out the side. After thinking about this, it made so much sense that I took apart my H-151 at home and reversed its fan (see my article "Cheap Speed", in the December 1986 Issue of REMark).

Another reason for reversing the fan is that when it blows in, directly on components of the power supply, it causes dust to collect on those components. When I took my H-151 power supply apart, there was quite a bit of dust on the heat sink and other components. I've heard of TV sets blowing up because of years of accumulated dust.

The HS-248 went together without any problems, and when it was finished and I turned it on, it worked perfectly. The kit does not come with a video card, and we did not have any Zenith video cards at first, so I brought the machine up with an Everex graphics board. The Everex board is, in my opinion, a poor excuse for a video board (scrolls slowly, and flickers), but it was good enough to bring up the computer and prep my hard disk. I also tried a Paradise graphics card in it. In the text mode, that card is almost as good as a Zenith card. It scrolls rapidly, and does not flicker, but one drawback is that it does not support the 50 line interlace mode properly (see "Undocumented Features of Heath/Zenith PC Series Computers" in the November 1986 REMark). Finally I got my Z-409 video card, and now the screen flies like it should. But it's nice to know that just about any PC-compatible video card you put in an HS-248 will work.

While different video cards will work fine in the HS-248, that may not be true for other types of expansion cards. Like the HS-241, the HS-248 runs with no wait states. Memory cards, in particular, from third party vendors may not be able to work in an 8 MHz no wait state environment. You can jumper in a wait state, but that would be like driving with one foot on the brake and the other on the accelerator. The Zenith memory expansion card for the HS-248 is called the Z-445, and although it looks like the Z-405 card for the HS-241, it uses a new board that has a few differences here and there. However, we were able to get some Z-405 cards to work in the new computer by putting faster memory ICs in them, changing the position of a SIP resistor, and changing some jumpers. To fill in the base memory to 640k, you must use 256k chips in bank zero instead of 64k chips, simply because they don't make 64k chips that are fast enough.

Since the HS-248 runs without wait states, it does indeed maintain the speed advantage over Big Blue. In Jim's review of the HS-241, he reported that benchmark program he had (one of those prime number things) took 3.3 seconds to complete on the '241. On the HS-248, it takes 2.2 seconds. Jim reported that on a 4.77 MHz H-161, the test took 14.3 seconds. So you can see that the HS-248 is quite a bit faster than an ordinary PC. Now maybe it won't take so long to assemble the next version of ZPC.

First Capitol Delivers Zenith. Your Way.

Z248

Z181

Specialists in

CAD Systems
Local Area Networking

Z148

Have it your way.

If you're looking for a PC, you couldn't do better than Zenith. And you couldn't buy your Zenith at a better place than First Capitol. We'll put together a system the way YOU want it. And we'll do it for a price that competes with the guy who just sells boxes that fall on his loading dock.

Great Prices. And A Great Machine.

We're committed to Zenith. And to you. the standard Zenith models that come from the factory are a good buy. And we'll be glad to sell you one at a price that will match anyone. But if you want the full potential, we'll fix it up for you. Larger disk drives. More memory. Internal tape back up. Removable winchesters. All at a budget pleasing price.

We make Zenith Scream.

When a new model comes out, we're among the first to get it. Our Zenith fanatics take it apart, prod it and poke it until it screams for mercy. Then we put it back together (usually with some new parts) so that it REALLY screams. With power.

You'll be back.

Check with us. Check elsewhere. We'll put together a custom quotation for you we GUARANTEE you'll find interesting. We think you'll come back to us. There's a reason for the growing list of companies, government agencies, and end users that know us as the source for Zenith. Your Way.

First Capitol Computer

1106 First Capitol Dr.
St. Charles, MO 63301

1-800-TO-BUY-IT (800-862-8948) Orders and quotes

1-314-724-2336 Technical support and order status

The StockSystem Challenge.

We will meet or beat any dealer's price on new stock Zenith systems. And for enhanced systems, no one can touch our combination of value, performance, and custom service.

Before you buy . . . . . .

Call Us First. And Last.

ZENITH | data systems

AUTHORIZED SALES AND SERVICE

UnProtect: The Easy (And Safe) Way

William P. Simpson
5600 Glenford Street, #1255
Los Angeles, CA 90008

BASIC's P-option SAVE may, indeed, prevent the casual viewer from reading, listing or editing a program, but for an expenditure of less than 20 minutes, you can have a clear source copy of your own. And, after having completed the once-only preliminary steps, even that time can be reduced by half. This may not be the easiest way to recover such a file (the title only claims "easy"), but it is absolutely straight forward and there are no "maybes".

Although the procedure used for recovering programs protected by MBASIC or GWBASIC are logically identical, the details are sufficiently different to justify separate treatment in some areas. (Actually, the differences are created by the operating systems used by the two BASICs and the two different debuggers.) Rather than present them as two complete and separate procedures, they are presented together in logical steps. This avoids the necessity of repeating common information and permits those of you with knowledge of both systems to gain insight into each step by reading both procedures. Where necessary, each step will be presented in two parts, one for MBASIC and the other for GWBASIC.

If you are using MBASIC, it is helpful, but not necessary, to know how much memory is occupied by your particular version, or more precisely, where it ends when resident in memory. The easiest way to do this is to use DDT to load your BASIC into memory. Immediately following the DDT banner are two numbers: the first is an address just beyond the end of BASIC, while the second is the starting address. Make a temporary note of the ending address. (Don't be concerned about the exact end point of the program; the purpose of doing this is simply to learn approximately where the end is in order to save a little time in Step 0.2).

The first formal operation is a preliminary, but essential, step that needs to be done only once; accordingly, it is identified as Step 0. The address derived from this step will be useful whenever you want to examine a BASIC program in memory, it is, therefore, suggested that you make a permanent note of it.

Step 0.0

The one essential piece of information needed in this recovery procedure is the memory location that your particular version of BASIC uses as the starting address for the working copy of your programs. The preliminary task is to find that address. This is easily done by first loading BASIC and then writing this simple one line program: 100 a$="ZZZZZZZZ". You need not save or run it; writing it is sufficient. Now exit BASIC using the SYSTEM command.

Step 0.1

Call DDT or DEBUG. (Do NOT enter anything else on the command line.)

Step 0.2 MBASIC

Using the Dump command, D, begin an examination of memory for your program; the string of Zs will be conspicuous in the ASCII presentation given in the column on the right side of your screen. To save a bit of time, start your search a little before the location you found (in the non-essential preliminary step) to be the end of the BASIC interpreter; your one-line program will be found not too far beyond the actual end.

Step 0.2 GWBASIC

Using the Search command, S, locate your string of Zs in the code sector. For example, enter scs:0 fffe 5a 5a 5a 5a 5a 5a 5a 5a. (The fffe is not a typo; my DEBUG will not accept ffff.) You need not, of course, include all of the ASCII Zs (5a) in your string, but if you use fewer, DEBUG will present each occurrence of the shorter group. It is probable that more than one location will be found, however, the likelihood is that the last is the one you are seeking. The correct address can be verified by using the Dump command for a detailed look. The following sequence should immediately precede the series of 5As: xx yy 64 00 41 24 E7 22 (assuming your program is identical with the one given in Step 0.0).

Step 0.3

Note the exact address of the first "Z". Then subtract 8 (hex arithmetic) from that address to get the starting address of the program. RECORD THIS NUMBER; you will surely have future occasions to use it. (A discussion at the end of this article will give you other details to assist in the recognition of your program and explain why there are exactly 8 bytes ahead of the first string character.)

Step 1.0

Now that the preliminaries are out of the way, you can begin the actual recovery of the program. You know that you have a protected program because you can load and run it in BASIC, but any attempt to list it, edit it or save it results in an error message.

In order to obtain a readable, unprotected copy, the first thing that should be done is to determine the length of the protected program. Again, it is not important that this be exact but later, when you transfer the recovered program back to disk, you want the assurance that you are not transferring too little.

Step 1.1 MBASIC

Exit DDT with a CTRL-C and then load the protected program into memory again using DDT, i.e., include the name, complete with extension, on the DDT command line. (If you are familiar with the I and R commands of DDT, then, of course as previously mentioned, DDT responds by identifying itself on the first line while presenting on the second line the ending and beginning addresses for the program you specified. The first number is not the exact ending address but is a higher address somewhat beyond the actual end. The difference of these numbers is the length of your program (plus a little bit more). Note this length for future use and then exit DDT by entering CTRL-C.

Step 1.1 GWBASIC

Using DEBUG, load the protected program into memory by including the name and extension on the command line. DEBUG will store the length, in hex bytes, in the CX register. Observe this number by entering RCX at the DEBUG prompt. Record the length for future use and exit DEBUG by entering the Quit command, Q.

Step 2.0

Load BASIC and then, while in BASIC, load the protected program. As soon as it has been loaded, exit BASIC using the SYSTEM command.

Step 3.0

The feature of BASIC that makes this approach possible is that the working copy of the program is stored in memory in the same tokenized form regardless of how it was stored on disk. This is the same form as that found on the disk when a program is saved in the routine fashion, i.e., without either the A or the P option being specified and, therefore, is always LISTable. Thus, the task at hand is simply to locate the working copy of the previously protected program and write it to disk. Once that is done, it is as available as any other. However, BASIC must be able to recognize it as a standard tokenized program. The identification is carried in the first byte; if it is FFh, then the program is in standard tokenized form; if it is FEh, it is in protected form; if neither, it is an ASCII file.

Step 3.1 MBASIC

Call up DDT; do not specify a filename. Then, using the Set command, S, insert the value FF into location 100. This is done by entering "S100" at the DDT prompt. DDT will respond with "100:xx", where "xx" is the current contents of location 100. Simply enter "FF" and a RETURN. The next address will be presented along with its contents; however, we have no interest in this, so exit by entering a period followed by a RETURN. The task is accomplished; the previous value at 100 has been replaced with FFh.

Step 3.2 MBASIC

Use the Move command, M, to transfer the entire tokenized program to location 101. (Not 100 — you just put FF there). As an example, assume that you determined in Step 0.3 that your program is stored starting at address 6207 (that's where mine are stored) and that the length of the protected program is something less than 3930 bytes (from Step 1.1). Add these two numbers to arrive at the last address in memory occupied by your program, 9B37 (remember, these are hex numbers). You might look at this area of memory just to see if you recognize the ending portion of your program to assure yourself that your numbers are correct; again, information given later in this article will assist you in this. Specifically then, for this example, you would enter "M6207,9B37,101" and the entire program will be copied into memory beginning at address 101.

Step 3.3 MBASIC

Exit DDT with a CTRL-C and save your effort by entering, at the system prompt, "SAVE 57 <progname.bas>". (Use a name other than that of the original protected program; otherwise, it will be overwritten.) The 57 specifies the amount of memory to be saved and is given in DECIMAL pages (a page is 256 bytes). The example program was 3930 bytes long. Thus, 39 hex pages or 57 decimal pages is enough to include the entire program (because the first byte is at address 100, 39 pages actually covers all addresses up through 39FF).

This completes the MBASIC procedure and a LISTable copy of the formerly protected program resides on your disk with the file name you specified when it was SAVEd. Load it into BASIC and have a look.

Step 3.1 GWBASIC

Call DEBUG without including a program name. Using the Enter, E, command, place the hex value FF in the byte immediately before the starting byte of your program which you found in Step 0.3. (My GWBASIC stores the working program at CS:DD00; therefore, the FF should be put in the address DCFF.) All you need to do is enter "EDCFF"; DEBUG will respond by repeating the address along with the current contents of that address. Enter your "FF" plus a RETURN and the step is done.

Step 3.2 GWBASIC

Add one to the length of the program to account for the byte added in the preceding step and put this value into the CX register. (You found the length in Step 0.1.) This is done by again using the Register command in the form RCX and then entering the incremented value.

Step 3.3 GWBASIC

The final operation is simply to write this working program to disk using the DEBUG Name and Write commands. To do this, enter N <progname.ext>; where <progname.ext> is the name under which you want this unprotected version saved. Then enter Wxxxx; where xxxx is the beginning address of the material to be saved which is, of course, the address in which you installed the FF. DEBUG will respond "Saving yyyy bytes". This will place on your disk a program that, when loaded into BASIC, can be listed, edited and run.

Tokenized Programs

Throughout the foregoing discussion, the term "tokenized" has been used. This concept, the use of tokens, was relatively thor-

Figure 1-1
Tokens Used By Heath/Zenith MBASIC

```
B = next two bytes comprise an octal number
C = next two bites comprise a hex number

E = next two bytes comprise a line reference
F = next byte is a decimal number

11-1A = numbers Ø-9 in excess 11H (17 dec) notation

1C = next two bytes comprise an integer
1D = next four bytes comprise a single precision number

1F = next eight bytes comprise a double precision number
```

```
3A 8F DB  '(REM)   A5 ======== SWAP   CB ======== SAVE   FF 84 ====== SGN
3A A2 ===== ELSE   A6 ======= ERASE   CC ======= RESET   FF 85 ====== INT
81 ========= END   A7 ======== EDIT   CD =============   FF 86 ====== ABS
82 ========= FOR   A8 ======= ERROR   CE ========== TO   FF 87 ====== SQR
83 ======== NEXT   A9 ====== RESUME   CF ======== THEN   FF 88 ====== RND
84 ======== DATA   AA ====== DELETE   DØ ========= TAB   FF 89 ====== SIN
85 ======= INPUT   AB ======== AUTO   D1 ======== STEP   FF 8A ====== LOG
86 ========= DIM   AC ======= RENUM   D2 ========= USR   FF 8B ====== EXP
87 ======== READ   AD ====== DEFSTR   D3 ========== FN   FF 8C ====== COS
88 ========= LET   AE ====== DEFINT   D4 ========= SPC   FF 8D ====== TAN
89 ======== GOTO   AF ====== DEFSNG   D5 ========= NOT   FF 8E ====== ATN
8A ========= RUN   BØ ====== DEFDBL   D6 ========= ERL   FF 8F ====== FRE
8B ========== IF   B1 ======== LINE   D7 ========= ERR   FF 9Ø ====== INP
8C ===== RESTORE   B2 =============   D8 ===== STRING$   FF 91 ====== POS
8D ======= GOSUB   B3 =============   D9 ======= USING   FF 92 ====== LEN
8E ====== RETURN   B4 ======= WHILE   DA ======= INSTR   FF 93 ===== STR$
8F ========= REM   B5 ======== WEND   DB =============   FF 94 ====== VAL
9Ø ======== STOP   B6 ======== CALL   DC ====== VARPTR   FF 95 ====== ASC
91 ======= PRINT   B7 ======= WRITE   DD ====== INKEY$   FF 96 ===== CHR$
92 ======= CLEAR   B8 ====== COMMON   EF =========== >   FF 97 ===== PEEK
93 ======== LIST   B9 ======= CHAIN   FØ =========== =   FF 98 === SPACE$
94 ========= NEW   BA ====== OPTION   F1 =========== <   FF 99 ===== OCT$
95 ========== ON   BB === RANDOMIZE   F2 =========== +   FF 9A ===== HEX$
96 ======== NULL   BC =============   F3 =========== -   FF 9B ===== LPOS
97 ======== WAIT   BD ====== SYSTEM   F4 =========== *   FF 9C ===== CINT
98 ========= DEF   BE =============   F5 =========== /   FF 9D ===== CSNG
99 ======== POKE   BF ======== OPEN   F6 =========== ^   FF 9E ===== CDBL
9A ======== CONT   CØ ======= FIELD   F7 ========= AND   FF 9F ====== FIX
9B =============   C1 ========= GET   F8 ========== OR   FF AB ====== CVI
9C =============   C2 ========= PUT   F9 ========= XOR   FF AC ====== CVS
9D ========= OUT   C3 ======= CLOSE   FA ========= EQV   FF AD ====== CVD
9E ====== LPRINT   C4 ======== LOAD   FB ========= IMP   FF AE ==========
9F ======= LLIST   C5 ======= MERGE   FC ========= MOD   FF AF ====== EOF
AØ =============   C6 ======= FILES   FD =========== \   FF BØ ====== LOC
A1 ======= WIDTH   C7 ======== NAME   FE =============   FF B1 ====== LOF
A2 =============   C8 ======== KILL   FF 81 ==== LEFT$   FF B2 ===== MKI$
A3 ======== TRON   C9 ======== LSET   FF 82 === RIGHT$   FF B3 ===== MKS$
A4 ======= TROFF   CA ======== RSET   FF 83 ===== MID$   FF B4 ===== MKD$
```

```
'(REM)  3A 8F DB   EDIT ======== A7   LOC ====== FF BØ   RETURN ====== 8E
* =========== F4   ELSE ===== 3A A2   LOF ====== FF B1   RIGHT$ === FF 82
+ =========== F2   END ========= 81   LOG ====== FF 8A   RND ====== FF 88
- =========== F3   EOF ====== FF AF   LPOS ===== FF 9B   RSET ======== CA
```

```
/ =========== F5   EQV ========= FA   LPRINT ====== 9E   RUN ========= 8A
< =========== F1   ERASE ======= A6   LSET ======== C9   SAVE ======== CB
= =========== FØ   ERL ========= D6   MERGE ======= C5   SGN ====== FF 84
> =========== EF   ERR ========= D7   MID$ ===== FF 83   SIN ====== FF 89
ABS ====== FF 86   ERROR ======= A8   MKD$ ===== FF B4   SPACE$ === FF 98
AND ========= F7   EXP ====== FF 8B   MKI$ ===== FF B2   SPC ========= D4
ASC ====== FF 95   FIELD ======= CØ   MKS$ ===== FF B3   SQR ====== FF 87
ATN ====== FF 8E   FILES ======= C6   MOD ========= FC   STEP ======== D1
AUTO ======== AB   FIX ====== FF 9F   NAME ======== C7   STOP ======== 9Ø
CALL ======== B6   FN ========== D3   NEW ========= 94   STR$ ===== FF 93
CDBL ===== FF 9E   FOR ========= 82   NEXT ======== 83   STRING$ ===== D8
CHAIN ======= B9   FRE ====== FF 8F   NOT ========= D5   SWAP ======== A5
CHR$ ===== FF 96   GET ========= C1   NULL ======== 96   SYSTEM ====== BD
CINT ===== FF 9C   GOSUB ======= 8D   OCT$ ===== FF 99   TAB ========= DØ
CLEAR ======= 92   GOTO ======== 89   ON ========== 95   TAN ====== FF 8D
CLOSE ======= C3   HEX$ ===== FF 9A   OPEN ======== BF   THEN ======== CF
COMMON ====== B8   IF ========== 8B   OPTION ====== BA   TO ========== CE
CONT ======== 9A   IMP ========= FB   OR ========== F8   TROFF ======= A4
COS ====== FF 8C   INKEY$ ====== DD   OUT ========= 9D   TRON ======== A3
CSNG ===== FF 9D   INP ====== FF 9Ø   PEEK ===== FF 97   USING ======= D9
CVD ====== FF AD   INPUT ======= 85   POKE ======== 99   USR ========= D2
CVI ====== FF AB   INSTR ======= DA   POS ====== FF 91   VAL ====== FF 94
CVS ====== FF AC   INT ====== FF 85   PRINT ======= 91   VARPTR ====== DC
DATA ======== 84   KILL ======== C8   PUT ========= C2   WAIT ======== 97
DEF ========= 98   LEFT$ ==== FF 81   RANDOMIZE === BB   WEND ======== B5
DEFDBL ====== BØ   LEN ====== FF 92   READ ======== 87   WHILE ======= B4
DEFINT ====== AE   LET ========= 88   REM ========= 8F   WIDTH ======= A1
DEFSNG ====== AF   LINE ======== B1   RENUM ======= AC   WRITE ======= B7
DEFSTR ====== AD   LIST ======== 93   RESET ======= CC   XOR ========= F9
DELETE ====== AA   LLIST ======= 9F   RESTORE ===== 8C   \ =========== FD
DIM ========= 86   LOAD ======== C4   RESUME ====== A9   ^ =========== F6
```

Figure 2-1
Tokens Used By Heath/Zenith GWBASIC

```
B = next two bytes comprise an octal number
C = next two bites comprise a hex number

E = next two bytes comprise a line reference
F = next byte is a decimal number

11-1A = numbers Ø-9 in excess 11H (17 dec) notation

1C = next two bytes comprise an integer
1D = next four bytes comprise a single precision number

1F = next eight bytes comprise a double precision number
```

```
3A 8F D9  '(REM)   AE ====== DEFSNG   DE ====== INKEY$   FE 9A ==== SHELL
3A A2 ===== ELSE   AF ====== DEFDBL   E6 =========== >   FE 9B == ENVIRON
81 ========= END   BØ ======== LINE   E7 =========== =   FE 9C ===== VIEW
82 ========= FOR   B1 ======= WHILE   E8 =========== <   FE 9D === WINDOW
83 ======== NEXT   B2 ======== WEND   E9 =========== +   FE 9E ===== PMAP
84 ======== DATA   B3 ======== CALL   EA =========== -   FE 9F == PALETTE
85 ======= INPUT   B4 =============   EB =========== *   FE AØ ==== LCOPY
86 ========= DIM   B5 =============   EC =========== /   FE A1 ==== CALLS
87 ======== READ   B6 =============   ED =========== ^   FF 81 ==== LEFT$
88 ========= LET   B7 ======= WRITE   EE ========= AND   FF 82 === RIGHT$
89 ======== GOTO   B8 ====== OPTION   EF ========== OR   FF 83 ===== MID$
```

oughly outlined in an article "Inside Microsoft BASIC (MBASIC)" appearing in REMark 5:8 13, August, 1984. A list of decimal tokens was included, but unfortunately, a few were omitted. Figure 1 lists all of the tokens used by MBASIC 5.21 while Figure 2 does the same for GWBASIC 2.11. The lists are presented in hexadecimal for two reasons: first, that is the way the debugger presents them, and second, if BASIC is used to examine an area of memory, the HEX$() function is available to automatically convert the returned decimal values to hex.

The use of tokens, i.e., using one thing to represent another, is a fairly common device (maybe even universal) in high level computer languages. Rather than require that the interpreter or compiler recognize a series of ASCII characters, say the six characters in LPRINT, it is much easier for it to recognize a single byte, in this case, 9E, which is the token for LPRINT. Furthermore, there is an obvious and significant saving in disk and memory space. While there may be reasons for assigning a specific token to a given function or command, as a general principle, it can be an entirely arbitrary relation. Most MBASIC and GWBASIC tokens are only one byte long, though some are longer. Tokens assigned for use by different interpreters need not be identical, but of course, programs using one set will not run on an interpreter using another.

It is a very simple procedure to determine the tokens used by your particular interpreter. (A relatively complicated, but more machine oriented, method was given in REMark 6:6 22, June, 1985). To determine the token for a given keyword, simply write a one line program containing only a line number and the keyword; e.g., 100 LPRINT. Of course, it makes more sense to write a multiline program using as many keywords as you choose; this avoids much work but requires a little more care in reading the

```
8A ========= RUN
8B ========== IF
8C ===== RESTORE
8D ======= GOSUB
8E ====== RETURN
8F ========= REM
9Ø ======== STOP
91 ======= PRINT
92 ======= CLEAR
93 ======== LIST
94 ========= NEW
95 ========== ON
96 ======== WAIT
97 ========= DEF
98 ======== POKE
99 ======== CONT
9A =============
9B =============
9C ========= OUT
9D ====== LPRINT
9E ======= LLIST
9F =============
AØ ======= WIDTH
A2 ======== TRON
A3 ======= TROFF
A4 ======== SWAP
A5 ======= ERASE
A6 ======== EDIT
A7 ======= ERROR
A8 ====== RESUME
A9 ====== DELETE
AA ======== AUTO
AB ======= RENUM
AC ====== DEFSTR
AD ====== DEFINT
```

```
B9 === RANDOMIZE
BA ======== OPEN
BB ======= CLOSE
BC ======== LOAD
BD ======= MERGE
BE ======== SAVE
BF ======= COLOR
CØ ========= CLS
C1 ======= MOTOR
C2 ======= BSAVE
C3 ======= BLOAD
C4 ======= SOUND
C5 ======== BEEP
C6 ======== PSET
C7 ====== PRESET
C8 ====== SCREEN
C9 ========= KEY
CA ====== LOCATE
CB =============
CC ========== TØ
CD ======== THEN
CE ======== TAB(
CF ======== STEP
DØ ========= USR
D1 ========== FN
D2 ======== SPC(
D3 ========= NOT
D4 ========= ERL
D5 ========= ERR
D7 ======= USING
D8 ======= INSTR
DA ====== VARPTR
DB ====== CSRLIN
DC ====== POINTR
DD ========= OFF
```

```
FØ ========= XOR
F1 ========= EQV
F2 ========= IMP
F3 ========= MOD
F4 =========== \
FD 81 ====== CVI
FD 82 ====== CVS
FD 83 ====== CVD
FD 84 ===== MKI$
FD 85 ===== MKS$
FE 81 ==== FILES
FE 82 === FIELDS
FE 83 === SYSTEM
FE 84 ===== NAME
FE 85 ===== LSET
FE 86 ===== RSET
FE 87 ===== KILL
FE 88 ====== PUT
FE 89 ====== GET
FE 8A ==== RESET
FE 8B === COMMON
FE 8C ==== CHAIN
FE 8D ==== DATE$
FE 8E ==== TIME$
FE 8F ==== PAINT
FE 9Ø ====== COM
FE 91 === CIRCLE
FE 92 ===== DRAW
FE 93 ===== PLAY
FE 94 ==== TIMER
FE 95 ==== ERDEV
FE 96 ==== IOCTL
FE 97 ==== CHDIR
FE 98 ==== MKDIR
FE 99 ==== RMDIR
```

```
FF 84 ====== SGN
FF 85 ====== INT
FF 86 ====== ABS
FF 87 ====== SQR
FF 88 ====== RND
FF 89 ====== SIN
FF 8A ====== LOG
FF 8B ====== EXP
FF 8C ====== COS
FF 8D ====== TAN
FF 8E ====== ATN
FF 8F ====== FRE
FF 9Ø ====== INP
FF 91 ====== POS
FF 92 ====== LEN
FF 93 ===== STR$
FF 94 ====== VAL
FF 95 ====== ASC
FF 96 ===== CHR$
FF 97 ===== PEEK
FF 98 === SPACE$
FF 99 ===== OCT$
FF 9A ===== HEX$
FF 9B ===== LPOS
FF 9C ===== CINT
FF 9D ===== CSNG
FF 9E ===== CDBL
FF 9F ====== FIX
FF AØ ====== PEN
FF A1 ==== STICK
FF A2 ==== STRIG
FF A3 ====== EOF
FF A4 ====== LOC
FF A5 ====== LOF
```

```
'(REM)  3A 8F D9
* =========== EB
+ =========== E9
- =========== EA
/ =========== EC
< =========== E8
= =========== E7
> =========== E6
ABS ====== FF 86
AND ========= EE
ASC ====== FF 95
ATN ====== FF 8E
AUTO ======== AA
BEEP ======== C5
BLOAD ======= C3
BSAVE ======= C2
CALL ======== B3
CALLS ==== FE A1
CDBL ===== FF 9E
CHAIN ==== FE 8C
```

```
DELETE ====== A9
DIM ========= 86
DRAW ===== FE 92
EDIT ======== A6
ELSE ===== 3A A2
END ========= 81
ENVIRON == FE 9B
EOF ====== FF A3
EQV ========= F1
ERASE ======= A5
ERDEV ==== FE 95
ERL ========= D4
ERR ========= D5
ERROR ======= A7
EXP ====== FF 8B
FIELD  === FE 82
FILES ==== FE 81
FIX ====== FF 9F
FN ========== D1
FOR ========= 82
```

```
LOCATE ====== CA
LOF ====== FF A5
LOG ====== FF 8A
LPOS ===== FF 9B
LPRINT ====== 9D
LSET ===== FE 85
MERGE ======= BD
MID$ ===== FF 83
MKDIR ==== FE 98
MKI$ ===== FD 84
MKS$ ===== FD 85
MOD ========= F3
MOTOR ======= C1
NAME ===== FE 84
NEW ========= 94
NEXT ======== 83
NOT ========= D3
OCT$ ===== FF 99
OFF ========= DD
ON ========== 95
```

```
RETURN ====== 8E
RIGHT$ === FF 82
RMDIR ==== FE 99
RND ====== FF 88
RSET ===== FE 86
RUN ========= 8A
SAVE ======== BE
SCREEN ====== C8
SGN ====== FF 84
SHELL ==== FE 9A
SIN ====== FF 89
SOUND ======= C4
SPACE$ === FF 98
SPC( ======== D2
SQR ====== FF 87
STEP ======== CF
STICK ==== FF A1
STOP ======== 9Ø
STR$ ===== FF 93
STRIG ==== FF A2
```

```
CHDIR ==== FE 97
CHR$ ===== FF 96
CINT ===== FF 9C
CIRCLE === FE 91
CLEAR ======= 92
CLOSE ======= BB
CLS ========= CØ
COLOR ======= BF
COM ====== FE 9Ø
COMMON === FE 8B
CONT ======== 99
COS ====== FF 8C
CSNG ===== FF 9D
CSRLIN ====== DB
CVD ====== FD 83
CVI ====== FD 81
CVS ====== FD 82
DATA ======== 84
DATE$ ==== FE 8D
DEF ========= 97
DEFDBL ====== AF
DEFINT ====== AD
DEFSNG ====== AE
DEFSTR ====== AC
```

```
FRE ====== FF 8F
GET ====== FE 89
GOSUB ======= 8D
GOTO ======== 89
HEX$ ===== FF 9A
IF ========== 8B
IMP ========= F2
INKEY$ ====== DE
INP ====== FF 9Ø
INPUT ======= 85
INSTR ======= D8
INT ====== FF 85
IOCTL ==== FE 96
KEY ========= C9
KILL ===== FE 87
LCOPY ==== FE AØ
LEFT$ ==== FF 81
LEN ====== FF 92
LET ========= 88
LINE ======== BØ
LIST ======== 93
LLIST ======= 9E
LOAD ======== BC
LOC ====== FF A4
```

```
OPEN ======== BA
OPTION ====== B8
OR ========== EF
OUT ========= 9C
PAINT ==== FE 8F
PALETTE == FE 9F
PEEK ===== FF 97
PEN ====== FF AØ
PLAY ===== FE 93
PMAP ===== FE 9E
POINTR ====== DC
POKE ======== 98
POS ====== FF 91
PRESET ====== C7
PRINT ======= 91
PSET ======== C6
PUT ====== FE 88
RANDOMIZE === B9
READ ======== 87
REM ========= 8F
RENUM ======= AB
RESET ==== FE 8A
RESTORE ===== 8C
RESUME ====== A8
```

```
SWAP ======== A4
SYSTEM === FE 83
TØ ========== CC
TAB( ======== CE
TAN ====== FF 8D
THEN ======== CD
TIME$ ==== FE 8E
TIMER ==== FE 94
TROFF ======= A3
TRON ======== A2
USING ======= D7
USR ========= DØ
VAL ====== FF 94
VARPTR ====== DA
VIEW ===== FE 9C
WAIT ======== 96
WEND ======== B2
WHILE ======= B1
WIDTH ======= AØ
WINDOW === FE 9D
WRITE ======= B7
XOR ========= FØ
\ =========== F4
^ =========== ED
```

results. Whether single line or multiline, after the program is written, exit BASIC; call up the debugger; dump the program beginning at the location you found in Step 0.0 above and there you will find the tokenized form of the program. Using the example just given, you will find first a two byte pointer to the next program line, then a two byte line number (in this case, 64 00; 64 being the hex equivalent of 100), then the token for LPRINT, 9E (remember, this is all in hex), and finally, a line terminator, 00. This will repeat for each line of your program — every 7 bytes if none of the tokens happens to be one that is two bytes long.

The converse of this operation, finding a key word for a given token, is a bit more complicated but still quite easy to do. This was the approach used to produce the data given in the figures. Write a 128 line BASIC program with each line consisting only of the line number and a keyword that is known to have a single byte token. (All tokens have a value of 129d (81h) or greater since the ASCII characters occupy all lesser values.) A simplification is to start your line numbers at 129 and increment each by one; the line number will then correspond to the token number. (For GWBASIC, an easy way to do this is to use AUTO 129,1 and the F9 key which enters KEY). Save this program. Call your debugger and load the just saved program. Then using either the (S)et command for DDT or the (E)nter command for DEBUG, change each occurrence of C9 (if KEY was the keyword you used) to a hex number starting with 81 and continuing through FF. (Caution: don't change the one C9 that is a line number.) A little tedious but easy. Note the length of your program; it should be 128*6 bytes or 3 pages. Save (or Write) this section of memory to disk. Now call BASIC; load the program you just saved and list it. You will now have a listing of all the one byte tokens with the line numbers corresponding to the token.

The same procedure can be used for the two byte tokens, but it is considerably more wasteful of time and effort since much of the 128 byte total capacity is not used but all of it must be investigated. Furthermore, while MBASIC uses only a single series of two byte tokens, those prefaced with FF, GWBASIC also has an FD and an FE series though there are few used in the latter two.

With the information given in the token lists, you should be able to read your tokenized programs and even make changes. The only thing that hasn't been discussed, either here or in the cited REMark articles, is the interpretation of numeric data. While the representation of integers is straight forward, the conversion from the format used for single and double precision numbers to a human readable form is considerably less than obvious.

Both of these are stored as floating point numbers: the single precision in four bytes, the double precision in eight. Both require exactly the same technique, but only single precision conversion will be demonstrated because of the length of the other. The token 1D precedes the four bytes of single precision data while 1F serves as the identifier for double precision.

The first two bytes after the 1D token are the least significant eight bits and the middle eight bits, respectively, of a 23 bit number. The third byte specifies the most significant seven bits. For demonstration purposes, assume that your program contains the following: 1D 78 E9 76 87. Arranging the three bytes in order, gives a single hex number 76E978. As a convention, there is assumed to be a 1 in the most significant bit of the most significant byte. If it is omitted, then 1 should be appended thereby making the number F6E978. If it is included, make no change but recognize that the number is negative. A further convention is that the binary point (hex point in this case) is assumed to be at the extreme left.

Converting the number to a decimal fraction produces the value 0.96449995.

The fourth byte of the original series is 87. This byte contains the position of the binary point in excess 80 code. Converting both numbers to decimal yields 135 and 128 which gives a difference of +7. This indicates that had the number been expressed in binary, the binary point should be shifted right by 7 positions. (If the number had been negative, the point would have been shifted left.) Shifting a binary point right one position, is equivalent to multiplying the number by two. Shifting it right by seven is the same as multiplying by two a total of seven times or 128. Therefore, multiplying our fraction by 128 yields 123.456 when rounded to the six digit maximum of single precision and is the number that would appear in the listing of a program.

Compared to changing a whole hex or binary number to decimal, converting a fraction is, to say the least, laborious. However, the conversion of whole numbers is relatively easy and many modern hand calculators will do it for you. For this reason, there follows a process for changing the tokenized number to decimal that requires working only with whole numbers.

Converting the original hex number, F6E978, to a whole decimal yields 16181624.

Since the original binary number consists of 24 bits (3 bytes), a shift of the point 7 places to the right from the left is identical to a shift of 17 places (24–7) leftward from the right. A shift left of one place is equivalent to a division by two so that a shift of 17 bits is the same as division by 2^17 or 131072 giving an identical result, 123.456. Using this simpler, whole number technique, the fourth byte of the original group of four should be considered as the binary point expressed in terms of excess 98 rather than excess 80. In this example, 87–98 = –17 places left.

If the number in the program is of double precision, the same techniques should be applied. And when the binary point is assumed to be to the right, the point position information should be considered as being in excess B8h or 184d.

One final point of interest. In MBASIC, the two byte pointer starting each line of the tokenized program points to the actual address of the pointer for the next line. This is simply a linked list and is terminated when a pointer of 00 00 is encountered. GWBASIC also uses pointers but they are apparently the offset from some other specified address and do not point directly as do those in MBASIC.

HUG Engineer's Note: For a much easier way than presented here to unprotect GW-BASIC programs, see the article "Recovering From the Protected Mode in ZBASIC or GW-BASIC" in the August 1985 issue of REMark.

-- SHIPPING NOW -- SHIPPING NOW -- SHIPPING NOW --

WILDFIRE

An INCREDIBLE PERFORMANCE IMPROVEMENT from Software Wizardry

SPEED... WILDFIRE has spirit. It will spur your Z-151 or Z-161* to run faster than the 8mHz Z-158! And WILDFIRE isn't just a simple crystal change like some products; it gives you more speed and much greater compatibility.

COMPATIBILITY...WILDFIRE uses the NEC V-20 chip plus lots of other enhancements to make an unbeatable combination of performance and reliability. It outdoes the 80286 processor chip additions for compatibility with the IBM PC and costs up to $600.00 less!

FEATURES... the heart of WILDFIRE is a daughter board that mounts on your processor board, saddled comfortably between it and the video board. Several higher-speed chips are included to replace socketed chips on your processor board. A high speed/low speed switch that mounts on your machine's front panel keeps WILDFIRE on a tight rein, under your control to trot at normal speed, or break into a real gallop when you really want to ride!

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

••• 15-DAY MONEY-BACK GUARANTEE! •••

If you order now, you can try WILDFIRE with a 15-day money-back guarantee! If you don't like it, you can return it within 15 days and get a full refund!

WILDFIRE...the ultimate solution to your speed problems. And with our special low price, AND our 15-day money-back guarantee, you can't go wrong! Order now!

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

Software Wizardry

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

A Winchester For The '89

Part Ten

Peter Ruber
P.O. Box 502
Oakdale, NY 11769

SigmaSoft Strikes Again [II]

With all Winchester hardware being relatively equal in terms of quality and reliability, the versatility and ease in setting up a system rests solely in the hands of the software developer. The SigmaSoft Disk System installation software is a series of easy to follow menu screens that guide you through the hard disk formatting and partitioning stages.

If you're accustomed to working with Winchester systems, the basic set-up procedures are quite similar no matter whose system you are working with. The amount of information you have to supply to the system software depends pretty much on the data needed by the hard disk controller and the software. Some systems require that you provide the Write Precompensation Cylinder Number, as well as the Write Reduction Cylinder Number for the hard disk drive you are using. This information is recorded on the system tracks for future reference by the controller. Sometimes the Write Reduction Cylinder number is ignored, and any number you supply is simply recorded as a Zero.

Other systems require that you insert the name of the controller and the name of the drive, in addition to the standard data on the number of cylinders available, interleave factor, step-rate, landing (or parking) zone, etc.

But whatever is required by the system software, there are 4 basic steps:

1. Hard disk formatting, which includes the entry of the data mentioned above.
2. Hard disk partitioning, which breaks up the available work space into manageable chunks.
3. Initializing (or formatting) the partition tracks the same way as you would a floppy.
4. Sysgening the partitions with the disk operating system so that you create bootable partitions.

The SigmaSoft Disk System is compatible with both Heath/Zenith CP/M and HDOS, and software for both of these operating systems is supplied. HDFORMAT.COM (for CP/M) and HDFORMAT .ABS (for HDOS) are the hard disk formatting utilities. It doesn't matter which one you use because they perform the same task.

Getting Started

When you invoke HDFORMAT, you enter the data for the drive you received with your system from the chart in the manual. The initial formatting procedure takes about five minutes for a 20-MB drive. The second portion of the utility takes about 12 minutes, during which a read/write is performed on every cylinder in order to check for bad sectors.

Generally, you will get a message saying NO BAD SECTORS FOUND. But, you may receive a list of a few bad sectors, which is nothing to be concerned about. In order to bring the cost of hard disk drives down to the level of what you paid for a floppy just a few years ago, manufacturers occasionally release drives with minor flaws. If HDFORMAT encounters any bad sectors, they will record the numbers on Sectors 11 and 12 on the drive's Track 0 so that the system will not access these when you read and write data to the drive.

Even if your drive was flawless when you formatted it for the first time, you may sometime in the future encounter bad sectors in a particular partition. SigmaSoft has included a Public Domain program called FINDBAD.COM that can be used to test a partition if you start receiving read/write errors. It will map out the newly-found bad sectors and write them on the system's tracks.

The next step in using the SigmaSoft Disk System is to run the HDPART program. Again, there are like versions for CP/M and HDOS, and either version can be used. This program is probably the most practical I have used to date.

After entering the data for the Hard Disk Base Port and the Hard Disk Drive Number, you enter a spreadsheet type of menu that lists several categories —

TYPE — (CP/M or HDOS partition)
NAME — (An Alpha code name you can assign to each partition)
FIRST CYLINDER — (The number of the next available cylinder number)
CYLINDER COUNT — (Number of free cylinders remaining on the hard disk)
PARTITION SIZE — (Kilobyte space remaining on disk)
DEFAULT BOOT — (Shows which partition will be your CP/M and HDOS primary boot partition)

At the bottom of the screen you have a choice to —

A — Add a Partition
D — Delete a Partition
B — Set Boot Default
X — Exit This Utility

And, you are informed that the cylinder size is 32k byte.

Now, before you jump ahead to create a partition, you should give special consideration to how you want to assign the partitions for the most productive purposes. SigmaSoft's documentation recommends that you establish your Default Boot Partitions for CP/M and HDOS at the beginning of the drive for faster access. That aspect of your work most frequently used should become your Default Boot Partition.

In order to illustrate this for you, I'll explain what I did during the initial partitioning of the drive.

I typed letter "A" to Add a Partition. Then I selected the type of partition (CP/M or HDOS) by typing 1 or 2, and the number of cylinders that I wanted to assign to the partition. I chose HDOS and a partition size of 40 cylinders. This would give me 1,280,000 bytes of work space with a small cluster size. The cluster size, as you'll recall in my discussion of the QUIKSTOR software from Quikdata (Part 3), is the minimum file space consumed. In a 1.0 to 1.5-MB partition your cluster size will be about 2k. From 2.0 to 4.0-MB, the minimum cluster size becomes 4k. And at 8.0-MB, that jumps to 8k. Thus, the larger your partition size, the more actual unused space you wind up wasting.

The program then asks you to enter a NAME for the partition. I selected LETTERS. When I hit the return key, the menu came up again showing the data I had just entered and the remaining number of cylinders and kilobytes remaining on the disk.

Then I set up a second HDOS partition of 40 cylinders called ARTICLES. A third called INVOICES, and a fourth called NOTES.

Under CP/M I created a partition called ARTICLES and one called NOTES. The program allows you to create duplicate names for each of the operating systems. I wasn't certain what I wanted to do with the rest of the space, so I left it blank for the time being. The HDPART program allows you complete the partitioning at any future date or to delete a specified partition at any time, and even to change the default boot partition.

I set the BOOT partition for each operating system from the menu and exited the program.

The next step with the SigmaSoft Disk System (which isn't generic to all hard disk installation software) is to run the ASSIGN utility. This assigns the partition names to each respective drive unit: A:, B:, or SY1:, SY2:, etc.

You invoke ASSIGN directly from the system prompt in the same way that you would use the SET option. This is done as follows:

```
>ASSIGN HDØ:=LETTERS, HD1:=ARTICLES, HD2:=NOTES, etc.
```

Since I started off with HDOS, I performed an INIT and a SYSGEN on each partition in the same way as I would a floppy disk. When INIT asked for a system label, I assigned the same name as I had for the partitions. If you jam the hard disk with the maximum of 16 allowable partitions and you forget what is where, simply run HDPART for the menu screen, or ">ASSIGN?"

My final step was to PIP my Text Editors and Word Processors to each partition. When this was accomplished, I reset the computer and booted from the hard disk.

Back Tracking

As I pointed out in the first part of this article, the SigmaSoft Disk System is a combination hard disk and Z-37 compatible floppy controller all on one board. Since a single IC (the 1015) controls access to and from both types of disks (via their respective controller chips) through a sector buffer, the SigmaSoft Disk System shares a single I/O Port (0).

As such, it does not conflict with the two available Disk I/O Ports (170Q and 174Q) that Heath/Zenith allows on the '89. If you presently own one or more floppy interfaces (H-17, Z-37 or Z-47) installed in your system, they will be fully functional as before. However, only 2 devices can be selected as PRIMARY and SECONDARY Boot devices through Switch S501 on the CPU board. The remaining devices can be accessed by the normal device selection manner peculiar to each operating system. Your Boot devices can, of course, be changed by simply rearranging S501. Also, the normal H/Z conventions of booting from different drives within a specified device has been preserved.

Turning To CP/M

The running of the ASSIGN program under HDOS applies as well to CP/M. Since I performed the initial formatting and partitioning under HDOS, I now had to boot up a CP/M disk containing the SigmaSoft support software. Assigning partitions is performed through the ASSIGN program:

```
A>ASSIGN A:=ARTICLES, B:=NOTES, etc.
```

SigmaSoft has allowed the following labels to each possible device on a CP/M system:

WD1ØØ2 Hard Disk	A:, B:, C:, D:
H-17, Z-37 or Z-47	E:, F:, G:
H-17, Z-37 or Z-47	H:, I:, J:
WD1ØØ2 Floppy	K:, L:, M:, N:

While 8 HDOS partitions can be created through the HD.DVD device driver, only 4 CP/M partitions are available. If you actually require 8 CP/M partitions (for a total of 16 partitions on the drive),

you can assign any unused floppy device designation as a hard disk partition via ASSIGN:

```
    A>ASSIGN K:=PARTNAME, L:=PARTNAME, M:=PARTNAME, etc.
```

In this fashion, while you could assign up to 16 CP/M, using any valid drive select letters from A: through P:, only 4 CP/M partitions can be assigned as bootable partitions at any given time. When you run the ASSIGN program to designate new CP/M partitions, floppy devices will automatically be reassigned to another drive letter if conflict occurs.

There are two methods of accessing a CP/M partition on the SigmaSoft hard disk. If you don't want to bother installing the appropriate SigmaBIOS version that corresponds with the type of disk system you have on line (which we will get to shortly), you can use the LOADD.COM utility.

The utility allocates space from the Transient Program Area (TPA) of the CP/M system and reserves it for the installed driver module. It's primary advantage is that it is simple to use and acts just like an HDOS device driver. The disadvantage is that you must initiate LOADD.COM from a floppy drive, and it remains active only until the next cold boot. A CTRL-C warm boot doesn't dismount the driver from the system. It also doesn't allow you to boot directly from the hard disk.

The LOADD utility can load multiple drivers, such as the HD.DVD (hard disk driver), WDC.DVD (the Western Digital floppy controller), the UPC.DVD (the parallel driver). If you're using the SigmaSoft Interractive Graphics Controller, you can load the Graphics Driver, Ramdisk Driver or Print-Spooler driver at the same time. In specialized applications, this is a simple way to go. But, as I said earlier, these drivers only remain active until you cold boot again.

A practical use, however, for the LOADD utility is that it helps to overcome any BIOS incompatibilities that are hardware dependent — such as the CDR 5/8 floppy controller, the Magnolia version of CP/M, etc. Since these BIOSes cannot be incorporated into a standard H/Z BIOS, you are likely to encounter occasional software problems. LOADD will allocate space from any incompatible BIOS for a device driver, so that specific hardware and software will function correctly with the SigmaSoft Disk System. SigmaSoft first used this method with their IGC in order to allow this graphics board to function with almost any proprietary Heath/Zenith compatible CP/M BIOS.

To boot from the hard disk, you will have to incorporate the SigmaBIOS into the CP/M operating system. You have 7 versions to choose from —

SBIOS1.SYS	WD1ØØ2 only
SBIOS2.SYS	WD1ØØ2 and H-17
SBIOS3.SYS	WD1ØØ2 and Z-37
SBIOS4.SYS	WD1ØØ2 and Z-47
SBIOS5.SYS	WD1ØØ2 and H-17 + Z-37
SBIOS6.SYS	WD1ØØ2 and Z-37 + Z-47
SBIOS7.SYS	WD1ØØ2 and Z-47 + H-17

Create a bootable CP/M disk and place the appropriate SBIOS? .SYS file on the disk. Make certain that the disk also contains the basic CP/M programs: SYSGEN.COM, MOVCPM??.COM, STAT .COM, PIP.COM.

Type:

```
A>ASSIGN ?:=PARTNAME (the "?" represents the letter of
                      the drive partition we will SYSGEN)
```

Type:

```
MOVCPM * SBIOS?.SYS (the SigmaBIOS provides a 64K system
                     as indicated by the "*" to SYSGEN)
```

When completed, type SYSGEN. When SYSGEN asks you for a source drive name, hit RETURN because the system has already been placed into memory by MOVCPM. When you are asked for the destination, type the designated letter for your drive (which was created above by ASSIGN. After the FUNCTION COMPLETED message appears, type an extra RETURN to exit from the SYSGEN utility.

The next step is to rename the newly created BIOS:

```
A>PIP "?":BIOS.SYS=SBIOS?.SYS[0]
```

This completes the installation of the SigmaBIOS on your CP/M partition.

If you are working with a non-Heath/Zenith CP/M system, you have to create a system size that allows for sufficient space above the CCP (Console Command Processor) to hold all the device drivers. The Bytes these drivers consume will reduce your system size. Each Kilobyte of driver overhead will cause a corresponding decrease in your system size.

The MOVCPM program will place the system size into memory:

```
A>MOVCPM 6Ø (this size is for illustrative purposes only)
```

Then use SYSGEN to place it from memory to disk.

Now you can SYSGEN your CP/M partitions and PIP your designated programs to the hard disk. Once you've installed the SigmaBIOS to your non-Heath/Zenith CP/M system and then placed it on your hard disk partition, the SigmaBIOS provides a full 64K system.

Plan Your System

As you've probably concluded by now, there is a certain amount of work involved in setting up your SigmaSoft Disk System for the first time. It is fair to assume that you will probably spend the better part of an evening doing it right. There isn't much point in rushing through all the steps, because you might overlook something and then wind up spending twice as much time redoing the work.

The more sophisticated PC-DOS (MS-DOS) already contains all the utilities for setting up a hard disk from start to finish in about 20 minutes. And while CP/M and HDOS may seem primitive by comparison, MS-DOS has some irritating limitations when it comes to setting up hard disks.

Most versions of MS-DOS distributed by computer manufacturers, however, exclude the FDISK utility that prepares the hard disk for formatting. Generally, the OEMs performs this task, because they want to sell you a controller and drive combination. Early versions of MS-DOS limit you to using 10-20 MB hard disk drives. Versions 3.0 or higher allow you to use up to 40 MB drives. But it is difficult to break up this space into manageable partitions unless you can find a special partitioning program.

All of the CP/M and HDOS hard disk software available for the systems I have discussed allow you to create partition sizes to suit your working requirements. In order to isolate yourself from a massive directory listing on an MS-DOS hard disk, you have to resort to creating numerous subdirectories for each program category so that you maintain some semblance of order. Not only is is easier to work with smaller partitions, but you will avoid the confusion caused by a subdirectory "tree" structure. In this respect, using hard disks with CP/M and HDOS has very practical advantages.

What isn't evident at first when you read the SigmaSoft documentation is that you should sit down and really plan out the physical mapping of the hard disk drive. If your system is used for even small business purposes, you are probably better off getting the larger capacity drive. The cost difference between a 10-MB and 20-MB drive is only about $100, which brings your cost per kilobyte of storage way down. The SigmaSoft Disk System offers enhancements in the partitioning program that aren't available with any other system on the market.

Other Utilities

A number of Public Domain programs and SigmaSoft utilities are supplied with the HDOS and CP/M support software. HDPARK is generic to both operating systems. When called up it will park the hard disk drive heads to an unformatted track area where they will be secure. This is recommended if you ever have the need to ship the drive or move it. I use HDPARK and its counterparts on all the hard disk drives I have because I am constantly shifting my equipment and I feel more confident if I know the heads are locked in place.

You must remember that if you park the read/write heads, and then change your mind, you must first reset your computer before you boot again. Trying to power up the hard disk through a warm boot or by trying to access a program can cause problems.

SigmaSoft has also provided generic HDUMP utilities for backing up hard disk files, and a CONFIG.COM utility for CP/M that works the same as standard CONFIGUR.COM, but allows you to establish all possible parameters for all the devices supported by the SigmaSoft Disk System.

The MAKESBC.COM utility will allow you to recreate the Software Boot Code portion of the format table on cylinder 0 of an installed hard disk. This will allow you to install an updated or revised version of the support software, repair certain types of partition damage, and enable you to move partition table information to another drive without the need to use HDFORMAT for this purpose.

Under HDOS, the WDC.DVD floppy device driver works the same way as the standard SY.DVD and DK.DVD do. It will support up to four 40- and/or 80-track DS/DD drives in a soft-sectored format. Your Z-37 and H-17 (or H-47) floppy controller cards can coreside with the WD-1002 controller. You do, of course, need to be mindful of the total number of device drivers placed on your system disk to allow room for printer device drivers and others which may be needed by plug-in cards in your '89.

It is, of course, impossible to describe all the software features in clinical detail. I have tried merely to illustrate how the basic set-up is performed so that you have a reference in comparing the SigmaSoft Disk System to the systems discussed in previous articles. SigmaSoft provides a series of examples, for instance, on how to use the various HDOS and CP/M device drivers for accessing the combined Western Digital hard disk and floppy disk system; setting the printer parameters for the Universal Parallel Interface. All these require some study and practice so that you are comfortable with them.

Summing Up

This has been a fun system to work with because there are so many options. And even though I have become accustomed to it, I still hold to my earlier comments that SigmaSoft should consider preparing a check-list type of installation procedure, as well as a step-by-step procedure for installing the software. This can be accomplished by providing a couple of supplement sheets. Details of how a software utility functions can be learned from the documentation itself, but the order of how the utilities are to be used is important, especially since they are scattered throughout the manual, and can be confusing to users exposed to a hard disk system for the first time.

Aside from this criticism, the SigmaSoft Disk System is intelligently conceived and a high caliber product. I encountered no problems using a variety of enhancement boards from Heath/Zenith and some independent manufacturers.

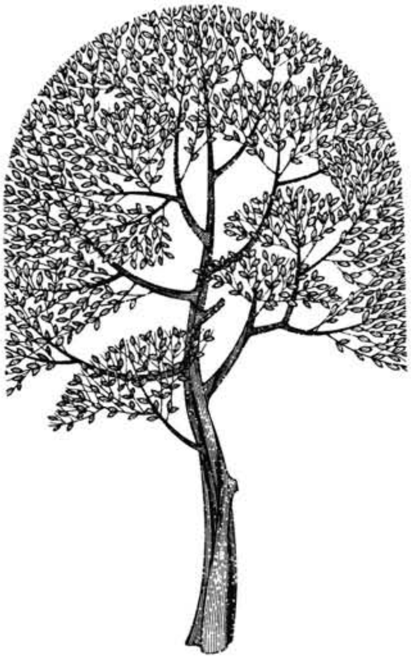
The Users Manual also includes a complete parts list and pin-out specs for the Universal Parallel Interface card which provides two separate Centronics compatible parallel ports. The product is warranted for 6 months.

* * *

Current pricing for 10- and 20-MB internal hard disk systems is $895 and $995. External systems begin at $995.

For additional information and an informative booklet explaining the SigmaSoft Disk System, please contact —

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



Roots II

A Dream Program For Genealogists

Tom Huber
Senior Technical Writer
Zenith Data Systems

Part 3

Roots II is the latest of a series of genealogical programs from CommSoft. It is designed for use on MSDOS-based computers, including some of the older computer designs that have been converted to use the PC (8088 processor) design. In short, it is the cadillac of all genealogy programs. In the previous two issues, I briefly described the program, provided an explanation about genealogy, talked about installing *Roots II* on the Z-100 and PC-compatible computers, and described some of its features. This month, I conclude the discussion on the features of *Roots II* and summarize my findings.

Roots II Features — Continued

PRINT — This is the report generator for Roots and is very powerful. Because six different reports can be generated by this function, I will cover each separately.

All subfunctions in the PRINT function will generate an index or add to the current index if it already exists. In addition, you may route the output to a disk file for processing by a word processor or printing later. However, the disk file will not contain any special printer commands that you may have specified when you installed Roots. And, this function can quickly fill a disk. For this reason, if you plan on producing a book by using a word processor on the output of Roots, I highly recommend you have a high-capacity disk available of at least 10 megabytes.

Before starting the print subfunction, you may specify a starting page number for that output. This allows you to produce your output in sections, although Roots will track the last page number used if the index function is on.

Finally, any individual who is to be the subject of the printout must appear in the f-table. This function will not allow you to transfer subjects into the f-table; you would use one of the other functions in Roots to do that.

PRINT PEDIGREE CHART — This is a complete printout of an individual's pedigree chart, which can be used as the basis for a published genealogy or for doing further research. My preference is to use this printout as a research tool and reserve the descendant chart for published genealogies.

Not all the vital statistics are recorded on this printout. Besides the person's names and his or her parents, birth, marriage, and death dates are recorded. Places for these events only print the second of the two placename fields. Because of this, I use the second field to record the county and state and the first field to record the town or township.

Normally, five generations are shown on a page. However, 132 print columns are needed for this. If you have a printer that is limited to 80 columns, only four generations will be put on a page.

PRINT AHNENTAFEL CHART — An ahnentafel chart is very similar to a pedigree chart, except that it can contain more information and is formatted differently. The word is German and it means "ancestor table." A unique number is assigned to each individual in the chart. The way the numbers are assigned and their relationships follow a set of rules that are spelled out in the Roots manual. The end result of the ahnentafel chart is that they tell you the exact relationship. For example, if I were the subject of the chart, my maternal grandfather's mother's father would be ancestor number 26. While I could also identify this person as my great-great-grandfather, that definition would also fit one of seven other people.

The PRINT AHNENTAFEL CHART subfunction produces a chart for the selected individual that includes each person's ahnentafel number, occupation, and external reference file name. You may specify which date/place fields to include/exclude in the printout. The 32-generation limitation is no limitation at all in my mind, since that would mean that your records span close to a thousand years. Even if you have an ancestry that goes back more than 32 generations, you can always start another ahnentafel chart for each member of the 32nd generation. Keep in mind that with a chart of that size, there could possibly be over four million direct ancestors listed!

PRINT DESCENDANT CHART — This subfunction produces a 16-generation chart of all direct descendants from the selected ancestor. Although this printout does not show spouses, it is a great chart from which to prepare a published work, since it clearly establishes relationships of all descendants from a common ancestor.

PRINT FAMILY GROUP SHEET — When combined with the pedigree chart, the family group sheet becomes the basis for most genealogy records. This subfunction prints all pertinent data concerning the family of a particular marriage. All text files tied to both the husband and wife may be printed along with the vital statistics.

The family group sheet is also the basic work sheet for genealogists and is used in researching vital records. Before visiting any genealogical repository, be it public library or LDS branch library, I arm myself with copies of the latest family groups for those subjects on which I plan to do research. Because the text files can also be printed with the vital records, this printout is invaluable.

PRINT EXTERNAL REFERENCE — This is primarily an indexing function and does not produce any printout. It allows you to enter page references into the index manually and assign a "priority number" to each.

PRINT INDEX — The index can produce a printed record of all subject and source records in the basefile. Optionally, unreferenced subjects, source records, and subject numbers may be included or omitted.

A large database can produce a very large index. For instance, the average page will hold only 55 names. So if your database has just 1,000 names, it will take 19 pages to print all the names. Right now, my database has about 2753 subject records and requires 51 pages to print the entire index. Therefore, I would have liked to see a two-column format for this printout.

The other thing I would like to see in the index is a date reference for each name. In LDS records, it is recommended that the year of birth be recorded along with the husband's name in the upper left corner of each group sheet. I find this particularly convenient when I have a large number of records with husbands that bear the same name. I think the same capability would be particularly helpful in the index.

RELATIONSHIP — Although I haven't used this function, it establishes blood (not marriage) relationships within 11 generations of each other (systems with more memory will handle up to 15 generation spans). By generation, I mean that the common ancestor (the only way blood relationships are established) must be no further than 11 (or 15) generations away from either of the two people whose relationship is being established. The two people involved must be selected from the f-table. The common ancestor will be identified and can then be moved to the f-table.

Optionally, you may use an ahnentafel number of the first-selected person to identify an ancestor, and then optionally move that ancestor to the f-table.

Relationships are computed and the degree of consanguinity displayed. This degree is described by two commonly-used methods: Civil law and Cannon law. Relationships are also given in either the modern or old convention; some older references call a "first cousin once removed" a "second cousin once removed." The former is the modern convention and describes the son or daughter of your grand uncle or aunt.

The relationship function can also detect redundant ancestors, that is, ancestors that appear twice or more on an ahnentafel chart. This is the case with my Huber ancestors where 2nd cousins, once removed married each other.

SPLIT/MERGE — This function contains three subfunctions: first, to split out families from the database; second, to join two databases; and third, to optimize the database.

When I first received *Roots II*, I loaded everybody I could find into the database. That included parents of people who married brother, sisters, and cousins of my ancestors. The result was a much larger file than I would normally attempt to gather if I had a better understanding of the number of people for which I had data and the limitation of my (then) system.

At first glance, you might think that I would exceed the upper limit of 4,095 names, but in my case, my real problem came from lack of memory in my Z-100. With 192K of memory, I found I was limited to about 1,200 names in the database. Later, when I updated my system to 256K devices, my database grew to 2753 names.

While I was limited to the 192K of RAM in my Z-100, I attempted to use the Split/Merge function. This presented a number of dilemmas, the least of which was the problem of deciding how best to split up the data in my database. The documentation gave me little help in deciding how to go about splitting up the database, so after backing up my files I set out to try several different methods.

My first thought was to break the file at a particular generation in my pedigree chart. That way, I might be able to include my wife's information for our children. In exploring this possibility further, I soon discovered that I would end up having to create a vast collection of smaller families to allow me the luxury of including the records of all lines to the most current generation.

The method I tried was the one I used until I could get more memory; it also appeared to be the best of those I tried. If any of you have any other ideas, I would appreciate hearing from you on how you would go about splitting up a really large family's database.

Rather than try to track the entire family, descendants, and shoestring relatives in one huge file, I elected to choose several distinct ancestors with major families as the basis for each database. To make the selection easier, I chose the surnames of my grandparent's generation. This would normally result in four databases for my records and eight for my children (counting the four databases for my wife's records). However, it did not quite work out that way: I combined my mother's families together since her father's records did not go back in time as many generations as those of my father.

Now the Kauffman family in America was published in a rather large book around the early part of this century and includes the name of my father. To make my records complete, I really should have included that work in my database. But if I did that, I would end up with a database larger than the capacity of *Roots II*. I decided not to, particularly since the work had already been published. Instead, I am including a reference to the work, for those who want to learn more about my shoestring Kauffman relatives.

Because I included cousins' spouses' parents, the split did not work satisfactory when I attempted to use it. But I also believe there could have been a better way to implement the split function.

The split function works its way from a selected descendant (perhaps the compiler or one of his children) up through the pedigree of that person to the farthest direct ancestors. Note that this does not include the ancestors of spouses (in-laws) of the ancestors' brothers and sisters.

From there, the function compiles (probably flags) all the descendants of these ancestors to create a new database. The operator is asked to enter the name of the new database, which is then built and optimized by the program.

The idea behind excluding in-laws' families from the resulting database is to prevent the program from "collecting" all the related people in the database. The result would be another database that would equal the size of the old one. But in examining this problem, I don't see any workable solution to it.

The second fault is that the program does not progress to the latest generation, but only the generation of the selected subject. Therefore, unless you have picked up a person from the latest generation, descendants from later generations are omitted. Again, the idea is a sound one, since if you want to break the database at a particular generation, you don't want to have the later descendants in the database.

The third fault is that the program does not have the option of progressing from an ancestor forward, but only works as I described it earlier: from a descendant backward to the farthest direct ancestors and then forward to the generation of the person selected.

As a suggestion to CommSoft, it would be nice to see the Split function set up as a program with specific options, most notably, the option to select between working from a specific ancestor forward (with a warning that the ancestor is not the oldest if that is the case) or a specified descendant as is now the case. I would also like to see the option of being able to pick either the latest generation (all descendants of all ancestors or the specified ancestor); a specified generation (from the specified ancestor); or the generation of the selected descendant.

As far as I could tell, the current version of the split function works well, although its limitations did not make it particularly practical for my applications. It is possible that others will find it better suits their purposes.

I used the merge function when I expanded my memory. It worked very nicely in combining (in separate steps) four databases.

Optimization simply removes any unused names or places not referred to by a subject or marriage file. It can take a long time to run (over an hour) if the database is as large as mine. The results are a compressed database that uses less memory. Optimization is only needed if you have done a lot of editing that has resulted in deleted entries. Adding information to the database does not create unused names or places.

EDIT — Here is where the real work of Roots is done. The EDIT function contains six subfunctions, all of which I use quit often.

EDIT SUBJECT RECORD — This subfunction allows you to enter or edit the vital statistics for an individual. This includes his *name, sex, occupation,* and dates and places for *birth, christening,* and *death*. Three optional fields provide date and places fields for *burial*, *probate*, and *immigration* records (CommSoft standard); *baptism*, *endowment*, and *sealed to parents* (LDS standard); or three user-defined areas. *Parents* may be entered, but must be pulled from the f-table.

At first, I considered this a real pain, but in thinking back to my own early attempts at a genealogical program, entering data once for an individual makes a lot more sense than repeatedly entering it for each child in a marriage of that individual. Adding parents to a subject record is a matter of pressing and holding down a SHIFT key while pressing two function keys, one for each parent.

Places are available for entering a parent code (Normal, Adopted, Foster child, or X for a non-person) and a birth code (Legitimate, Illegitimate, or Stillborn). As I mentioned before, there is also a place for an 8-character reference field.

Nine fields are available for flags; additionally, dates may be flagged as positive or unsure. If no death date is entered, you may specify that the subject is not living, if that is known.

The last date that the subject record was edited is recorded along with the source provided at that time. If the source has changed, you may have the record reflect the change.

The author and CommSoft have gone the extra mile: After you have finished entering all the data, you can "record" it to the database in memory. When you press the function key that does this, a number of factors are checked, including the parents marriage (if they are really married or not), the date of their marriage against the date of birth, and the dates of birth, christening, and death. If any of these do not relate correctly (that is, appear in the correct order, or appear too close together), you are flagged with an appropriate message. In most cases, you can manually override the message and record the data (for instance, if the child is illegitimate) anyway. Illegitimate children have a field changed to reflect this status, but, in the case of adopted children that may have come from a previous marriage, this field should not be changed. In this instance, you must temporarily "unmarry" the parents to record the subject's record without affecting the status field.

If certain events are out of order (the death date is before the birth date), then you cannot record the data without correcting the problem.

This extensive error checking does not catch all operator errors, but it certainly catches a good many. I was very impressed by this part of the program and consider it one of Roots' real bonuses.

EDIT MARRIAGE RECORD — This subfunction joins subject records together. The two individuals must be of opposite sex and be in the f-table. Two date/place fields are provided: *marriage*, and *divorce* or *annulment*. One optional field is provided: *engagement* (CommSoft standard) or *sealed to spouse* (LDS standard). Of course, it can be user-defined if desired. The last edit date and source are recorded with the record.

EDIT SOURCE RECORD — The latest version of *Roots II* offers the ability to create source records. This is convenient if you are building the majority of records from a single source. But if you are using more than one source, then you will want to use an external text file to record all the sources used to create the record.

There are some differences in a source record over a subject record. First, Roots will not tie text records to source records. This is a real drawback with some of the titles on genealogical books approaching a dozen or more words. It would be nice to be able to enter an abbreviated title in the name field and the full title in a text file.

Places are provided for the library call number, date and publisher (however, the room is limited in these fields), the volume or series number, and the author and place the book or material is found (however, these last two items must be drawn from the f-table).

I would have preferred larger fields for source records, perhaps a different arrangement, and not quite the limitations found in the current setup. For instance, it would make more sense to use a normal date field for the date the material was published.

EDIT: COMPUTE BASEFILE STATISTICS — This subfunction provides a quick summary of the current basefile in memory and of the one that is stored on the disk. When operating under MSDOS version 2 or later, *Roots II* also provides the amount of storage space left on the basefile's disk. The number of records in each of Roots' four subfiles along with the space required are shown, as well as the amount of space still available to Roots in the computer's memory. By dividing the amount of RAM in each subfile by the number of records, you can quickly determine how much an average record needs. Earlier, before I expanded my computer, I used this feature extensively as I approached the upper limit of 192K.

EDIT: SPECIFY PREPARER OR SOURCE — This subfunction allows you to change the name of the source being used in editing. It also allows you to set the name of the preparer for reports. The preparer will seldom change and applies to the entire basefile, not specific records. The source reference is attached to each record as it is edited or created.

EDIT: SAVE FILE ON DISKETTE — This subfunction allows you to periodically update the basefile stored on disk. It is particularly useful if you are working on a somewhat flaky computer or where there might be interruptions in the power to your system.

When the basefile is new and has not been previously saved, you will see two options: *Write new file without backup* and *Exchange disk*. Once the basefile has been recorded, *Save old file as .BAK and write new file* is added to the options. By providing these options, even large basefiles can be saved to standard floppy disks.

The *Exchange disk* option allows you to use *Roots II* on a single-disk system and swap the overlay and basefile disks.

Shortcomings

Although *Roots II* may be the Cadillac of genealogy programs, it is not without its shortcomings. Some of the drawbacks I noted while using the program (but did not necessarily document in this review) are related to the nature of genealogy, and not *Roots II*. Some surprising and unexpected bonuses illustrate that the program's authors have made a very strong effort to make the program error-free, not only in its execution, but also in the results it produces.

Both distribution disks carry the key, so the chances of a catastrophic disaster are very slim. I used *Roots II* extensively for over six months and experienced two hardware failures with an 8-inch drive, but did not lose either distribution disk. During this time, I also converted my Zenith Z-100 to 8-MHz operation and 256K memory ICs. Since the program files were transferred to an 8-inch system, the distribution disks were used only for starting Roots. While the 8-inch drive was being repaired, I used the main distribution disk without any problems for several weeks.

One of the real shortcomings of *Roots II* is the size limitation placed on the maximum number of entries because the entire database is placed in memory. Since I have attempted to write my own family tree programs that were disk-based, I was essentially limited to disk size and record number limits of BASIC. CommSoft has successfully optimized data storage by coding names, dates, and places. This has made it possible to condense the information considerably over any method I used in my own attempts.

However, when you attempt to gather as much information as is in the Kauffman history, you would like to be able to combine the material into a large collection, so that one large publication will cover all descendants, the 4,095-subject entry limitation is soon reached. Obviously, even a disk-based database is going to have limitations unless one has access to a high-density storage device (such as a hard disk) that is relatively uncluttered. Then the problem of speed becomes apparent. Even Roots' speed loses its transparency for the operator as the database grows toward the limit of memory. This is even more apparent in a disk-based system. Even with a limited number of people and marriages, my disk-based system was unbelievably slow when it came time to compile the family. In fact, the system would not complete the compilation process before repeated disk I/O destroyed the data disk on that older 8-inch DEC system. Needless-to-say, I gave up on the project until a more reliable system came along. As a result, I continued to maintain my records on a manual system.

Getting back to the size issue, I soon discovered that the merge program faced the same size limitation on my 192K system that the original database faced. When I attempted to merge two large databases, I ran out of room.

However, even if all the compression methods employed by Roots were used on a disk-based system, the speed could not be duplicated. In fact, the program would be very slow, particularly when it would come to storing information. Hashing methods might speed some things up, but I would think that a minimum of 1 megabyte storage would be needed.

With memory being relatively inexpensive, I expect the capacity of Roots to be increased once again, but only as an option for hard disk users. Perhaps a hard disk version might be provided that has a much larger capacity that would be created by merging a number of standard Roots basefiles. Then, the creation of a work such as the Kauffman genealogy might be practical.

Indexes produced by *Roots II* are of limited use. While they are accurate, I found entirely too many entries to appear to be the same person. That is, it wasn't unusual to find five "Jacob Huber" entries in the index, each listing a series of page numbers. When I examined these pages, I found the entries to be accurate for the five Jacob Huber entries. Of course, each Jacob Huber was a different person, but I could not separate them in the index. I would like to see the birth date in parenthesis added to each individual's entry. Then, I could distinguish the Jacob Huber that was born in 1853 from the Jacob Huber that was born in 1772.

Conclusions

Roots II comes very close to being a software program that bridges the gap between the current type of computer programs, those that require a great deal of computer "savvy" to use, and the next generation of computer programs, those that are almost "transparent" to the user. *Roots II* is a rare program in that it seldom requires the user to think about the computer and that is refreshing. While the screen is often cluttered with a lot of information, its organization is natural, particularly if you have ever used a manual system for keeping genealogical data. In some respects, it is even simpler to use than many manual systems, not only because you record information only once, but because the members of the database are linked together in a number of different ways. As a result, you don't need to worry about forgetting to enter all the information in several different forms.

I have yet to mention the manual, probably because it is *usable*. Most software documentation is either too skimpy or too much. The manual that comes with Roots is written in a narrative form, so it is easy to sit down and read from beginning to end. It also serves as an excellent tutorial. However, tutorials generally do not make good reference manuals. The manual for *Roots II* is different; the index is so good that I seldom had to look in more than one location to find the information I was seeking. If I were to rate the manual on a scale of 1 to 10, I would give it a 10; it is complete, concise, and to the point, without being too cryptic or boring. And that is a difficult achievement.

I have never used a program that has made making a mistake so difficult to accomplish without being able to quickly find it. Except for misspelling names and occasionally missing a date entry, places and events can be easily checked for accuracy.

If the imposed limitations were greater — that is, if the number of entries were limited to less than a couple of hundred names, then the limited number of names might become a major factor in selecting *Roots II* as an aid in genealogical research. But where the limitation exceeds four thousand entries, it becomes insignificant.

The sheer ease of operation, speed in processing the data and data entry, and comprehensive cross-checking of important data far outweigh any shortcomings that I have mentioned. As such, *Roots II* is a real value, even if it were to cost twice as much as the $195.00 for the keyed version or even the $295.00 for the unkeyed version.

The Portable Computer Genealogist

My evaluation of *Roots II* would not be complete without some comments on taking the program and a computer with you. Because my work-related projects involve preparing technical documentation on the two Zenith portable computers, the Z-170 and the Z-180, I have taken the opportunity to use Roots on both these computers "in the field." I can unequivocally state that being able to take a computer-based genealogy database with you into a library is far better than lugging around boxes of paper or finding that you left critical information at home. *Roots II* has tremendous capabilities when it comes to quickly locating people in the database (based on names, dates, or places), and when you include notes on your data disk, you end up with all your critical records at your fingertips.

The Z-180 features 3.5-inch disks that store an amazing 730,000 bytes of data on each disk. With just three small, compact, and durable disks — one for Microsoft Word and its spelling checker, one for the *Roots II* programs, and one for the database and text files — I can take a very large amount of information with me, *wherever* I choose to go. Furthermore, the footprint of the Z-180 is about one square foot, a very convenient size to sit next to a microfiche or microfilm reader or on a library table. And it fits nicely into a 4" briefcase with room to spare for a note pad, pencils, disks, power supply, spare battery, and a research itinerary.

The Z-170 is not quite as convenient because it is slightly larger and won't fit into a briefcase. Its footprint is not much different and it uses the standard 5.25-inch disks, so that means you don't have to convert *Roots II* to the 3.5-inch format. Furthermore, you can run all standard PC-based software, including the keyed versions of Roots and Lotus. While the portability is not quite as good as the Z-180, the shoulder strap works fine and allows you to use a smaller briefcase.

From an operator's standpoint, I found I preferred the keyboard of the Z-180 over the Z-170. The keyboard vendor for the Z-180 is Alps, the same as for the Z-100, Z-150, Z-160, and Z-200. The feel of the Z-170 keyboard is not as good as the Z-180's and the space bar appears to be considerably stiffer. Nevertheless, the size of the typewriter-arranged alphanumeric keys was fine for both machines.

The large screen and keyboard of both units, the memory capacity (640K), the battery power, and the portability make *Roots II*, Microsoft's Word, and the computer one of the best possible investments for the avid genealogist.

Roots II is available from:

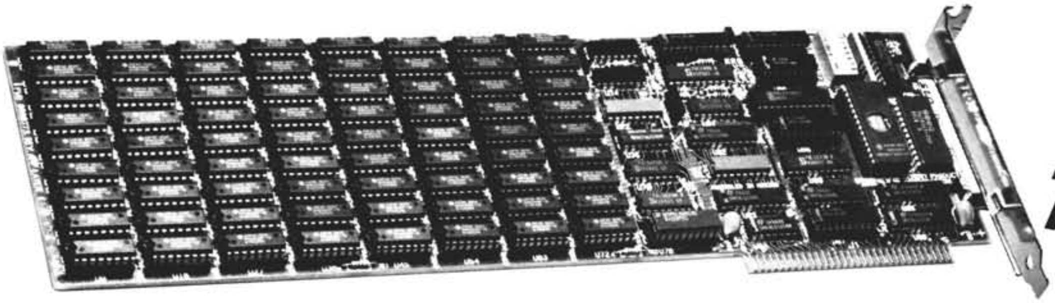
CommSoft
2452 Embarcadero Way
Palo Alto, CA 94303
(415) 493-2184
$195.00 (keyed version)
$295.00 (unkeyed version)

Throughout this review on *Roots II*, I refer to the Church of Jesus Christ of Latter-day Saints as *the LDS church* or simply, *the church* and its members as *LDS*.

Expand your ZENITH PC Memory with

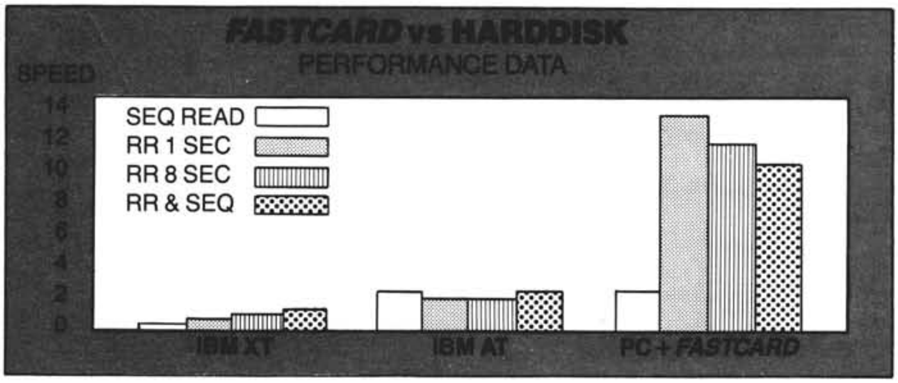
FASTCARD®

fully loaded with 2 *MBYTES* for just $395*



10 DAY FREE TRIAL!

FASTCARD speeds disk I/O by as much as tenfold, *as shown in the following chart.*



This graph shows benchmark test results of a PC equipped with *FASTCARD* versus a PC-XT and PC-AT. These tests were run by a major independent testing laboratory using their standard disk performance benchmark tests.

Specifically designed for compatibility with the 8 MHz Zenith bus, *FASTCARD* provides both extended and expanded memory plus the following standard features:

- Portable between all Zenith models
- Up to 2MB with Split Memory Mapping to
  - Fill memory to 640K
  - Provide Expanded memory over 640K
- Unique Disk Caching
- Ram Disks (up to 8MB)
- Custom Password Security
- Print Buffering
- Built-in Diagnostics and Automatic Fault Tolerance

*Each *FASTCARD III* comes with 2 MBytes of memory for $395. *FASTCARD IV*, available with 1 MByte, includes serial/parallel ports and a clock calendar for $299. *FASTCARD III* without memory is $140 and *FASTCARD IV* without memory is $169.

For additional information, call: PMi PERIPHERAL MARKETING, INC. **(602) 483-7983**
7825 E. EVANS RD., #600, SCOTTSDALE, AZ 85260

Super–Cheapcalc And Mini–Tutorial On Spreadsheets

Part 2

Luis E. Suarez
P.O. Box 66994
Caracas 1061–A
VENEZUELA

To start, here is the ubiquitous question: What is a Spreadsheet? Of course, this question deserves the classical answer: Accountants use those financial sheets, made in green optical paper, having 12, 14 or more columns and a bunch of numbered lines. Usually, the first column is wider and used mostly for items, while the remaining columns are narrower and intended to contain numbers.

In the electronic spreadsheet, each rectangle formed by the lines and columns are called "cells". In those sheets, like Super–CheapCalc, the cell has a variable width and is supplied in huge quantities. At whatever cell, you can write text or numbers and additionally, you can enter formulas or copy one cell content into another, even if it is text, a number, a formula or a calculation result. It is known as "row", the line of cells in the horizontal direction and "column" is the vertical grouping of cells. The columns are labeled from A to T and the rows from 1 to 50. Each cell has its own designation that is a combination of a column and row. Valid names are A1, B8, H23, N50. There are 1000 (count them) cells available in Super–CheapCalc.

Once you call the program from the operating system, the screen is filled with a grid of columns and rows. Each video screen contains 152 cells in a grid of 19 numbered rows and 8 labeled columns, from A to H. The three uppermost lines are used as a sort of control panel. You always know where your cursor is, for the cells are shown as a rectangular block in reverse video and by reading the cell name or address at line number one. At the beginning of the program, the cell at the top/left screen is in reverse video and the first line indicates A1: This means the cursor is at cell A1. The second line is the area where it shows whatever you typed in a cell. As a guide, the cell width is shown at line 1 also, in the form of " |<–nn". The vertical bar delimits the cell space and "nn" is a number corresponding to the cell width. Default width is 9 for all 1000 cells available.

You can move the cursor by simultaneously pressing the SHIFT key and any of the ARROW KEYS in the numerical pad. Each time you press the combination of keys, the cursor moves on the screen to the nearby cell, in the direction according to the selected arrow, and line 1 shows the new cell name, or cell address if so preferred. If you move the cursor farther to the right, the grid is moved to the left. As only 8 columns of 9 characters wide can fill the screen, then the first column is moved away and the column labeled "I" is moved within the screen, a sort of horizontal scroll. In the same way, if you move the cursor down the screen farther than line 19, the grid will move up (vertical scroll) making room for new lines.

Frequently, you need to move the cursor more than one cell. The use of arrows for that purpose is slow and tedious, especially if you move the cursor by a large amount of cells. The GOTO function is what you need. Type f2 and the second line shows: GOTO CELL, <CR> FAST MOVE:". <CR> means Carriage Return. If, on the prompt, you type F38 in capitals only, the display changes and the cursor is placed at cell F38. Type again f2 and <CR> and the cursor will move to cell J38, in a fast cursor movement.

Entering Data

At each cell, you can enter text, numbers or formulas, even if the text is larger than the available cell width. The cell will fill with as much text as possible, but the complete string is stored in memory. Similarly, if you try a number larger than the available cell width, the cell will be filled with a string of "#########" indicating that the number is larger and does not fit the cell width. However, the number is also stored in memory and available for calculation. You must finish the input to a cell with a <CR>.

Super–CheapCalc considers a number as a number, and hence, numerical data may begin with a number from 0 to 9, a - sign or a decimal point. The program accepts ahead of a number a format command, beginning with a colon. The available format commands are :$ (Dollar and Cents), :I (Integers), :F (Floating Point), :% (Percent) and :* (Graphics). All functions beginning with : are recognized as a switch, that is, that the function is valid until another switch is encountered by the program. The default format command is :$. Commas are not allowed within a number.

Valid numbers are:

.55
1234
-9999.99
23E+06
:I88
:$3
:F2.3456
:%10

Super-CheapCalc performs very good logical decisions and hence, it is able to detect a number, a function and text. However, sometimes text needs a prefix to differentiate a reserved math or numeric function from plain text. For instance, it may be necessary to enter text in a cell beginning with numbers or operators like -1 or a string whose first character is reserved, like @ Amp. Use " to tell the program that what follows is text, like "-1 or "@ Amp. To differentiate from non-text input, the string is placed at the cell with a reverse video space following the characters. This mark remains until you perform a recalculation or you move to another screen page. Thereafter, the space remains in normal video.

Whenever you enter a formula using arithmetic operators, be aware of the precedence. Super-CheapCalc does not use the BASIC precedence. Instead, all math is performed sequentially as entered: 2+5*10 results in 70. So, be careful while entering formulas. Besides, parentheses are not allowed.

- 1+1 returns 2
- A1+A2 return the sum of cell A1 content added to cell A2 content.
- A1+A2*13 returns sum of A1 content added to A2 content and the result multiplied by 13.
- (A1*A2+A3) is accepted as text only if preceeded by ".

Let's Start . . .

I will use a different approach in this tutorial. The form widely used in spreadsheet articles for data entry: shift-down-arrow, shift-down-arrow, 200, <CR>, shift-down-arrow, A1*2, <CR>, shift-left-arrow, @SQR(9), <CR> is distracting, difficult to follow and boring. Instead, I prefer the form:

CELL	CONTENT	CELL	CONTENT	CELL	CONTENT
A1	Names	B1	Income	C1	Expenses
A3	Peter	B3	12ØØØ	C3	9ØØØ

The table above means that you must enter the word "Names" at Cell A1, "Income" at Cell B1, the number 12000 at cell B3 and so on. This form is clearer, faster and resembles the printout of a form sheet. Don't forget the <CR> to end the input.

```
ESC-DELETE, WIDTH, FILL THE CELL, &SUM, LOAD, SAVE, PRINT,
HELP COMMAND.
```

To begin with, please type ESC-DELETE to clear the spreadsheet if you have done some practice while reading this article. If not, RUN the program at the CP/M prompt.

I always begin a spreadsheet at line 5 or so, in order to have enough room for titling. Our first sheet is a simple budget application.

Enter the following:

A6	CAR PAYMENT	B6	25Ø	C6	25Ø
A7	CAR FUEL	B7	8Ø	C7	6Ø
A8	HOME GAS	B8	5Ø	C8	87.13
A9	POWER	B9	8Ø	C9	49.55
A1Ø	WATER	B1Ø	25	C1Ø	9.5Ø
A11	PHONE	B11	5Ø	C11	74
A12	RENT	B12	42Ø	C12	42Ø
A13	GROCERY	B13	15Ø	C13	175
A14	MAGAZINES	B14	25	C14	45

The reverse slash \ is used as a command to fill the cell with a string of characters. The form is \[character]<CR>. The sum function is used to add the contents of all cells in a row or in a column. When you type & the program returns &SUM(and you must enter the range of the cells to sum. The program takes care of the close parentheses in case you don't include it. Please Continue . . .

		B15	\=	C15	\=
A16	TOTALS	B16	&SUM(B6-B14)	C16	&SUM(C6-C14)

Now lets include titling:

A1	SAMPLE1				
A3	EXPENSE	B3	BUDGET	C3	ACTUAL
A4	CATEGORIES	B4	AMOUNT	C4	AMOUNT

Your screen shows some truncated words as in cell A4, whose content is CATEGORI instead of CATEGORIES. This is caused by the default width, which is 9 characters per cell. Also note that cell B15 and cell C15 shows a string of = filling the cell. Place the cursor at A4 and you will see the complete string at line 2. As you guessed, the complete string is saved in memory and the cell shows as many characters as allowed by its width.

Press /<CR> to see the available sub-commands. Line 25 shows the available sub-commands, each above the corresponding special function keys. You don't need to type the help command / to enter any sub-command. Just type the desired function key, directly at any time. f3 and f4 are HELP1 and HELP2, respectively. Press f3 and line 25 moves to the second level of the Help command listing 10 math functions. Press <CR> to move to the third level of the Help command and get the remaining math functions. Another <CR>, place the Form Functions available in the fourth level of the Help command at the 25th line and the program control returns to the sheet. Similarly, the f4 function key displays another two levels of the Help command. To exit from HELP type <CR> twice.

With the cursor at column A, type f1 to change WIDTH. The prompt requests a new width value. Type 10<CR> and the screen is repainted showing the new width of column A. If you move the cursor from column A to column B, you will see at line 1 that at column A the width-mark shows ten spaces and at column B nine spaces. That mark is used as a reference while entering strings. Now press the RED key and store the sheet under the name SAMPLE1.DOC. Then type ESC-DELETE and answer the question with 'Y' to clear up the sheet. Congratulations, you have successfully accomplished your first lesson.

Now let's review what we did. Press / to recall what the function key for files retrieval is. Aha! It is the BLUE function key. Press it and answer the prompt with SAMPLE1.DOC. After some clicking, the budget sheet is once again in the screen. Now, we'll use the printer. Yes, you are right, the GREY (or is it white?) key should be depressed. The first prompt asks what you would like to print. Type 1<CR> for printing the sheet. The next prompt asks what area of the sheet you would like to print. Type A1 and C16. Another prompt requests printer character width. Type 2<CR> for 96 characters-per-inch. The sheet is now being printed. Now, press again the GREY key and select 2 for Form Printout. The hardcopy shows the cell name at the left column, the cell content at the second, and the cell result at the third column. Note that cells B16 and C16 shows the formula and the result is shown to the right.

The selection of the character width is important, especially while printing large sheets. Be sure to have your printer ready for this task or the program will hang up until you switch it on.

WIDTH GLOBAL, ?INPUT, (!)RECALC, ERASE

Now, we'll try our second sheet to show the use of the new function ?INPUT. We'll use the same sheet published by Mr. Bob McFarland in REMark issue 44. This sheet, requires a wider column width. 14 characters per cell seems to be OK. To modify all columns, we'll use one single keystroke. Type f1 and at the prompt "WIDTH:" type "14.<CR>" (Don't forget the period!) Presto! All columns are 14 spaces now. Then, enter the sheet as follows:

```
A2   SAMPLE2.DAT    B2   TEST SHEET FOR  C2   CHECKING
A5   INTEREST       B5   .1Ø
A6   INVESTMENT     B6   25ØØ
A7   Compounded     B7   for one year
A8   DAILY          B8   B5/365          C8   1+B8^365*B6
A9   WEEKLY         B9   B5/52           C9   1+B8^52*B6
A1Ø  MONTHLY        B1Ø  B5/12           C1Ø  1+B1Ø^12*B6
A11  SEMI-ANNUALLY  B11  B5/2            C11  1+B11^2*B6
A12  ANNUALLY       B12  B5              C12  1+B12*B6
                    B14  \=              C14  \=
```

The resulting sheet should match the display shown in the mentioned article. So far, so good. Super-CheapCalc has a new function that is very useful. It's the ?INPUT command. The proper syntax is ?[string prompt]<CR>. This command allows you to enter a constant at the time of recalculation, instead of changing the value (manually) at each desired cell. This way is faster if several constants should be replaced in a sheet. Let's see how it works. Place the cursor at cell B5 and type: ?Interest<CR> and use the RECALC command typing !<CR>. The program recalculates all cells from up to down and from left to right (remember this) and stops at cell B5. The second line will prompt: "?Interest:" and waits for your input. This time, please enter .10<CR>. The program resumes and the final calculation is displayed at the screen with the same results as before. Now, try the recalculation command again typing !<CR>. This time at the prompt "?Interest:" type just <CR>. The same interest rate of .10 is still available and the recalculated display so shows. Try different interest rates and compare results. Then try the ERASE command typing the ERASE function key. All results are erased from the screen but the formulas and strings are still present. Try again !<CR> and the sheet once again is recalculated.

Now, let's see the new :% format command. Modify the sheet as follows:

```
B6      :$25ØØ
B8      B5/1ØØ/365
B9      B5/1ØØ/52
B1Ø     B5/1ØØ/12
B11     B5/1ØØ/2
B12     B5/1ØØ
```

Use again the recalc command ! and at the prompt: "?Interest:" type :%10<CR>. Now the cell B5 should shows 10% which looks much better. Once again print the Sheet and the Sheet Form. You can save the data using the RED function key now. Use the name SAMPLE2.DAT.

GRAPHICS, CENTER TEXT, LEFT-JUSTIFY TEXT, MAX, MIN, FORMAT COMMAND

This time we'll layout a display using the graphics capabilities of Super-CheapCalc and how text justification is used. The caret ^ is used to center the text in the cell, the < is used to left-justify the text, and entering text without prefix (default,) will right-justify it. Do not pretend to center an odd string length into an even cell length, or viceversa. It is your responsibility to adjust the cell width if necessary. Use again ESC-DELETE to clear up your sheet. At this time, please perform a GLOBAL WIDTH so that each cell is 30 spaces. Type f1 and 30.<CR> to make all columns 30 spaces and enter the following data:

```
A1       HEATH-ZENITH-          B1       <HEATH-ZENITH
A2       ^HEATH-ZENITH          B2       ^HEATH-ZENITH
A3       <HEATH-ZENITH          B3       HEATH-ZENITH
A4       ^HEATH-ZENITH          B4       ^HEATH-ZENITH
A5       HEATH-ZENITH-          B5       <HEATH-ZENITH
```

Please check your typing and be sure the screen is like this:

```
                 HEATH-ZENITH-HEATH-ZENITH
        HEATH-ZENITH                  HEATH-ZENITH
HEATH-ZENITH                                    HEATH-ZENITH
         HEATH-ZENITH                  HEATH-ZENITH
                 HEATH-ZENITH-HEATH-ZENITH
```

Now use GLOBAL WIDTH, and remember to use the period after the new selected width. Type f1 and respond to the prompt with 6.<CR> Ooopss!!

Please don't tell the Zenith people what we did. It was just a joke to show the versatility (?) of Super-CheapCalc. Nevertheless, here are the real graphics and also we'll learn the use of MAX and MIN function. Please clear your sheet with ESC-DELETE and enter the following data.

```
A1   SAMPLE3.
A3   ^USA         B3    MACH
A4   ^BOMBER      B4    SPEED
A6   B-52 D       B6    :FØ.95          D6     :*B6*1Ø
A7   B-52 G       B7    :FØ.95          D7     :*B7*1Ø
A8   B-52 H       B8    :FØ.95          D8     :*B8*1Ø
A9   FB-111 A     B9    :F2.5           D9     :*B9*1Ø
A1Ø  F-4 C/D/E    B1Ø   :F2.4           D1Ø    :*B1Ø*1Ø
A11  F-111 E/F    B11   :F2.5           D11    :*B11*1Ø
A12  F-16 E       B12   :F2             D12    :*B12*1Ø
A13  A-6 E        B13   :FØ.9           D13    :*B13*1Ø
A14  A-7 E        B14   :FØ.9           D14    :*B14*1Ø
A15  \-           B15   \-              D15    \-
A16  TOTAL        B16   &SUM(B6-B14)    D16    :*B16
A17  AVERAGE      B17   B16/9           D17    :*B17*1Ø
A18  MAXIMUM      B18   @MAX(B6-B14)    D18    :*B18*1Ø
A19  MINIMUM      B19   @MIN(B6-B14)    D19    :*B19*1Ø
A21  \=           B21   \=              D21    \=
A23  ^USSR
A24  ^BOMBER
                  B25   SPEED
A27  TU-96        B27   :FØ.78          D27    :*B27*1Ø
A28  MYA-4        B28   :FØ.87          D28    :*B28*1Ø
A29  TU-16        B29   :FØ.8           D29    :*B29*1Ø
A3Ø  TU-22        B3Ø   :F1.5           D3Ø    :*B3Ø*1Ø
A31  SU-7         B31   :F1.7           D31    :*B31*1Ø
A32  MIG-21       B32   :F2.2           D32    :*B32*1Ø
A33  MIG-27       B33   :F1.7           D33    :*B33*1Ø
A34  SU-19        B34   :F2.3           D34    :*B34*1Ø
A35  SU-17 A35    B35   :F1.6           D35    :*B35*1Ø
A36  \-           B36   \-              D36    \=
A37  TOTAL        B37   &SUM(B27-B35)   D37    :*B37
A38  AVERAGE      B38   B37/9           D38    :*B38*1Ø
A39  MAXIMUM      B39   @MAX(B27-B35)   D39    :*B39*1Ø
A4Ø  MINIMUM      B4Ø   @MIN(B37-B35)   D4Ø    :*B4Ø*1Ø
```

Now place the cursor at column C and make the column four characters wide. Then type the following:

```
C15     \-
C21     \=
C36     \-
```

The @MAX and @MIN functions selects the maximum or minimum value from a range of cells in a row or column. Syntax is @MAX[from-to]. The range should be within a row or column.

Try the command ERASE and ! RECALC. Then move the cursor around the sheet and see how the result is shown at the current cell, while the formula is shown at the second line. The left justified graphics are much better than was in the original program. It is convenient to place a blank narrow column before a graphic column to serve as delimiter, that's why we made the width of column C just 4. Otherwise, the asterisks will merge with the right-justified precedent column. Save your sheet under the name SAMPLE3.DOC. Now, it is time to see the effect and behavior of the formatting commands. Please clear your sheet with command ESC-DELETE. Make the A column 10 spaces wide and enter the following data:

```
A5      :*1Ø
A6      9
A7      :$8
A8      7
A9      :I6
A1Ø     5
A11     :%4
A12     3
```

After typing or after using the RECALC ! command, the screen will display the following:

```
              A
5       **********
6       *********
7             8.ØØ
8             7.ØØ
9                6
1Ø               5
11              4%
12              3%
```

Now move your cursor from A12 up to A1. Your screen will display now:

```
              A
5       **********
6             9.ØØ
7             8.ØØ
8                7
9                6
1Ø              5%
11              4%
12              3%
```

What happened? Just remember that the Format command acts like a switch each time a : is encountered by the program. The program executes the formula as soon as the cursor reaches a cell or from up-to-down and left-to-right if a RECALC ! is done. Hence, moving the cursor from down up, results in the last format command being executed if the current cell does not contain a format command. In the given example, if you move the cursor down again, the result is similar to the first screen. Hence, use a format command in the proper sequence or at every input of a number if several formats are within a sheet.

*

On The Leading Edge

MS-DOS 3.2, PC Perks

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

Zenith has generally done an outstanding job on their operating system software, and I suspect that the MS-DOS 3.2 for the PC series computers is one of their best achievements. During my testing, it appears that they have overcome one of the major IBM hardware limitations that was part of the original design of the IBM XT and AT computer systems — the fact that the hard disk characteristics were part of the system ROM.

It is an unfortunate fact of life that many people have found that the Zenith PC Series and other Zenith compatibles (e.g. IBM, Compaq, etc.) cannot effectively use all the space available on some of the hard disks that are advertised as being "PC compatible". There are, oddly enough in these days of increased emphasis on PC compatibility, a number of manufacturers who insist on developing and selling hard disk with some rather unusual characteristics.

As I mentioned before, one of the biggest problems is that the IBM hardware design standard includes the hard disk characteristics in the system ROM. If the hard disk characteristics are not in the ROM, the only alternative is to pick the closest characteristics (in the Z-200 series SETUP) which generally result in the loss of effective space on the hard disk. It is also important to note that hard disk characteristics are generally contained in the ROM for any XT or AT compatible computer as a result of the hardware design.

That doesn't cause too many problems in the H/Z-148/151/158 computers since all of that is taken care of when PREP is run. In fact, the whole issue usually becomes a moot point on non-Zenith systems since standard IBM systems only allow hard disks with a maximum of 32 megabytes as I discussed in the October 1986 REMark. And so sets the stage for the best part of MS-DOS 3.2.

The PREP Command

Zenith has finally been able to implement a PREP command that will allow the use of just about any hard disk on the PC series computers. This updated PREP command is now available in MS-DOS 3.2 for the PC series computers and includes everything from the H/Z-148 to the H/Z-248.

The best news is that you can now implement just about ANY hard disk with virtually any characteristics on any of the Zenith computers. If you have had a '100 with a hard disk, you are used to the idea that PREP will prompt you for the hard disk characteristics — PREP for the PC series did not since the characteristics were contained in ROM.

The MS-DOS 3.2 PREP command basically asks you for the following information:

- Drive number (e.g. 0, 1, ...)
- Will the disk be a cartridge drive?
- Will the hard disk require track information?
- The number of cylinders in decimal
- The number of heads in decimal
- The starting write precompensation cylinder in decimal
- The landing (shipping) zone cylinder in decimal

As you can see, the above information is critical to being able to identify a hard disk to the system. The biggest caution here is that you MUST have specific information on each of these characteristics for a "non-standard" hard disk. That means that you must specifically check to ensure that this information is included with any hard disk you buy — my experience is that the information is NOT included with most hard disks since they are advertised as "PC compatible" (whatever that means). In particular, you should be especially cautious about buying a hard disk larger than 32 megabytes since it may or may not include the information you will need for the PREP command if it is an unusual type of disk.

There will always be the criticism that Zenith took a long time to implement this feature even though it has been available on the Z-100 series since I have known about them. Why did it take so long? Basically the answer I got is that Zenith wanted to ensure that the implementation would be compatible with current and future releases of DOS for these systems. That involves a significant amount of coordination with Microsoft, and I suspect that took longer than most of us would expect. I doubt that Zenith's hard disk compatibility issues were at the top of Microsoft's priority list which probably accounts for a lot of the delay time. Plus, any major departure from the established "standard" will always require additional testing and such. I personally have spent quite a bit of time fooling around with this new PREP command, and it seems to work quite well.

Speaking Of Hard Disks

One of my friends recently decided to implement a hard disk on his H-151 computer, and that turned out to be a real interesting process. Since the price for a 20 megabyte hard disk and controller was about $200 higher at a Heathkit store, he decided to buy the hard disk and controller at a local computer discount house.

The end result of the installation was that it did not work for several reasons. Perhaps the most frustrating was that we found out that the Zenith boot commands do not work on a non-Zenith hard disk controller since there is some code in the hard disk ROM (on the controller board) that is required to execute the Zenith boot commands like BW (boot winchester). And when the hard disk system started up, the floppy drive made all kinds of interesting crunching sounds which we later found out were a characteristic of the Western Digital controller on a Zenith system.

He finally got the system to work more or less correctly with an Omti hard disk controller, but it was a long, involved process — there are no good installation manuals for IBM related hardware like we are used to in the Heathkit world. He still cannot use the Zenith boot commands — the controller acts like an IBM XT. That is, it first looks for a boot disk in drive A. If none is found, then it tries to boot the hard disk. If you have never seen an IBM XT boot a hard disk, that can be an interesting experience since I have seen many people forget that they had a data disk in drive A. Then DOS gives you an error message saying something like there is no system on the floppy.

Despite all of these problems, it is difficult to reconcile the cost of a Heath/Zenith hard disk and controller with another of comparable capacity. It is not at all difficult to buy a 20 megabyte hard disk for under $400. The HS-317-20 is listed at $895 in the current catalog. I think more than a 100% premium is too much for some of the Zenith features. But back to MS-DOS 3.2.

MS-DOS Version 3.2

I should probably be quite specific at this point and note that this version 3.2 of MS-DOS is available only for the PC series computers. I doubt that we will see the H-100 in the catalog much longer, and I do not expect to ever see an "official" version of DOS for my H-100 system.

Although there have been a number of command additions and enhancements, I will mention just a few of them here. For the Zenith unique commands, there has been a change to CONFIGUR which is linked to a new command called DSKSETUP. And a couple of new commands, ZCOM and ZSPOOL, perform file transfers between computers and allocate a print buffer respectively. And Zenith has now included a "Setup" disk that makes it even easier to install new software. This is a nice command that makes it easy for anyone to install a new version of MS-DOS on a floppy or hard disk system. IBM has absolutely nothing like it.

New standard DOS commands for version 3.2 include REPLACE and XCOPY. When I say that these are standard DOS commands, this means that these same commands are included in PC-DOS 3.2. I found that even though MS-DOS 3.2 includes the APPEND command, PC-DOS 3.2 does not. As you will see, APPEND provides some useful support, and I think it is surprising that IBM did not include it in their 3.2 release.

CONFIGUR And DSKSETUP

The only noticeable change to the CONFIGUR command is that the option for manual/automatic partition assignment has been moved to the DSKSETUP command. But this one will sneak up on you if you use batch files to change CONFIGUR options like I do.

It turns out that a lot of the letters for the option selection are quite a bit different from version 3.1, not to mention version 2. It turned out to be quite an exciting operation to watch CONFIGUR change some things that I really didn't want to change. It is always a dumb assumption to believe that things will always work like they did in a previous version of an operating system, and I should know better. I did not lose any data, but it took me a while to get my system back into a "normal" configuration.

The DSKSETUP command is interesting and provides us with a number of things that we can speculate about. DSKSETUP provides three options: change the automatic partition assignment flag, configure the floppy drive type, and change hard disk format protection.

The automatic partition assignment flag selection performs just like it did in version 3.1 — no real change there except that the command was under CONFIGUR.

The most interesting change is the "configure floppy drive type" option. You can select four drive numbers to configure (numbers 0-3) three types of drives: a 360K, 1.2 MB or 720K (3.50" drive) format. I suppose that the 720K format is to allow compatibility with the new 3.50" drives that IBM decided to make available on the XT 286 system, so it will be interesting to see how that works. I wonder when we will see a 3.50" drive upgrade kit available for our systems . . .

The last DSKSETUP option allows you to enable or disable hard disk format protection. This is a nice little feature that can help you protect yourself against the strange and wonderful mysteries of the FORMAT command. Although I have seen a public domain utility like this around, Zenith is the first manufacturer that I have seen include this as part of the operating system software.

ZCOM — A File Transfer Utility

I suppose that it was inevitable that Zenith would add something like this to the operating system software given the wide variety of computers on the market today. This utility allows you to transfer files between computers using either modems or a direct serial connection. The documentation indicates that any baud rate up to 115200 can be set by the program for a communications port. I bet that speed really makes a system hum, and it would probably be interesting to try to run it on a Cray II. I wonder if the Cray is IBM PC compatible . . .

In any case, the ZCOM utility must be running on both computers — one computer acts as the "server" and the other operates in the "user" mode.

ZSPOOL — Creates A Print Buffer

This command allows you to create a print buffer in conventional (up to 640K) or expanded (above 1 megabyte) memory. You can allocate up to 320K of conventional memory or a whopping 4200K of expanded memory for use as a print buffer.

REPLACE — Replaces (Updates) Files With New Versions

The REPLACE command is an interesting addition to the DOS command set. Although it is not clear to me why Microsoft thought it would be useful in DOS, both 3.2 versions of Zenith MS-DOS and IBM PC-DOS contain the REPLACE command.

The command is used to automate the updating of file versions which is an interesting concept. The most common use that I can think of is, of course, the update of DOS and application software. REPLACE will look at the files on the target disk and replace them from the source disk if they have exactly the same name. That can be an unusual experience with program files particularly if the manufacturer has changed from a COM file to an EXE file or vice-versa. The point is that the file names must be exactly the same or they will not be replaced. As I tell my students: "Close is fine for horseshoes, but it doesn't work for computers".

XCOPY — An Advanced File Copy Utility

Have you ever tried to copy files from a subdirectory and forgotten to copy files in a lower level subdirectory? Windows, for example, has a subdirectory that contains all of the PIF files, and it is easy to forget to copy that subdirectory. If so, you will certainly be pleased to see the XCOPY command.

XCOPY allows you to copy files and subdirectories (with the /S switch if they exist) with incredible ease. Perhaps the most important feature of this command is that it provides an easy way to backup files in a standard DOS format. It resembles the DOS BACKUP command in many ways, although aside from the format difference, XCOPY does not copy files across multiple disks like BACKUP can. Still, it is a very useful command since you can copy files that have been updated (based on the archive bit) or you can copy files that have been modified on or after a specified date.

APPEND — Specifies A Data File Search Path

When Microsoft introduced the concept of the command PATH in version 2 of DOS, I thought it was a mystery that a similar command was not included for data files. As you may know, a number of public domain programs provide the data file search capability similar to the PATH command, plus a few software vendors have introduced similar programs.

APPEND finally allows you to specify a search path for data files. The syntax is suspiciously like the PATH command so you will not have to learn ANOTHER strange type of command form. Except for the fact that APPEND is not a built-in command like PATH, everything seems to work like you would expect.

The Zenith documentation notes that APPEND searches a specified path for files only when a program performs the following DOS functions: Open a file with FCB (0Fh), Get file size (23h), and Open a file handle (3Dh). Since there is no way to know ahead of time how a program works, you will have to experiment to see if the search path works for a specific program.

The New Perks-PC

If you have taken a look at some of the ads in this magazine, you have probably noticed that Barry Watzman has released Perks-PC for the PC series. Perks-PC also runs on compatible IBM systems as well. And it works just fine on my '241. It is quite similar to the Z-100 Perks (the original), and the capabilities and functions are the same in both. In case you missed the original Perks review in the August 1985 REMark, most of the comments here also refer to the updated Z-100 version too.

Perks-PC (and Z-100 Perks) has all of the needed features in the "desktop" utility packages . . . a calculator, an appointment calendar, an ASCII table with graphics characters for programmers, a "notepad" with editing features, a card file, autodialer, and various configuration options in the "filer". For those of you concerned about the problem of "burning" an image on your monitor when it's left on for long periods of time without use, Perks has a Screen Saver which turns off the video after a set "inactivity threshold" (which you can adjust). The screen image is restored when you press any key on the keyboard — a particularly useful feature for me.

Perks-PC is loaded into memory one time, and can then be activated at any time (even during another program like WordStar, Lotus 1-2-3, etc.) by pressing both shift keys (Shift-Break on the Z-100). Windows are used for both Perks versions and are provided by Barry's own PANE-RELIEF window manager. That is, you can activate one or more Perks' functions at any time, and the window(s) will be displayed on the screen.

Like Z-100 Perks, Perks-PC is menu driven and provides appropriate help screens. Obviously Perks-PC can adjust for either color or monochrome, and although it's great in monochrome, it is really spectacular in color on the C. Itoh CM-1000 color monitor.

The Calculator provides the normal 4 functions with memory and provides the capability to do operations in both hex and decimal. Function keys and the keypad are used in the operation of the calculator.

The Notepad provides basic editing functions using WordStar-like commands. Cursor movement including scrolling, insert/delete functions, search/replace operations, and block/file functions are all provided. As if that weren't enough, the display size of the notepad is also adjustable, and you can also use the import module to copy any of the alphanumeric contents of the screen that existed prior to the activation of Perks. If you were working on a SuperCalc file prior to activating Perks, that means you can copy part of the spreadsheet into the notepad.

The ASCII Table shows all of the standard 128 characters plus the standard PC graphics characters starting at 128 decimal. Scrolling through the table is easy . . . just use the PgUp and PgDn keys. The Perpetual Calendar shows any month in the 20th century. The Appointment Calendar can be activated by ALT-L and allows you to create, change or print your appointments.

The card file feature allows you to set up a miniature database like a phone directory or just about anything else you can think of. It is particularly neat since it allows "free form" comments so you are not limited to a specific format like a phone directory. In addition, you can use the card file with the autodial function to dial telephone numbers.

The Filer and Setup module provides a number of features. You can adjust the Screen Saver threshold, set timers, list the directory, erase files, set the default drive and directory, and define the filespec for the Perks files.

Movement within Perks is accomplished by using the ALT key in combination with the first letter (usually) of the function. As previously mentioned, Perks itself is activated by pressing both shift keys which results in a display of the menu. If you need the calculator, press "C". If you're not in the menu and want to bring up additional modules of Perks, you can use the ALT key in combination with the first letter of the function as usual. For example, if you're currently using the calculator and decide that you need the ASCII Table, ALT-A will display the table leaving the calculator displayed on the screen. All of the functions can be displayed in a similar manner. You can also press ALT-H at any time to get information for Perks.

As many of you know by now, I am personally against any of the "copy protection" schemes since they are a pain in the neck for users, particularly when you have a hard disk. For those of you who don't know, I will not even review any of the copy protected software in this column as a matter of personal policy.

Perks is not copy protected, and it can be used to import and export information from most other programs. Barry has always provided excellent software (and hardware) to the Heath/Zenith user community, and Perks-PC is no exception.

Both Perks-PC and the original Z-100 Perks are very useful desktop utilities, and both are highly recommended. They are easy to use, well documented, and performs a number of functions that I have found to be very helpful.

Closing REMarks

There are a number of items that I am waiting for right now, so we will see what develops for next month.

I'll be glad to answer any questions about information in this article if you enclose a stamped, self-addressed envelope with your letter.

Products Discussed

Software

MS-DOS Version 3	
PC only (OS-63-32)	$150.00
Z-100 only (OS-63-30)	150.00

Hardware

Advanced Personal Computer (HS-248)	$2499.00
Monochrome/Color Video Card (Z-409)	239.00
20MB Winchester (ZD-200)	1499.00
40MB Winchester (ZD-400)	2499.00

Heath/Zenith Computer Centers
Heath Company Parts Department
Hilltop Road
St. Joseph, MI 49085
(800) 253-7057 (Heath Catalog orders only)

Perks (Z-100 MS-DOS only)	$69.95
Perks-PC (PC Series and IBM)	69.95

Heathkit Stores (Perks only)
Barry Watzman
560 Sunset Road
Benton Harbor, MI 49022
(616) 925-3136
Master Card and VISA accepted

*

ZPC Update #13

Pat Swayne
Software Engineer

This is the thirteenth in a series of articles in support of ZPC, a program that allows you to run IBM PC software in H/Z-100 (dual processor) computers. ZPC is available from HUG as part no. 885-3037-37. An upgrade disk for ZPC is also available as part no. 885-3042-37.

In this installment of the never ending (so it seems) ZPC Update series, I will present a couple of little support programs that may help you run some things you possibly haven't been able to run before, and I will also present patches for these programs: Framework II version 1.1, SuperCalc 4 version 1.0, Javelin version 1.1, and Quick Dos version 1.21.

Help For The New ZHS Circuit

In the last ZPC Upgrade, I presented a new version of the ZPC Hardware Support (ZHS) circuit. It is similar to the circuit on the Scottie Board, a commercially available ZHS circuit. The new circuit (whether mine or Scottie's) supports an emulated video control port that the old versions did not support. The video control port is supported by ZPC Version 2 after it has been upgraded to version 2.1 using the Upgrade disk. I have found that the video control port may not work properly with a few programs, and so you might want to disable it if you find a program that ran properly before acting up under ZPC Version 2.1 and the new ZHS circuit. You can disable the video control port using DEBUG as follows:

```
A>DEBUG
-OF1,7F
-Q
A>
```

You can also re-enable the port with DEBUG:

```
A>DEBUG
-OF1,3F
-Q
A>
```

The above methods should not be used if INT14 (the serial port driver supplied with the Upgrade disk) is loaded. I have written a short program called VIDCON.COM that can be used to disable or enable the video control port, and it works whether INT14 is loaded or not. For those of you who have the MS-DOS assembler (MASM), the source for the VIDCON is listed at the end of this article. If you have ZBASIC or BASICA, you can create the VIDCON by typing in and running the following program.

```
10 REM THIS PROGRAM CREATES VIDCON.COM
20 DEFINT A-I:OPEN "O",1,"VIDCON.COM
30 S=0:S1 = 6692 :FOR I=1 TO 75
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 186,36,1,180,9,205,33,128,62,93
80 DATA 0,79,186,61,1,228,241,116,7,186
90 DATA 68,1,36,191,235,2,12,64,230,241
100 DATA 180,9,205,33,205,32,13,10,86,105
110 DATA 100,101,111,32,99,111,110,116,114,111
120 DATA 108,32,112,111,114,116,32,105,115,32
130 DATA 36,111,102,102,46,13,10,36,111,110
140 DATA 46,32,13,10,36
```

To use VIDCON, copy it to your ZPC system disk, and enter

```
VIDCON OFF
```

to turn the video control port off, or

```
VIDCON ON
```

to turn it back on.

The Screen Width Variable

On a real PC-type computer, the width of the screen in characters is stored at location 40:4A in memory, with a zero at location 40:4B, so that the width can be read as a word. When ZPC is in the

PC mode, it also stores the width at 40:4A, but at 40:4B there is a jump to the Z-100 BIOS print utility that is preserved even in the PC mode so that PSC (print screen) utilities will work. That works out ok usually, because most PC programs access the width variable only as a byte. However, I recently came across the first known program to access the variable as a word, and it did not work properly under ZPC. The program was a demonstration version of the Active Investor Series.

You can patch the first byte of the print routine jump to zero and make Active Investor work by using DEBUG as shown below, but be sure that ZPC is in the PC mode before you do.

```
A>DEBUG
-E40:4B
0040:004B  E9:00
-Q
A>
```

This patch will remain in effect until you put ZPC back into the Z-100 mode. The next time you enter the PC mode, the patch will have to be re-done if you want the byte zeroed. I have written a short program called FIXWI.COM that can make the patch for you. To create FIXWI, use the source code at the end of this article, or type in and run the following BASIC program.

```
10 REM THIS PROGRAM CREATES FIXWI.COM
20 DEFINT A-I:OPEN "O",1,"FIXWI.COM
30 S=0:S1 = 8446 :FOR I=1 TO 105
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 184,64,0,142,216,128,62,0,0,233
80 DATA 117,6,186,34,1,235,9,144,198,6
90 DATA 75,0,0,186,73,1,14,31,180,9
100 DATA 205,33,205,32,13,10,83,119,105,116
110 DATA 99,104,32,116,111,32,80,67,32,109
120 DATA 111,100,101,44,32,116,104,101,110,32
130 DATA 114,117,110,32,70,73,88,87,73,46
140 DATA 13,10,36,13,10,66,73,79,83,32
150 DATA 115,99,114,101,101,110,32,119,105,100
160 DATA 116,104,32,105,115,32,102,105,120,101
170 DATA 100,46,13,10,36
```

To use FIXWI, just type

```
FIXWI
```

at the system prompt, and hit RETURN. ZPC must be in the PC mode when you run it. You will have to run FIXWI each time you enter the PC mode if you want the width variable fixed, because ZPC restores the print utility jump when you enter the Z-100 mode, and leaves it alone when you re-enter the PC mode.

Framework II Version 1.1 Patch

To make version 1.1 of Framework II work under ZPC, add these lines to your PATCHER.DAT file:

```
FRAMEWORK II v. 1.1
Insert system disk 2.
   (You must also use FIXFWII, unless ZHS is used.)
FW.OVL
15B59,B0
15B5B,0,0
```

Use PATCHER as directed in your ZPC manual to apply the patch to System Disk no. 2 of the Framework set. Then put ZPC in mode 7 (type PC 7) and run the Framework SETUP program. Select item 1 from the first menu if you have not already installed Framework for use on floppies or your hard disk. When you get to the list of screen drivers, select item 4 (All other IBM drivers). On the list of IBM drivers, you can select 2, 3, or 4. If you select item 2, you will have to put ZPC in mode 7 each time you want to run Framework. If you select 3 or 4, you will be running Framework in the default PC mode (mode 3).

When you get to the printer port menu, select number 1 (parallel) regardless of whether your printer is actually connected by a parallel or serial cable. ZPC will take care of mapping the Framework printer driver to your printer.

When you have completed the setup procedure, run SETUP again, but select item 2 from the first menu. When you get to the Main menu, select item 4 (Telecommunications), and from that menu, select item 4 (Port Addresses). If you do not have any PC-type serial ports installed in your computer (such as the optional ports on the Scottie Board), set all of the port base and vector addresses to 0. If you have one PC serial port, set the base and vector addresses for ports 2 and 3 to 0. If you have two ports, set only the port 3 addresses to 0. Now type M to get back to the main menu, and select item 7 to save your changes.

After you have run SETUP, you will have to patch the file it made using the FIXFWII program that comes with ZPC Version 2. To use FIXFWII, copy it to your Framework system disk 2, log on to the disk, and enter

```
FIXFWII
```

and hit RETURN. Framework II will now be ready for use with ZPC.

SuperCalc 4 Patch

If you have a ZHS board installed, you can run SuperCalc 4 without patching it, but if you do not have the board, you will have to add some data to your PATCHER.DAT file and make a patch. As with SuperCalc 3, there seems to have been more than one release of SuperCalc 4 version 1.0. The differing releases are distinguishable only by the date on the SC4.COM file (and other files) on the disks. Use the DOS DIR command to see the date on your SC4.COM file. If it is either 6-17-86 or 10-24-86, then you can use the patches presented here. If you have the 6-17-86 release, add these lines to your PATCHER.DAT file.

```
SUPERCALC 4 v. 1.0 (IBM, 6-17-86)
Insert Product Disk no 1 (containing SC4.COM and SCRIBM DRV)
SC4.COM
349E,90,90,90,90,90,90,90,90,90,90,90
34B8,90,90,90,90,90,90,90,90,90,90,90
B4BC,90,90,90,90,90,90,90,81,E3,FF,7F
D90C,90,90,90,90,90,90,90,90,90,90,90
D92C,90,90,90,90,90,90,90,90,80,E1,7F
35F6,90
3612,90
D52C,90
D52F,90
D532,90
D535,90
D53B,90
x
SCRIBM.DRV
23AC,90
23B3,90
z
```

If you have the 10-24-86 release, add these lines to your PATCHER.DAT file.

```
SUPERCALC 4 v. 1.0 (IBM, 10-24-86)
Insert Product Disk no 1 (containing SC4.COM and SCRIBM.DRV)
SC4.COM
3445,90,90,90,90,90,90,90,90,90,90,90
345F,90,90,90,90,90,90,90,90,90,90,90
B509,90,90,90,90,90,90,90,81,E3,FF,7F
D8DF,90,90,90,90,90,90,90,90,90,90,90
D8FF,90,90,90,90,90,90,90,90,80,E1,7F
```

```
359D,90
35B9,90
D4FF,90
D502,90
D505,90
D508,90
D50E,90
x
SCRIBM.DRV
2432,90
2439,90
z
```

Make the patch using PATCHER as directed in your ZPC manual.

Javelin Version 1.1 Patch

Javelin will not run in the default PC mode (PC 3) without patching under ZPC, even if you have a ZHS board installed, because it accesses the video retrace port on the board using a word input instruction (IN AX,DX) instead of a byte instruction. The ZHS board normally toggles a couple of bits each time the retrace port is accessed so that the software sees the bit in the opposite state each time it looks at the port. However, when the IN AX,DX instruction is used, the bits are toggled twice for each access, and the software always sees the bit at the same state, and hangs up.

To patch Javelin for use with ZPC, add these lines to your PATCHER.DAT file.

```
JAVELIN version 1.1 Startup disk
Insert the Startup disk.
JAV0.OVL
B4C8,90,90,90,90,90,90
B4DA,90,90,90,90,90,90
B517,90,90,90,90,90,90
B529,90,90,90,90,90,90
B57C,90,90,90,90,90,90
B59A,90,90,90,90,90,90
B601,90,90,90,90,90,90
B61F,90,90,90,90,90,90
B65A,90,90,90,90,90,90

B676,90,90,90,90,90,90
B708,90,90,90,90,90,90
B71A,90,90,90,90,90,90
B74E,90,90,90,90,90,90
B762,90,90,90,90,90,90
B7A5,90,90,90,90,90,90
1E307,90,90,90,90,90,90,90,90,90,90,90,90
1E318,90
1E31E,90
1E32C,90,90,90,90,90,90,90,90,90,90,90,90
z
JAVELIN version 1.1 Program disk
Insert the Program disk (JCONFIG must have been run first).
IBMGRAPH.DRV
1247,0,0
124B,B0
1E36,90,90,90,90,90,90,90,90,90,90,90,90
1E47,90
1E4D,90
1E5B,90,90,90,90,90,90,90,90,90,90,90,90
z
```

Set ZPC in the default PC mode (PC 3), and run INSTALL (if you want to put Javelin on a hard disk or an 8" disk (treat it like a hard disk), and then run JCONFIG. When JCONFIG displays three squares on the screen and asks if you see red, blue, and green squares, you can answer Y even if you have a monochrome screen, but color memory in your computer. That way, you will see gray scale. Next, JCONFIG will display light and dark yellow squares and ask if you see them as different colors. It does not matter how you answer that question. When it asks you to choose the manufacturer of your screen adapter, select the first item (IBM). When it asks the kind of IBM adapter you have, also select the first item (Color Graphics Adapter). You can answer the printer questions as required.

After you have run JCONFIG, make the required patches with PATCHER to your startup and program disks if you are running from floppies. If you are running from a hard disk or 8" drive, temporarily copy PATCHER.COM and PATCHER.DAT to your JAV directory, log on to the JAV directory, make both patches, and then erase PATCHER.DAT and PATCHER.COM from the JAV directory.

If you want to run Javelin in video mode 7 (PC 7), you do not need to install the above patch. You should run JCONFIG in mode 7, and when it asks what kind of IBM adapter you have, select the last item.

Quick Dos Version 1.21 Patch

Quick Dos can be run without patches if you have a ZHS card, but the video control port must first be turned off if you have the new ZHS circuit and ZPC version 2.1. To patch it for use without ZHS, or for use with it without having to bother with the video control port, add these lines to your PATCHER.DAT file.

```
QUICK DOS version 1.21
Insert the disk containing QD.EXE, QDSTART.EXE, and
     QDCOLOR.COM
QD.EXE
3E1,90
1AD9,90
361D,90,90,90,90,90,90,90,90,90,90
A70E,90,90,90,90,90,90,90,90,90,90
A76C,90,90,90,90,90,90,90,90,90,90
D63C,90
x
QDSTART.EXE
F19,90
F2A,90
x
QDCOLOR.COM
97,90
19C,90
1ED,90,90,90,90,90,90,90,90,90,90
528,90
535,90
57A,90,90,90,90,90,90,90,90,90,90
z
```

Make the patch with PATCHER, and then run QDSTART to install Quick Dos as described in the manual.

HUG NEW PRODUCTS

P/N 885-8047- 37 CP/M
P/N 885-8048-37 Z-DOS/MS-DOS
P/N 885-8049-37 MS-DOS
Accounting System $20.00

Introduction: Accounting System is a very user-friendly, double-entry accounting program capable of handling up to 999 separate accounts during any calendar year. It permits easy interaction between specified accounts, in an add/deduct condition, during the printing of the depreciation, returns, allowances, etc., if such interaction is required. All options and account parameters are easily set on initialization, and are readily modifiable at any time. Payroll disbursement is not included.

Requirements: Accounting System is available in versions for CP/M, Z-DOS/MS-DOS (for the H/Z-100 . . . not PC), and MS-DOS for the H/Z-100 PC compatibles. Since all system calls are generic in nature, no special system modifications should be necessary. The CP/M version works with either CP/M-80 or CP/M-85.

Dual disk drives or a hard disk system are highly recommended for program execution and data storage. The CP/M version will be in soft-sector only.

An 80-column printer (tractor-feed recommended) is required for report listings.

For the MS-DOS versions (885-8048 and 885-8049), COBRUN .EXE (which is included) must be present at run time.

The CP/M version requires 64k of memory. The MS-DOS versions require at least 94k, exclusive of the operating system.

The following files are included on the various distribution disks:

CP/M Version: HUG P/N 885-8047- 37

ACCOUNT1	.COM
ACCOUNT2	.COM
ACCOUNT3	.COM
README	.DOC

Z-DOS/MS-DOS Versions: HUG P/N 885-8048-37 and P/N 885-8049-37

COBRUN	.EXE
ACCOUNT	.EXE
README	.DOC

Authors: John A. Liotta and Carl D. Rise

All three versions also come with an extensive users manual.

Program Content: During program execution, the following capabilities are available from the main menu:

A = Add New Daily General Journal Entries
B = Print the Monthly Trial Balance

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.

C = Add Monthly Journal Adjusting Entries
D = Print Monthly Adjusted Trial Balance
E = Print Income Statement and Balance Sheet
F = Add Monthly Journal Closing Entries
G = Print the Post-Closing Trial Balance
H = Update the Chart of Account Control File
I = Print your Chart of Account Control File
J = Print an Active Account Recapitulation
K = Print the Daily General Journal Entries

ESC = Return to System

Comments: Excellent low cost small business or personal accounting system!

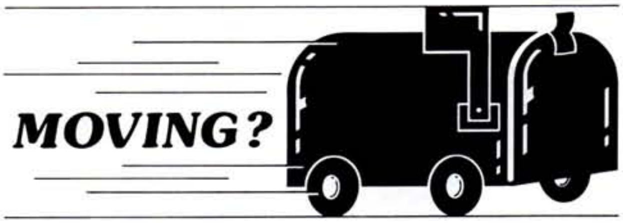
TABLE C Rating: (10)

HUG P/N 885-3027-37 & 885-3028-37 HUGPBBS for MS-DOS UPDATE

Since its' first release in July of 1985, HUGPBBS has undergone many changes, including bug fixes, and enhancements. As its' author, it was my first exposure to assembly language programming under MS-DOS. Although not alien to assembly language programming (I'm a 10 year veteran), it was a learning experience. Since that time, I've learned a lot, and current and would be owners of HUGPBBS can all benefit. The latest version of HUGPBBS, 1.50.M, is available, to current owners, FREE! Simply return your original distribution disk (or disks, if you have the source code), to Nancy Strunk here at HUG. Your disk(s) will be updated at no extra charge, and returned to you. For those of you who are interested, the following are some of the enhancements added to HUGPBBS since it was first introduced:

- Eliminated password echo at caller's screen
- Sysop can flag the caller to <T>alk if he wishes
- CRC protocol added to XMODEM file downloads
- Connect time feature added
- <M>inutes function added to check connect time remaining
- Date and Time stamping on caller printout
- Sysop private message length changed from 3 to 5 lines
- Additional information can now be added to the user log
- Added <A>nswer command to the Scan & Retreive function
- Grouped all system files into a single drive designation making configuration even simpler
- Source assembly no longer requires the DEFMS.ASM file
- Three different 32 megabyte partitions can now be used for the data base. Use of these different drives is totally transparent to the caller
- The upload drive can now be specified seperately from the data base and system file drives
- Software now responds immediately to a loss of carrier
- Many obscure bugs and oversights have been eliminated

UPDATE YOUR COPY TODAY!



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

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.

HDOS HARDCOPY SOFTWARE

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

GAMES

HDOS

Part Number	Description of Product	Selling Price	Vol. Issue
885-1010	Adventure Disk H8/H89	10.00	4
885-1029-[37]	Disk II Games 1 H8/H89	18.00	8
885-1093-[37]	D&D H8/H89 Disk	20.00	16
885-1124	HUGMAN & Movie Animation Pkg	20.00	41
885-8009-[37]	HDOS & CP/M Galactic Warrior	20.00	32
885-8022	HDOS SHAPES	16.00	45
885-8032-[37]	HDOS Castle	20.00	59

CP/M

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

Part Number	Description of Product	Selling Price	Vol. Issue
885-3004-37	ZDOS ZBASIC Graphic Games	20.00	37
885-3009-37	ZDOS ZBASIC D&D	20.00	50
885-3011-37	ZDOS ZBASIC Games Disk	20.00	52
885-3017-37	ZDOS Contest Games Disk	25.00	58
885-8042-37	ZDOS/MSDOS Poker Party	20.00	77

UTILITIES

HDOS

Part Number	Description of Product	Selling Price	Vol. Issue
885-1025	Runoff Disk H8/H89	35.00	
885-1063	Floating Point Disk H8/H89	18.00	
885-1079-[37]	HDOS Page Editor	25.00	15
885-1082	Programs for Printers H8/H89	20.00	
885-1089-[37]	Disk XVIII Misc H8/H89	20.00	20
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
885-1126	HDOS Utilities by PS:	20.00	42
885-1127-[37]	HDOS Soft Sector Support Pkg	30.00	45
885-1135-[37]	HDOS Variety Pkg	20.00	76
885-8001	SE (Screen Editor)	25.00	28
885-8004	UDUMP	35.00	28
885-8006	HDOS SUBMIT	20.00	31
885-8007	EZITRANS.	30.00	30
885-8017	HDOS Programmers Helper	16.00	42
885-8024	HDOS BHBASIC Utilities Disk	16.00	46

CP/M

Part Number	Description of Product	Selling Price	Vol. Issue
885-1212-[37]	CP/M Utilities H8/H89	20.00	21
885-1213-[37]	CP/M Disk Utilities H8/H89	20.00	22
885-1217-[37]	HUG Disk Duplication Utilities	20.00	26
885-1223-[37]	HRUN HDOS Emulator 3 Disks	40.00	37
885-1225-[37]	CP/M Disk Dump & Edit Utility	30.00	40
885-1226-[37]	CP/M Utilities by PS:	20.00	40
885-1229-[37]	XMET Robot Cross Assembler	20.00	40
885-1230-[37]	CP/M Function Key Mapper	20.00	42
885-1231-[37]	Cross Ref Utilities for MBASIC	20.00	43
885-1235-37	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
885-5003-37	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

Part Number	Description of Product	Selling Price	Vol. Issue
885-3005-37	ZDOS Etchdump	20.00	39
885-3007-37	ZDOS CP/EMulator	20.00	47
885-3008-37	ZDOS Utilities	20.00	47
885-3010-37	ZDOS Keymap	20.00	51
885-3022-37	ZDOS/MSDOS Useful Programs I	30.00	63
885-3023-37	ZDOS/MSDOS EZPLOT	20.00	63
885-3026-37	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
885-3039-37	ZDOS/MSDOS HelpScreen	20.00	82
885-3042-37	MSDOS ZPC Upgrade Disk	20.00	83
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

Part Number	Description of Product	Selling Price	Vol. Issue
885-3012-37§§	ZDOS HUG Editor	20.00	52
885-3014-37§§	ZDOS/MSDOS Utilities II	20.00	54
885-3016-37§	ZDOS/MSDOS Adventure	10.00	57
885-3020-37§	MSDOS HUG Menu System	20.00	62
885-3021-37§§	ZDOS/MSDOS Cardcat	20.00	63
885-3024-37§	ZDOS/MSDOS 8080 To 8088 Trans	20.00	64
885-3025-37§§	ZDOS/MSDOS Misc. Utilities	20.00	64
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
885-3040-37§	MSDOS HADES	40.00	83
885-3041-37§	MSDOS ScreenDump	20.00	83
885-8039-37§§	MSDOS DPATH	20.00	74
885-8040-37§§	MSDOS HELP Programs	20.00	74
885-8045-37§§	MSDOS MATT	20.00	80
885-8046-37§	MSDOS ASM Language Utilities	20.00	82

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

PC/IBM COMPATIBLE

Part Number	Description of Product	Selling Price	Vol. Issue
885-6001-37	MSDOS Keymapper	20.00	59
885-6002-37	CP/EMulator II & ZEMulator	20.00	59
885-6003-37	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

Part Number	Description of Product	Selling Price	Vol. Issue
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
885-1132-[37]	HDOS Tiny BASIC Compiler	25.00	59
885-1134	HDOS SMALL-C Compiler	30.00	63

CP/M

Part Number	Description of Product	Selling Price	Vol. Issue
885-1215-[37]	CP/M BASIC-E	20.00	26

BUSINESS, FINANCE AND EDUCATION

HDOS

Part Number	Description of Product	Selling Price	Vol. Issue
885-1070	Disk XIV Home Fin H8/H89	18.00	
885-1071-[37]	MBASIC SmBusPk H8/H19/H89	75.00	17
885-1131-[37]	HDOS CheapCalc	20.00	47
885-8010	HDOS Checkoff	25.00	32
885-8021	HDOS Student's Statistics Pkg	20.00	44
885-8027	HDOS SciCalc	20.00	50

CP/M

Part Number	Description of Product	Selling Price	Vol. Issue
885-1218-[37]	CP/M MBASIC Payroll	60.00	31
885-1233-[37]	CP/M CheapCalc	20.00	47
885-8011-[37]	CP/M Checkoff	25.00	32
885-8036-[37]	CP/M Grade	20.00	70

ZDOS/MSDOS H/Z100 ONLY

Part Number	Description of Product	Selling Price	Vol. Issue
885-3006-37	ZDOS CheapCalc	20.00	47
885-3013-37	ZDOS Checkbook Manager	20.00	54
885-3018-37	ZDOS Contest Spreadsheet Disk	25.00	58
885-8028-37	ZDOS SciCalc	20.00	50
885-8030-37	ZDOS Mathflash	20.00	55
885-8043-37	MSDOS Calc	20.00	80

DATA BASE MANAGEMENT SYSTEMS

HDOS

Part Number	Description of Product	Selling Price	Vol. Issue
885-1107-[37]	HDOS Data Base System H8/H89	30.00	23
885-1108-[37]	HDOS MBASIC Data Base Sys.	30.00	23
885-1110	HDOS Autofile (2 Disks)	30.00	23

CP/M

Part Number	Description of Product	Selling Price	Vol. Issue
885-1219-[37]	CP/M Navigational Program	20.00	31

MSDOS

Part Number	Description of Product	Selling Price	Vol. Issue
885-6008-37	MSDOS NAVPROG	20.00	73
885-8034-37	DBZ-A Database For The Z100	25.00	69

AMATEUR RADIO

HDOS

Part Number	Description of Product	Selling Price	Vol. Issue
885-8016	Morse Code Transceiver Ver 2.0	20.00	42

CP/M

Part Number	Description of Product	Selling Price	Vol. Issue
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

Part Number	Description of Product	Selling Price	Vol. Issue
885-8038-37	MSDOS RFCAD Ver. 3.50	30.00	73

COMMUNICATION

HDOS

Part Number	Description of Product	Selling Price	Vol. Issue
885-1122-[37]	HDOS MicroNET Connection	16.00	37
885-8005	MAPLE (Modem Appl. Effector)	35.00	29

CP/M

Part Number	Description of Product	Selling Price	Vol. Issue
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
885-5006-37	CP/M-86 HUGPBBS	40.00	62

Continued on Page 47☛

Now There Is A PERKS For PC Owners

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

A Brief Overview/ Highlights Of "PERKSPC" . . .

HUGCON V Comments

Before I start my discussion on the new version of PERKS, I would like to take this opportunity to make a few comments on the recent HUG Conference in Chicago, HUGCON V.

HUGCON V was my third conference and I am really sorry that I missed the first two. The conferences just keep getting better. Heath/Zenith users from around the world come to the Annual International Conference for many reasons: to hear the various session speakers, to talk and trade tidbits about their Z-machines, to get some exceptional bargains on hardware and software, to hear the various guest speakers at the welcoming session and Saturday dinner, or just to take a few days of vacation to the Chicago area. I guess I go for all the above reasons.

As usual, there was a wide range of interesting topics to listen to over the two days of sessions. I personally try to attend as many of the talks as I can (there were eight sessions, so I went to eight talks) to get as much information about what others are doing with their Z-machines and what more I can do with my machines.

The Saturday night dinner was just super, not only the food but the guest speakers, the awards including the Outstanding Heath/ Zenith User award that was given to me, and topped off with the show that Bob Ellerton and Jim Buszkiewicz put on in the raffling off of the many Heath and Zenith products. It was truely a great evening to remember. I would like to say "Thank You" again to all those at HUG who made the above possible.

Now with HUGCON V behind us, we can start looking forward to the Midwest HUG Conference in the Spring, HUGCON VI in August and the CHUG conference in October, and possibly the Western Regional HUG conference in November. In the meantime, we continue to work like little beavers with our Z-machines.

PERKSPC

Now lets talk about Barry Watzman's latest version of PERKS, the Desktop Utility package that Barry originally wrote for the H/Z-100 machines. The newest version is the PC version or what Barry calls: PERKSPC.

At the 1985 HUG Conference, Barry was selling PERKS V1 for the H/Z-100 machines and many of us, and that includes myself, purchased copies for our H/Z-100's. The features and capabilities of Version 1 were limited in comparison to some of the desktop utility packages that were available at that time to users that had PC-type machines. However, we started to use PERKS and before long we were dependent upon the features that were available in Version 1.

Then came Version 2 in the Spring of 1986 with many additional capabilities. These new capabilities were not necessarily unique amongst desktop utility packages but they were better in a number of ways: quality, more individual features per capability or function (in other words, one can do more with Barry's functions), and more complete, such as Barry's modem module. A number of articles have been written in the past, including Pat Swayne's article in the September 1986 Issue of REMark (Volume 7, Issue 9), on PERKS that describes the features in detail. See Pat's article for some details on the new features of Version 2. I will not cover the details.

Since a number of Heath/Zenith users were still using Z-DOS V1 (and may very well still be using Z-DOS today), Barry made sure that Version 2 worked on Z-DOS, as well as MS-DOS Versions 2 and 3. However, now we had an excellent desktop utility package for our H/Z-100s but we couldn't use the same package for our PC series machines. Yes, many of the Heath/Zenith users have more than one Heath/Zenith machine including one of the PC types: H/Z-150 up to and including the H/Z-200. It sure would be nice if there were a PC version of PERKS. Never fear, Barry is here and he came through again.

About a couple of months before HUGCON V, I heard a rumor that Barry was working on a PC version of PERKS, but it was never advertised as an upcoming product or even mentioned that he would be demonstrating it at HUGCON V. Barry was indeed running PERKS on a PC-type machine at the conference. From all appearances, it looks exactly like the H/Z-100 version. Barry calls this PC version the "PERKS Desktop Utility for the IBM-PC and Compatibles."

Just as with the H/Z-100 Version of PERKS at HUGCON IV, the PC version of PERKS was selling at a special conference price at HUGCON V of $49.95 versus the regular price of $69.95. Since Barry just finished the PC version, what he was selling was a Beta Test version which he felt was working quite well with no known bugs at the time of the conference. Before making a general announcement, Barry wanted to do some extensive testing of all the features on a variety of machines and graphics adaptors. All those getting the Beta Test version can get an updated version at the time it is formally released. To get an updated version, one only needs to send in the original disk to Barry.

I purchased a copy and so far I have not encountered any problems. The PC version is identical as far as I can see to the H/Z-100 version in features/capabilities except for a number of "keyboard" differences. The documentation that I received with the PC version is the H/Z-100 Version 2 manual with a two-sheet differences document. For those who already have PERKS for their H/Z-100, the following section, describing the differences, will be of importance to you. Following that section, I will briefly list, with a short description, all the features (or modules) that are in Version 2 of PERKS for those users who either don't have an H/Z-100 or have one but don't have PERKS. This will give the latter users some idea of the features and power of the PERKS package.

PERKS PC Differences

At this point I would like to go through the differences between the H/Z-100 and PC versions, and also point out where future changes will be made to the H/Z-100 version to bring the two versions closer together (in other words, to try to keep the "keyboard" differences to a bare minimum).

1. The default activation key sequence is pressing both the left and right SHIFT keys at the same time instead of using the SHIFT-BREAK key sequence as for the H/Z-100.
2. Since there is no HELP key on the PC as such, ALT-H is used instead.
3. To move between modules within PERKS, the ALT-x sequence is used instead of the BREAK-x sequence. The x is the module select character. The reason for not using the BREAK key here and in 1 above, is the fact that on PC-type machines, the BREAK key is really a CTRL-SCROLL LCK key sequence. This makes it difficult to press a sequence like CTRL-SCROLL LCK-x.
4. To remove PERKS, uses the same key sequences (in name only) as that for the H/Z-100 which is "BREAK, SHIFT-BREAK". Remember what was said in 3 above, the BREAK on the PC is CTRL-SCROLL LCK. So the sequence is as follows: "CTRL-SCROLL LCK, SHIFT-CTRL-SCROLL LCK".
5. Function key assignments have changed because the H/Z-100 has 13 function keys (F0 through F12) while the PCs have only 10 F1-F10). As far as I can tell, the function key assignments start with F1 on the PC version while they start with F0 on the H/Z-100. In a future release of PERKS for the H/Z-100, the function key assignments will be changed to match those used in the PC version. This is what I meant earlier when I said changes will be made to keep the differences to a bare minimum. In the Filer and Setup module, 12 function keys are used. In the PC version SHIFT-F1 and SHIFT-F2 are used for the last two. Again, the H/Z-100 version will be changed to reflect this difference.
6. The PC version uses the ALT-arrow key sequence to change the size of the NOTEPAD window, whereas the H/Z-100 version used the BREAK-arrow sequence.
7. To move the PERKS module windows on the screen, the PC version uses the CTRL-arrow sequence rather than the SHIFT-arrow sequence for the H/Z-100.
8. The last thing that is different that I can see is the way one can change the "activation" key sequence which is one of the functions in the Filer and Setup module. I will not go into the new procedure, but will say that is very different from the procedure used in the H/Z-100 version.

Barry mentioned two other things that are worth bringing up at this time. First, although testing is not complete, PERKS is believed to work with most graphics boards except for the EGA board when using a mode greater than 7. The second item is that the PC version seems to be working just fine on AT-type machines including the Z-200. However, more testing is to be done here.

All you H/Z-100 owners who have PERKS for the H/Z-100, and also have a PC-type machine, can now have the same excellent Desktop Utility package for your PC machines (Pat Swayne, in your September 1986 article you ended it with the statement that you wished that there was a PERKS for your Z-158 - here you are). For those who do not have PERKS, here is a chance to get an excellent product for your H/Z-100 or PC-type machines.

Brief List/Description Of Features

- Calculator. The Calculator is the standard four-function calculator with memory. However, this calculator also has HEX mode and Logical operations on HEX values. Integer and floating point are both supported.
- Notepad. This is a full-window (the size of which can be changed) text editor intended for generating/editing short notes. The editor has a wide range of commands which are WordStar-like commands.
- ASCII Table. Provides access to the numeric codes in both decimal and HEX corresponding to the 128 characters of the 7-bit ASCII character set.
- Perpetual Calendar. A perpetual calendar for the 20th century.
- Appointment Calendar. This is a sub-module of the perpetual calendar which is used to enter and save appointments on disk.
- Cut/Paste. Allows the transfer of text information on screen generated by any applications program to the same applications program or to another one.
- Card File. This is basically equivalent to the "Rolodex" type filing system of up to almost 3000 entries each consisting of up to five lines of 32 characters each.
- Modem. Provided to support communications with a remote computer or database, even while in the middle of another applications program. It supports the XMODEM protocol.

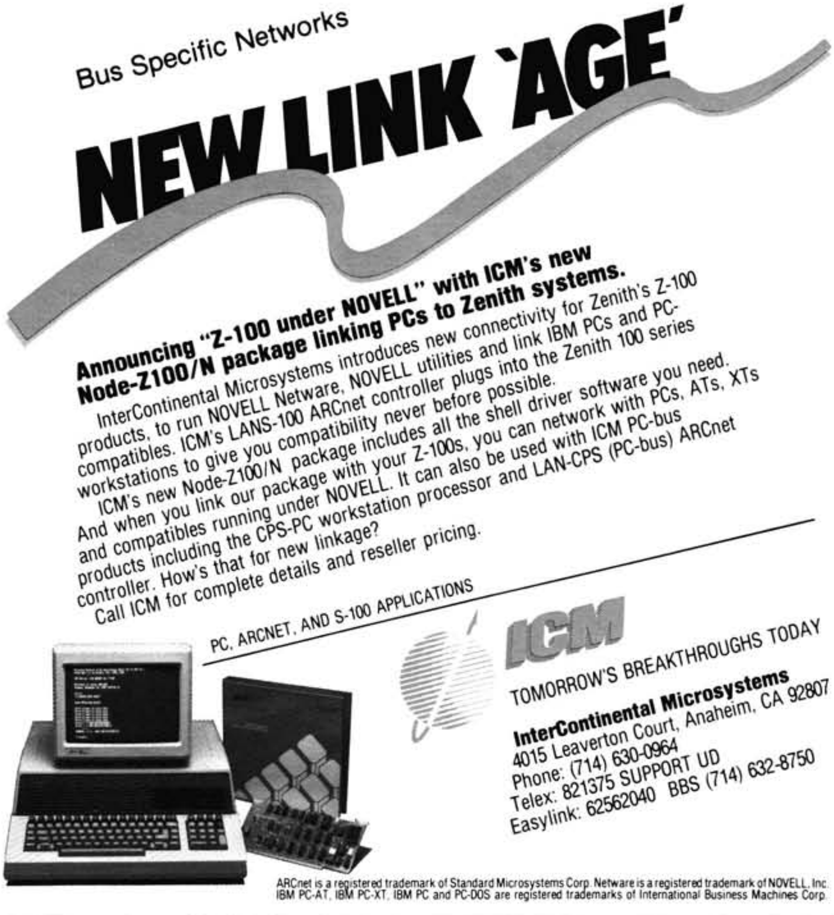
- Phone Dialer. Supports dialing a Hayes Smartmodem or equivalent external modem connected to one of the H/Z-100's internal serial ports. It can be activated via the Card File, the Notepad, or main menu.
- Typewrites. Provides a simple means to use the computer keyboard with your printer to provide typing on the printer directly from the keyboard. Useful in addressing envelopes.
- Filer and Setup. This supports a number of miscellaneous functions such as: set one or more of the eight alarm timers, set the screen saver threshold time (i.e, will blank screen if no activity on keyboard or CRT for specified time), list the directory of any disk on the system, erase files, saves the position of all PERKS windows for future use, set current default drive, set current directory, modify the PERKS activation key sequence, and set up the file specifications for a number of PERKS files such as the appointment file, card file, etc.

That is all I have to say on PERKS V2 for the PC-type machines. As I said before, I am sold. Since Barry Watzman was selling the PC version at HUGCON in August '86, I am sure that anyone who wants one now can place an order with him for the $69.95 price, the current price for the H/Z-100 version. Hope all of you who have PERKS have been enjoying it as much as I have. It makes some of my tasks easier.

Barry A. Watzman
560 Sunset Road
Benton Harbor, MI 49022-7142
(612) 925-3136

Keep up the good work, Barry!!!

*



Continued from Page 44

Part Number	Description of Product	Selling Price	Vol. Issue
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

Part Number	Description of Product	Selling Price	Vol. Issue
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

Part Number	Description of Product	Selling Price	Vol. Issue
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.

*

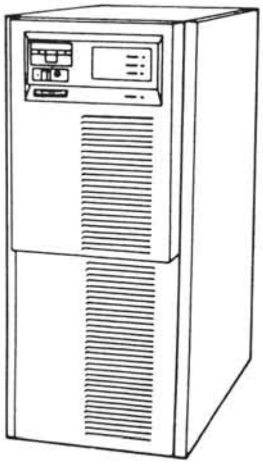
Now with Cambridge Script Processing

HONEYWELL USERS

Connect Your Z-100 Series Or Z-100 PC Series Computer To Honeywell Host Systems

The ultimate micro-to-mainframe link

PC77/78 — Emulates the Honeywell VIP7700/VIP7800 series of synchronous terminals



Features:
• color attribute mapping • hot key to DOS • access host via multiple windows • print redirection • script processing • help facility • print spooling

File Transfer:
• menu driven • macro capability to bypass menus • data compression • file identification security features • attended and unattended mode of operation • restart capability • help facility

PC73/78 — Emulates the Honeywell VIP7200/VIP7300/ VIP7800 series of asynchronous terminals

Features:
• color attribute mapping • hot key to DOS • print redirection • script processing • help facility • print spooling

File Transfer:
• Kermit • Xmodem • ASCII • menu driven • macro capability to bypass menus • attended and unattended mode of operation

The personal computer can be connected to any Honeywell host via a modem, modem-bypass or multiple interface unit (MIU) and may reside on a communications line with other PC's or terminals. No host system or application software modification, regardless of the operating system is required.

High-performance communications software from:

Cambridge Computer Corporation
80 Mount Sanford Road, Mount Carmel, CT 06518-1210
203/288-6004, Telex 6502236599

Honeywell is a trademark of Honeywell, Inc.

Enhanced PAINT.ASM!!!

Steven W. Vagts
9509 Gray Mouse Way
Columbia, MD 21046

ADD a `HELP' function and the capability to `MOVE' or `COPY' blocks on the screen to other areas of the "PAINT.ASM" screen!

The Challenge

As some of you may recall, the March and April issues of "REMark" contained my articles on "PAINT.ASM". The articles described a program which permitted the drawing of a graphics screen, in a relatively simple process, and upon completion of the "picture", loading it into computer memory, saving it to disk, or dumping it to a dot matrix, near letter quality, multi-mode printer, such as the Panasonic KX-P1092. The program also permitted the loading of special character fonts, developed by us and easily modified, to use with the printer.

I only received a few letters expressing problems with the program and I hope that my responses assisted those individuals satisfactorily. I also appreciate the kind words of praise I received from several of you.

In part one of the article, March issue of "REMark", I had placed a challenge to the readers to present ideas on how one might be able to move blocks of text from the screen to other areas of the screen. I'm sorry to say that no one returned with any ideas.

So . . . I took the project on, and little did I know what I was getting us into. This month, to continue with my purpose of developing interest in assembly programming, I thought I would share the trials and tribulations of this enhancement.

Setup And Background

For those of you who kept a copy of the program presented in "PAINT.ASM — Part 1", dust it off — it's easiest to begin here for the purpose of this article. Also, I'm sure some of you may not have done Part 2 because you didn't have the type printer that would be needed. This enhancement does not require the printer routines, and will still round out your program's capabilities nicely. If you only kept the whole program, that's fine also — just make a copy and use that for starters. DON'T MESS WITH YOUR ONLY COPY.

For those of you who purchased my complete program, the changes are more significant. If you will just send me your hard-sectored disk again, I will update your software free of charge.

Since my version of PAINT was fully utilizing all of the function keys, I first had to free up several of these keys for other purposes. The color keys not used in the article's version of PAINT were utilized for various delete functions — erase to beginning of line, erase to end of line — and a 'help' function.

These functions I combined, such that upon using the `ERASE' key, the program looks for an arrow to be typed next. The arrow then determines the direction from the cursor that will be erased — up erases to beginning of page, down erases to the end of the page, left erases to the beginning of the line and right erases to the end of the line. Also, typing `HOME' at this point erases the entire page. On any other keystroke, the erase function is ignored. This worked out very well and freed up two of the desired color keys.

The `HELP' function, which displayed a row of alphanumerics on the top line and the corresponding graphics characters on the second row, was changed to respond to a `CTRL-G'. The addition of these functions will be discussed below.

Now that I had the keys I needed, I decided to use the `BLUE' key to `mark' the upper left corner and the lower right corner of a desired block of text. It would place an asterisk in each place as a visual reference. The `RED' key would perform the `move' function — place the desired block of text in the new location and delete it from its old position on the screen. The `WHITE' key would perform the `copy' function — place the desired block of text in the new location and replace the previous data that was overwritten by the two marks.

By using the `IC' (Insert Character) key before doing a `move', the block of text could be inserted in the appropriate area of the screen, rather than overwriting text already in that area. This will not work for the `copy' command, however, because the need to redisplay the screen before copying disables the `IC' function. It would also be nice if the `copy' command could make repetitive copies without having to remark the block, but I haven't been able to do that yet.

Theory

The first step toward approaching the programming problem was to develop an outline of what was intended to be done. Break the problem into blocks of functions, then you attack each function separately — the old divide and conquer rule. This was easily done already because of the natural break created by the separate functions of each of the colored keys.

Next, I created a preliminary outline of what each function will need to do, with ideas on how to proceed. For the `BLUE' key, for example, this was relatively easy and can be broken down as:

a. Move cursor to upper left corner of desired block of text — depress `BLUE' key.
b. Save present contents of screen to a known memory location — use SAVE routine.
c. Report cursor position, using `ESC-n', and save this information in a buffer area labeled MARKS.
d. Place an asterisk in this screen location as a visual cue.
e. Inform the operator to move to lower right corner of the desired block.
f. On second depression of `BLUE' key, do steps "c" and "d" again, then inform operator to move the cursor to the upper left corner of the new desired location.
g. Return to normal input mode.

For the `RED' or `WHITE' keys, things got more hairy:

a. Move cursor to upper left corner of new desired location — depress `RED' key.
b. Save cursor position, using `ESC-j'.
c. Reload saved screen on display — removes the asterisks from the screen.
d. Move cursor to previously saved position, using `ESC-k'.
e. Report cursor position, using `ESC-n', and save this information in MARKS also.
f. Test relationship of the two marks to determine legal block description.
g. Compare first marker with origin (upper left corner of screen) to determine beginning of desired block of information in the screen buffer, BUFR.
    1. Get row count down from origin to determine how many times to skip 80 characters.
    2. Get column count from origin to column of first character of interest.
    3. Set an `insert' flag to begin inserting characters on the screen.
h. Compare column information of two marker locations to determine how many characters are to be inserted from each row of the desired block.
i. Compare row information of two marker locations to determine how many rows of new information we will insert.
j. Maintaining count of characters on each line, do a loop, inserting characters from the BUFR to the screen in the desired place.
    1. Watch the character count in each line — to determine the start of a new line.
    2. Watch the number of characters to be inserted on each line — to determine whether `insert' flag is on or off.
    3. Watch the number of rows completed — to know when done.
k. Delete prior characters from screen.

The list for the `WHITE' key is identical to the `RED' key except line k. is not done for `WHITE' and line c. is not done for `RED'.

This list went through numerous changes before finally getting a working solution. From here, a logic flow chart can easily be developed before the actual program is written.

Keeping the number count correct was the most critical part of the programming involved in this change. From what little I know of other computers, there is generally a video map in memory that is an exact duplicate of what is displayed on the screen. I have not been able to locate a similar area in the H/Z-89, which is probably the reason the "transmit page", ESC-#, function is necessary. Such a memory map of the screen would permit simply poking new information as necessary and would probably save considerable work on our part.

As we saw in Part 1 of "PAINT.ASM", the "transmit page" function causes the contents of the screen, including its imbedded escape sequences, to be sent to memory — at location BUFR at the end of our program. The screen is transmitted from the upper left corner of the screen, 80 characters per line, including blanks, and escape codes do not count as part of the 80 character line. At the end of line 24, the first and only carriage return is sent, marking the end of transmission.

Since the entire screen of characters is kept as one group, there is no way to find a specific location, say the tenth character of the third row, for example. We have to look at the BUFR and count 250 characters before we have the correct spot. If we wanted to move the next ten characters, we would read these, display them on the screen in the desired location, then continue our count to 80 and begin again to locate the start of the next line of desired characters. All this time we must continue to account for escape codes.

To make matters worse, when an escape code is encountered, it applies to all the characters that follow, until another escape code is encountered to change it back. Reverse video, for example, remains in effect for following characters until another escape code causes an exit to normal video. When we move blocks of text, we must be watchful of this. If we wanted to copy a portion of reverse video text, starting at some point after the first character, to another area of the screen, we would only end up moving normal text unless we remember that this text was reverse video and take appropriate action to account for this.

The easiest way I have found to do this, especially since we have to check for escape codes anyway to keep accurate count, is to send all escape codes to the console, whether we want to insert the next character or ignore it. It doesn't matter to the console if it receives 3 sets of alternating 'ESC-p's and 'ESC-q's before it gets its first character to insert, as long as the right code is active when we insert the character.

The same thing occurs after our desired block. If we don't reset escape codes after the block is copied, any additional characters sent to the screen will continue with the same characteristics set. It took me a while to figure out why, when I exited the program because of an error, I was still in the reverse video mode.

Modifications To PAINT.ASM (Published In March And April, 1986)

Before I start on the other changes, there was one minor problem with the length of file names that I discovered since PAINT.ASM was published. It would not permit file names of eight characters, unless a decimal and file type (those letters to the right of the decimal) were used. This is remedied by removing the last line of the FILIT routine, just before FILTYP, which read

```
      JNZ      LNERR    ;LENGTH TOO LONG
```

and inserting:

```
      JZ       FILTYP
      CPI      ' '      ;BLANK?
      RZ                ;END OF NAME
      JMP      LNERR
```

The following changes are necessary to incorporate the new enhancements:

In the INPUT routine, change the two

```
      JZ       DELET
```

lines to

```
      CZ       DELET
```

Add four new lines immediately following these in the INPUT routine:

```
      CPI      Ø07H     ;CTRL-G - DISPLAY GRAPHICS
      JZ       GRAFX
      CPI      Ø02H     ;CTRL-B?
      JZ       RBLUE
```

In the DELET routine, replace the last line

```
      JMP      INPUT
```

with "RET".

Add the following lines to the end of the XCAPE routine, just before the TXSPCL routine:

```
      CPI      Ø4AH     ;ERASE KEY?
      JZ       ERASR
      CPI      Ø5ØH     ;BLUE?
      JZ       BLUE
      CPI      Ø51H     ;RED?
      JZ       RED
      CPI      Ø52H     ;WHITE?
      JZ       WHITE
```

Following the EGRAF routine, and before the MSG50 routine, add the lines shown in Listing 1.

Several changes must be made to the DSPLY routine. First, change the name of the routine to DSPLY1. Next, just before the DSPLY1 routine, insert the two lines:

```
DSPLY   CALL     DSPLY1
        JMP      EMSGD
```

Then, replace the third line in the DSPLN routine, which read

```
      JZ       EMSG     ;END DISPLAY
```

with

```
      RZ                ;END DISPLAY
```

Just after the PMSG routine, and before the DELAY routine, we need to add the following routine, a long (in relative terms) time delay:

```
TDLY    PUSH    H
        PUSH    B           ;16 BIT TIME DELAY
        MVI     B,45        ;ADJUST AS NECESSARY
TDLY1   MVI     C,255
TDLY2   NOP
        NOP
        DCR     C
        JNZ     TDLY2
        DCR     B
        JNZ     TDLY1
        POP     B
        POP     H
        RET
```

Carefully change the message section of the program so each of your messages are the same as those messages given in Listing 2. DON'T REMOVE OTHER MESSAGES. Ensure all quotes and dollar signs are placed properly — an area that I suspect caused numerous problems before — at least it did me. I changed the HEXadecimal codes to decimal to reduce program size and improve readability. If you prefer, you need not change these.

Finally, after the line beginning

```
    CHK     DB....
```

insert the following lines:

```
RSCNT   DB      Ø       ;START BLOCK COUNT - ROW
CSCNT   DB      Ø       ;START BLOCK COUNT - COL
RCOUNT  DB      Ø       ;ROW COUNT
CCOUNT  DB      Ø       ;COL COUNT
INSFG   DB      Ø       ;1 IF INSERTING CHARS
MOVFG   DB      Ø       ;Ø IF MOVE; 1 IF COPY
MARKFG  DB      Ø       ;Ø=MARK1; 1=MARK2
MARKS   DS      12      ;MARKED CURSOR POSITIONS
```

That completes the changes.

Problem Areas

While the modifications themselves went much easier than I expected, I did have a severe problem that took several weeks and much hair pulling (no surprise that I'm balding) to finally solve, though I'm not happy with the solution.

The problem involved trying to move large blocks of characters at a time. This was extremely frustrating because I had no problem moving small blocks of text. When I moved text and deleted it from the old area, I could only move approximately 32 characters total — whether it was two rows of 16 each, or a single line of 32. If I attempted to move more than that, the end of line 25, displaying the key representation following the move, would mess up — characters were skipped and escape codes were not recognized.

If the block was even larger, line 25 wouldn't print at all and the other 24 lines of text would begin to mess up. The copy function would do the same thing after about 190 characters. The only real difference between the two functions was the deletion of text. Performing a `CTRL-L', displaying the screen from memory, would cancel the operation and bring back the screen as it last appeared. You could then move a smaller block. It also showed that the BUFR was not the problem.

Finally, I found that the characters were being placed on, and removed from, the screen too fast!!? First, I inserted the DELAY routine after each line moved and after each line of a block being deleted — which cured the problem!!! Then, I inserted various versions of the new TDLY routine, building up time delays until I reached the current version, presented above, which is a substantial delay for computer evolutions.

Running the console at 9600 baud, I really didn't expect this problem. Maybe one of you could enlighten me as to the true cause and solution. In the meantime, if you get similar symptoms, try increasing the TDLY routine. It also wouldn't hurt to reduce the number temporarily and try moving the top 11 lines of a crowded screen to the bottom half, just to see what actually happens.

I discovered another problem recently that occurs when a screen is saved to disk. Lines of complicated graphics — more than 30 to 40 'ESC-F's and 'ESC-G's, intermixed with some reverse video — caused the line to truncate to 10 or less characters when saved to disk. Saving to memory and reloading to the screen had no problem with the line, and it would also print fine on the printer.

The problem only occurs with a complicated line, and cleaning or shortening the line slightly takes care of it. I've tried limiting the number of characters sent to disk to 128 byte blocks, DMA (Direct Memory Address) record block size, but it only complicated the program and didn't cure the problem. I'd simplify the program here by not worrying about sending unnecessary spaces to the disk, which alleviates the need to check for reverse video characters too, but that would no longer permit superimposing disk stored graphics over the current screen — the entire screen would be overprinted with spaces. The problem was so minor that I finally decided just to live with it.

Closing

This completes my work on "PAINT.ASM". I hope that the above information has been helpful in your endeavors with other programs. Through these articles, I especially hope that I might have encouraged at least a few of you to give assembly language a try. It represents a challenge I have not found in other programming efforts.

I would be happy to send the source code for everything except the data files (these consume a lot of paper and are easy to develop) for $4.00. I'll also send the finished product on disk for $10.00, if you include two preformatted disks with your request. I generally use H/Z CP/M 2.2.4 and single-sided, single-density, hard-sectored disks. I also use CDR CP/M and CDR double-sided, double-density extended, soft-sectored disks. Please include a phone number in case I have problems.

I'm interested in any improvements readers feel may be appropriate and I'd be happy to address questions if self-addressed, stamped envelopes are included.

Listing 1
Enhanced PAINT.ASM!!!

```
ELNE    MVI     A,Ø4BH   ;ERASE TO LINE END
        JMP     TXSPCL
ELNB    MVI     A,Ø6FH   ;ERASE TO LINE BEGIN
        JMP     TXSPCL
EPGE    MVI     A,Ø4AH   ;ERASE TO PAGE END
        JMP     TXSPCL
EPGB    MVI     A,Ø62H   ;ERASE TO PAGE BEGIN
        JMP     TXSPCL
EALL    MVI     A,Ø45H   ;ERASE SCREEN
        JMP     TXSPCL

GRAFX   LXI     D,MSG22 ;HELP
        CALL    GRAFX1
        LXI     D,MSG23 ;LIST GRAPHICS
        CALL    GRAFX1
        JMP     EGRAF

GRAFX1  CALL    PMSG
        MVI     B,33
        MVI     D,Ø5EH
GRAFX2  PUSH    B
        PUSH    D
        MOV     E,D
        MVI     C,CONOUT
        CALL    BDOS
        MVI     E,Ø2ØH
        MVI     C,CONOUT
        CALL    BDOS
        POP     D
        INR     D
        POP     B
        DCR     B
        JNZ     GRAFX2
        CALL    CRLF
        RET

ERASR   MVI     E,ØFFH   ;REQ INPUT CHAR
        MVI     C,CONSIO
        CALL    BDOS     ;A=CHAR
        ORA     A
        JZ      ERASR    ;A=Ø - NO CHAR READY
ERA1    MVI     E,ØFFH   ;REQ INPUT CHAR
        MVI     C,CONSIO ;FIRST CHAR = ESC
        CALL    BDOS     ;GET NEXT
        ORA     A
        JZ      ERA1     ;A=Ø - NO CHAR READY
        CPI     Ø44H     ;LEFT ARROW
        JZ      ELNB
        CPI     Ø43H     ;RIGHT ARROW
        JZ      ELNE
        CPI     Ø42H     ;DOWN ARROW
        JZ      EPGE
        CPI     Ø41H     ;UP ARROW
        JZ      EPGB
        CPI     Ø48H     ;HOME
        JZ      EALL
        JMP     INPUT

BLUE    CALL    ESCN
        LDA     MARKFG   ;CHECK FLAG
        ORA     A
        JNZ     MARK2
        MVI     A,Ø31H
        STA     MARKFG
        LXI     H,MARKS
        MVI     B,4
        CALL    BL1
        CALL    ESCJ     ;SAVE CURSOR POSIT
        LXI     D,MSG3   ;DUMP SCREEN TO MEMORY
        CALL    MSG5Ø
        CALL    SAVE
        LXI     D,MSG51 ;MOVE TO MARK 2
        CALL    MSG5Ø
        CALL    ESCK
        MVI     E,Ø2AH   ;'*'
        MVI     C,CONOUT
        CALL    BDOS
        JMP     INPUT

MARK2   MVI     A,Ø32H
        STA     MARKFG
        LXI     H,MARKS+4
        MVI     B,4
        CALL    BL1
        MVI     E,Ø2AH   ;'*'
        MVI     C,CONOUT
        CALL    BDOS
        CALL    ESCJ
        LXI     D,MSG52
        CALL    MSG5Ø
        CALL    ESCK
        JMP     INPUT

RBLUE   CALL    DSPLY1   ;ZERO MARKS SET
        JMP     MKDELT

ESCJ                     ;SAVE CURSOR POSIT
        MVI     E,Ø1BH   ;ESCAPE
        MVI     C,CONOUT
        CALL    BDOS
        MVI     E,Ø6AH   ;'j'
        MVI     C,CONOUT
        CALL    BDOS
        RET

ESCK                    ;SET CURSOR TO SAVED POSIT
        MVI     E,Ø1BH  ;ESCAPE
        MVI     C,CONOUT
        CALL    BDOS
        MVI     E,Ø6BH  ;'k'
        MVI     C,CONOUT
        CALL    BDOS
        RET

ESCN                    ;CURSOR POSIT REPORT
        MVI     E,ESCAPE ;SEND ESC-n
        MVI     C,CONOUT
        CALL    BDOS
        MVI     E,Ø6EH  ;'n'
        MVI     C,CONOUT
        CALL    BDOS
        RET

BL1     PUSH    H
        PUSH    B
BL2     MVI     E,ØFFH  ;REQ INPUT CHAR
        MVI     C,CONSIO
        CALL    BDOS
        ORA     A       ;CHECK IF Ø
        JZ      BL2
        POP     B
        POP     H
        MOV     M,A
        INX     H
        DCR     B
        JNZ     BL1
        RET

RED     MVI     A,ØØØH  ;SET MOVFG TO MOVE
        STA     MOVFG
        JMP     BLOK

WHITE   MVI     A,ØØ1H  ;SET MOVFG TO COPY
        STA     MOVFG
BLOK    CALL    ESCJ    ;SAVE CURSOR POSIT
        CALL    ESCN    ;REPORT CURSOR POSIT
        LXI     H,MARKS+8
        MVI     B,4
        CALL    BL1
        CALL    STBLK   ;GET START BLOCK LOCATION

BLKCT   LDA     MARKFG  ;CHECK FLAG
        CPI     Ø32H
        JNZ     NEGERR  ;NO 2 MARKS
        ANA     A       ;CLEAR THE CARRY BIT
        LDA     MARKS+2 ;ROW OF MARKER1
        MOV     B,A
        LDA     MARKS+6 ;ROW OF MARKER2
        SBB     B
        JM      NEGERR  ;RESULT IS NEG
        INR     A
        STA     RCOUNT  ;ROW COUNT
        ANA     A
        LDA     MARKS+3 ;COLUMN OF MARKER1
        MOV     B,A
        LDA     MARKS+7 ;COLUMN OF MARKER2
        SBB     B
        JM      NEGERR  ;RESULT IS NEG
        INR     A
        STA     CCOUNT  ;COLUMN COUNT

        LDA     MOVFG   ;COPY?
        ORA     A
        CNZ     DSPLY1  ;YES - REDISPLAY SCREEN
        LXI     H,BUFR  ;FIND FIRST CHAR
        LDA     RSCNT   ;ROW COUNT TO 1ST CHAR
        DCR     A
        JZ      COLCNT
        MOV     C,A
ROWCT1  MVI     B,8Ø    ;CHAR COUNT/ROW
```

```
ROWCT2  MOV     A,M     ;GET CHAR
        CPI     ESCAPE
        CZ      ADDB
        INX     H
        DCR     B
        JNZ     ROWCT2
        DCR     C
        JNZ     ROWCT1
COLCNT  LDA     CSCNT   ;COL COUNT TO 1ST CHAR
        MOV     C,A
        MVI     B,8Ø    ;CHAR COUNT/ROW
COLCT1  DCR     C
        RZ
        MOV     A,M     ;GET CHAR
        CPI     ESCAPE
        CZ      COLCT2
        INX     H
        DCR     B
        JMP     COLCT1

COLCT2  PUSH    B
        INX     H
        PUSH    H
        MOV     E,A     ;E=ESCAPE
        MVI     C,CONOUT
        CALL    BDOS
        POP     H
        PUSH    H
        MOV     E,M     ;E=NEXT CHAR
        MVI     C,CONOUT
        CALL    BDOS
        POP     H
        POP     B
        INR     B
        INR     C
        RET

COLCT3  MVI     A,1     ;SET INSERT FLAG
        STA     INSFG
        LDA     MOVFG   ;COPY?
        ORA     A
        CZ      DBLKLN  ;NO - DELETE BLOCK LINE
        PUSH    H
        PUSH    B
        LXI     H,MARKS+8 ;CURSOR TO NEXT POSIT
        CALL    SCRNP   ;INSERT POSIT
        POP     B
        POP     H
        LDA     CCOUNT  ;SET COLUMN COUNT
        MOV     C,A

BLKCT1  LDA     INSFG
        ORA     A
        JNZ     INSCH   ;INSERT CHAR ROUTINE
        MOV     A,M     ;A=CHAR
        CPI     ESCAPE
        CZ      ADDB
        INX     H
        DCR     B       ;END OF ROW?
        JNZ     BLKCT1
        PUSH    H
        LXI     D,MSG2+15 ;EXIT RVID AND GRAPHICS
        CALL    PMSG
        POP     H
        LDA     RCOUNT  ;ROW COUNT
        DCR     A
        STA     RCOUNT  ;BLOCK DONE?
        JZ      MKDELT
        LDA     MARKS+1Ø
        INR     A       ;INR TO NEXT ROW
        STA     MARKS+1Ø
        LDA     MARKS+2
        INR     A       ;INR TO NEXT ROW
        STA     MARKS+2
        JMP     COLCNT

INSCH   MOV     A,M     ;CHAR=A
        PUSH    H
        PUSH    PSW
        PUSH    B
        MOV     E,A
        MVI     C,CONOUT
        CALL    BDOS
        POP     B
        POP     PSW
        CPI     ESCAPE
        CZ      ADDCB
        POP     H
        INX     H
        DCR     B
        DCR     C
        JNZ     BLKCT1
        CALL    TDLY
        MVI     A,Ø
        STA     INSFG    ;RESET INSERT FLAG
        JMP     BLKCT1

DBLKLN  PUSH    H
        PUSH    B
        LXI     H,MARKS
        CALL    SCRNP
        LDA     CCOUNT
        MOV     B,A
DCHAR   PUSH    B
        CALL    DELET
        POP     B
        DCR     B
        JNZ     DCHAR
        CALL    TDLY
        POP     B
        POP     H
        RET

STBLK   LDA     MARKS+2 ;ROW INFO OF MARKER1
        SUI     Ø1FH    ;SUBTRACT ORIGIN-1
        STA     RSCNT   ;COUNT = # OF ROWS
        LDA     MARKS+3 ;COLUMN INFO OF MARKER1
```

```
        SUI     Ø1FH      ;SUBTRACT ORIGIN-1
        STA     CSCNT     ;COUNT = # OF COLUMNS
        RET

ADDCB   INR     C
        INR     C
ADDB    INR     B
        INR     B
        RET

NEGERR  LXI     D,MSG53 ;NEGATIVE RESULT
        CALL    MSG5Ø
        CALL    DELAY
        LDA     MOVFG     ;CHECK MOVE FLAG
        ORA     A
        CZ      DSPLY1    ;DISPLAY SCREEN
MKDELT  MVI     A,ØØØH    ;ZERO MARKFG
        STA     MARKFG
        LXI     H,MARKS
        MVI     B,12
        MVI     A,ØØØH
MKCLR   MOV     M,A
        INX     H
        DCR     B
        JNZ     MKCLR
        JMP     EMSGD

SCRNP   MVI     B,4       ;SCREEN POSITION
MKD1    MOV     E,M
        PUSH    H
        PUSH    B
        MVI     C,CONOUT
        CALL    BDOS
        POP     B
        POP     H
        INX     H
        DCR     B
        JNZ     MKD1
        RET
```

Listing 2
Enhanced PAINT.ASM!!!

```
MSG1    DB      27,69,'Welcome to PAINT.',13,1Ø
  DB  'Paint a picture you desire using the arrows to move the cursor.',13,1Ø
  DB  'Special function keys are defined as:',13,1Ø
  DB  '    f1     - sets  REVERSE VIDEO',13,1Ø
  DB  '    f2     - exits REVERSE VIDEO',13,1Ø
  DB  '    f3     - sets  GRAPHICS CHARACTERS',13,1Ø
  DB  '    f4     - exits GRAPHICS CHARACTERS',13,1Ø
  DB  '    f5     - changes PRINTER FONT (NOT H-14)',13,1Ø
  DB  '    ERASE & ARROW or HOME key - ERASES portions of screen.',13,1Ø
  DB  '    BLUE   - sets BLOCK MARKER location.  CTRL-B will reset.',13,1Ø
  DB  '    RED    - MOVES BLOCK to new cursor location.',13,1Ø
  DB  '    WHITE - COPIES BLOCK to new cursor location.',13,1Ø
  DB  '    ESC-E - ERASES entire screen (EVERYTHING IS LOST!)',13,1Ø,1Ø
  DB  '    CTRL-G  - DISPLAYS GRAPHICS on top 2 lines.',13,1Ø
  DB  '    CTRL-D  - DUMP  the painting.',13,1Ø
  DB  '    CTRL-L  - LOAD  the painting.',13,1Ø
  DB  '    CTRL-P  - PRINT to printer.',13,1Ø
  DB  '    CTRL-W  - WRITE to disk.',13,1Ø
  DB  '    CTRL-R  - READ  from disk.',13,1Ø
  DB  '    CTRL-E  - ENDS  the program.',13,1Ø
  DB  'When ready to start, hit "ERASE""HOME" to clear the screen.'
MSG  DB  27,121,49,27,120,49,27,121,53,27,89,56,37
  DB  27,'pRVID',27,'q    ',27,'pNVID',27,'q    ',27,'pGRAF',27,'q    '
  DB  27,'pNORM',27,'q    ',27,'pFONT',27,'q    ',27,'pERASE',27,'q    '
  DB  27,'pMARK',27,'q    ',27,'pMOVE',27,'q    ',27,'pCOPY',27,'q'
  DB  27,89,32,32,27,120,54,'$'
MSG2    DB  27,121,49,27,120,49,27,89,56,43,27,120,53,27,79,27,113,27,71,'$'
MSG3    DB  'DUMPING SCREEN TO MEMORY.',27,35,'$'
MSG4    DB  'RECALLING SCREEN FROM MEMORY.',27,89,32,32,'$'
MSG8    DB  27,121,54
MSG9    DB  27,79,27,89,32,32,27,121,49,27,121,53,27,69,'$'
MSG16   DB  '  TYPE ANY KEY:$'
MSG21   DB  27,69,'$'
MSG22   DB  27,72,'Normal:    $'
MSG23   DB  'Graphics: ',27,7Ø,'$'
MSG51   DB  'MOVE CURSOR TO SECOND MARK.  CTRL-B WILL RESET.',27,121,53,'$'
MSG52   DB  'MOVE CURSOR TO NEW LOCATION.  CTRL-B WILL RESET.',27,121,53,'$'
MSG53   DB  13,1Ø,'BLOCK INCORRECTLY DEFINED.$'
```

*

Something COMmon About MS-DOS And CP/M

Bill Wilkinson
Heath Company

Here's an interesting programming technique I've discovered. I doubt that I was the first one to discover it, but I haven't seen it on any of the bulletin boards nor in any of the computer magazines. Basically, the technique lets you write COM-type programs that will run unmodified under both MS-DOS and CP/M.

The program can do this because of two facts: First, COM files in both CP/M and MS-DOS start at address 100 hex. Second, there exists certain bit patterns common to both the 8080 and 8088 instruction set that cause both CPUs to behave predictably. All that's necessary is that this bit pattern be placed at the start of the program. As a specific example, consider the following binary sequence.

```
11001101 00010001 00100100 11111111
```

For the 8088, this pattern translates to the following opcodes:

```
INT        11H
AND        AL,0FFH
```

For the 8080, the first three bytes translate to:

```
CALL       2411H
```

and the fourth byte is ignored.

See where this is heading? The 8088 instructions tell MS-DOS to return the I/O configuration and AND it with 0FFH, which, of course, does nothing to the returned data. Your MS-DOS program can immediately follow. Depending on what it does, you may or may not want to use the data returned by the interrupt instruction. The 8080 sees the first three bytes as an unconditional call to address 2411 hex, so that's where the CP/M code should start. This gives about 8K of memory for MS-DOS and whatever length is necessary for CP/M. Of course, if you need more memory for your MS-DOS code, you can always jump around the CP/M portion.

To use this technique requires two computers, one that runs CP/M and a PC machine that runs MS-DOS. (Note that this program may not work on the 8088-side of the original H/Z-100 series. Unless I overlooked it while reading version 2 of the Programmer's Utility Pack, the H/Z-100 does not recognize INT 11H.) The machines I developed and tested this program on were an H8/H19 and an HF-241. I transferred data between the two machines by hooking up a cable between the two serial ports and using XMODEM. I used CPS on the HF-241 side and a public domain version of Christensen's MODEM7 program on the H-8.

I was tempted to follow up on that INT 11H instruction and make the program display a large number of facts about the operating system it was running under, but decided to keep it simple. So the example program, DOSINFO.COM, will only print a sign-on message on the screen and then state whether it's running under MS-DOS or CP/M.

DOSINFO.COM must be built out of two binary files called DCPM .BIN and DMSDOS.BIN. These in turn are assembled from their associated ASM files shown in listings one and two. In the CP/M MASM Listing 1, the "call start" instruction is there for testing the program before transferring it to MS-DOS. However, it doesn't make any difference whether or not that command is there when you merge the program with DMSDOS.BIN. The remainder of the code begins at 2411 hex as defined by the "org 2411h" statement. When "start" is called, the stack pointer puts the return address in the stack. Since the CP/M program will never return to "main", the "pop h" instruction removes this address from the stack. The rest of the program is the standard CP/M procedure for printing text to the display. The DE register points to the start of the text while the C register has the "print text" command. Calling BDOS performs the action, then "jmp 0" causes the program to exit to CP/M.

The source code for DMSDOS (Listing 2) is similar to that in DCPM. After performing the INT 11H/AND 0FFH code (which is ignored), the program points to the text with the DX register, loads the "print text" command into the AH register, then prints it by performing an interrupt 21H. The program exits to MS-DOS when it encounters "int 20h".

Listing 1

```
;
;       DCPM.ASM
;
;       Disk Operating System Information Version 1.Ø
;       William A. Wilkinson 1Ø-Nov-86
;
;       Assemble under CP/M then transfer the binary code
;       to MS-DOS.
;
cr      equ     Ødh             ; Carriage return.
lf      equ     Øah             ; Line feed.
tab     equ     9               ; Tab.
bdos    equ     5               ; Function entry point.

        org     1ØØh
;
;       MS-DOS thinks that the following instruction is
;       a "get i/o configuration" command (INT 11H).
main:
        call    start

        org     2411h           ; CP/M program starts here.

start:
        pop     h             ; Clean up.  We're not returning.
        lxi     d,cptxt         ; Point to the CP/M message.
        mvi     c,9             ; Print string function.
        call    bdos
        jmp     Ø               ; Exit.

cptxt:  db      cr,lf,lf
        db      tab,tab,tab
        db      'DOS Information Version 1.Ø',cr,lf
        db      tab,tab,tab
        db      '   William A. Wilkinson',cr,lf
        db      tab,tab,tab
        db      '          1Ø-Nov-86',cr,lf,lf
        db      tab,tab
        db      '    This program is running under CP/M.'
        db      cr,lf,lf
        db      '$'

        end
```

To write the program, use your favorite text editor and enter Listing 1 on the CP/M machine. Assuming that A: is your default drive and all your programs are there, assemble DCPM by typing:

```
ASM DCPM.AAZ
LOAD CPM
```

Transfer the resulting COM file to MS-DOS with XMODEM (or other error-checking protocol) and name it DCPM.BIN.

On the MS-DOS side, type in the code in Listing 2 and assemble it as follows:

```
MASM DMSDOS;
LINK DMSDOS;
EXE2BIN DMSDOS
```

MASM is part of the Programmer's Utility Pack, while LINK and EXE2BIN come with MS-DOS. Note that without the semicolons at the end of the first two commands, MASM and LINK will stop to ask you a bunch of unnecessary questions. EXE2BIN will automatically name the assembled program to DMSDOS.BIN.

Use DEBUG to merge DCPM.BIN and DMSDOS.BIN. The following steps show how. (Do not enter the comments in parenthesis. That's just for your information.)

`DEBUG DMSDOS.BIN`	(Load the MS-DOS code first.)
`R`	(Note the value in the CX register pair. It should be 93 for this program.)
`M1ØØ L 93 8ØØØ`	(Move the program to 8000H. If CX was a different number in the previous step, use it instead of 93.)
`NDCPM.BIN`	(Load the CP/M file.)
`L`	
`M8ØØØ L 93 1ØØ`	(Move the MS-DOS program back to the beginning.)
`NDOSINFO.COM`	(Write the merged programs to DOSINFO.COM. It's not necessary to calculate the length, since the CX register was set up properly when DCPM.BIN was loaded.)
`W`	
`Q`	(Exit to MS-DOS.)

You should now have a runable copy of DOSINFO.COM on your disk.

Listing 2

```
;
;       DMSDOS.ASM
;
;       Disk Operating System Information Version 1.Ø
;       William A. Wilkinson 1Ø-Nov-86
;
;       MS-DOS code.
;
cr      equ     Ødh                     ; Carriage return.
lf      equ     Øah                     ; Line feed.
tab     equ     9                       ; Tab.

dosinfo segment

        org     1ØØh

        assume  cs:dosinfo,ds:dosinfo

main    proc    near

;       CP/M thinks that the following 1-1/2 instructions
;       form an unconditional call to 2411H.

start:
        int     11h                     ; Get the I/O configuration.
        and     al,Øffh                 ; Do nothing with it.
        mov     dx,offset mstxt         ; Tell 'em MS-DOS.
        mov     ah,9
        int     21h
        int     2Øh                     ; Goodbye.

mstxt   db      cr,lf,lf
        db      tab,tab,tab
        db      'DOS Information Version 1.Ø',cr,lf
        db      tab,tab,tab
        db      '   William A. Wilkinson',cr,lf
        db      tab,tab,tab
        db      '          1Ø-Nov-86',cr,lf,lf
        db      tab,tab
        db      '    This program is running under MS-DOS.'
        db      cr,lf
        db      '$'

main    endp                            ; End of main procedure

dosinfo ends                            ; Close the segment.

        end     main                    ; Program starts at main.
```

Instead of DEBUG, you can use the C program shown in listing three. I wrote this to avoid continually repeating the above procedure while developing DOSINFO. The code is heavily commented, so if you don't have a C compiler, you shouldn't have too much problem translating it to your preferred language. Basically, DMERGE will copy DMSDOS.BIN to DOSINFO.COM and write zeros in the area between the end of the MS-DOS code and the start of the CP/M code. Next, it opens DCPM.BIN, jumps to the start of the code at 2411 hex, and copies it to DOSINFO.COM. DMERGE closes all open files, exits to MS-DOS, and DOSINFO is ready to run.

Once you're satisfied that it runs properly on MS-DOS, transfer it to your CP/M system and test it there. This time, it will tell you that it's running under CP/M.

As you see the program run, you may be thinking, "it's clever, but what use is it?"

Beats me.

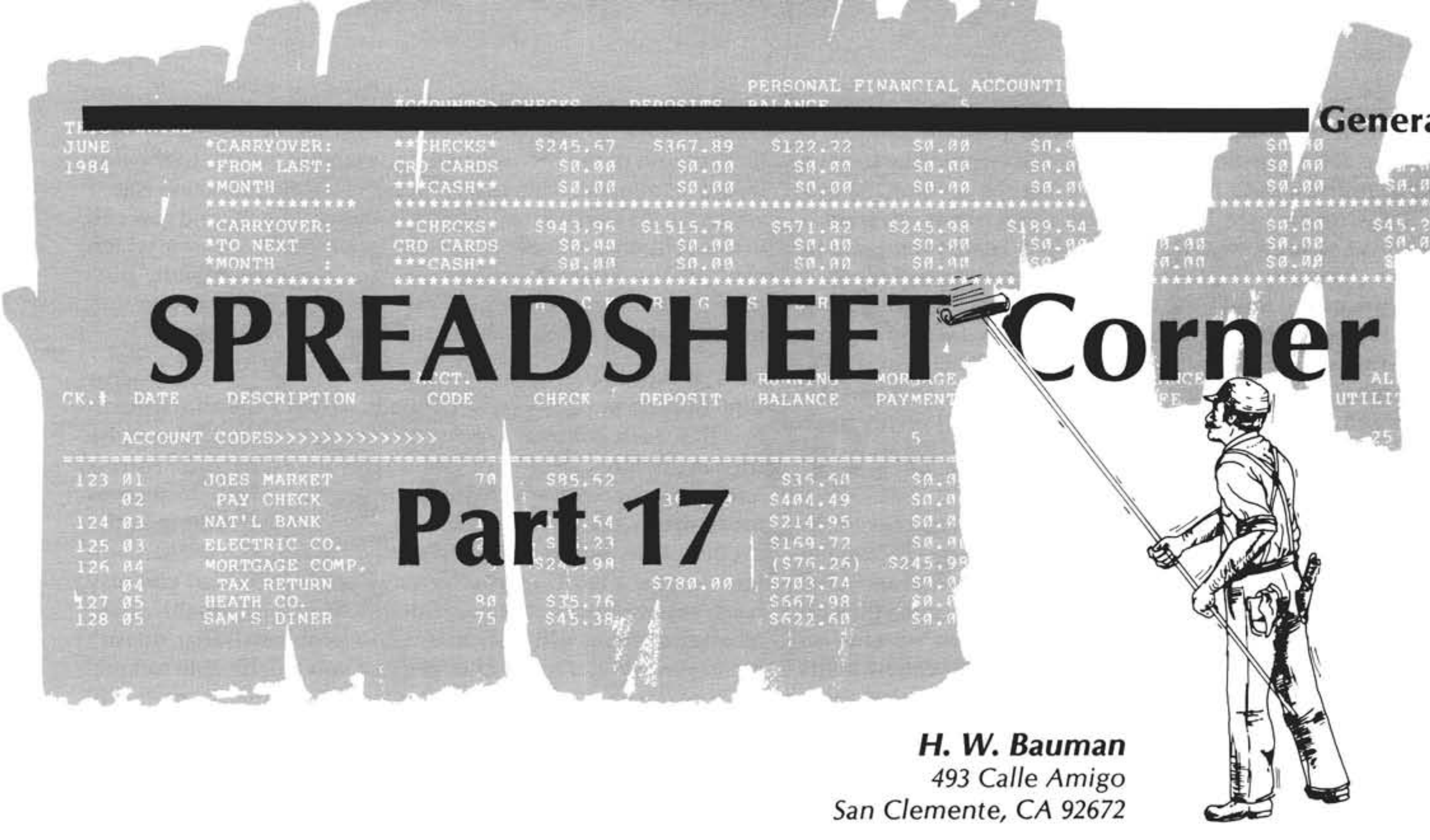
I got a kick out of the concept, but can't think of any way to get rich off it. It lets you write programs that are transportable between the two disk operating systems, but you have to write twice as much code. In such a case, writing your program in a high-level language and compiling it twice is more practical. The only useful thing I can think of is as a protection device for public domain and shareware programs. For example, somebody downloads CPMCHESS.COM from the local bulletin board and tries to run it on an MS-DOS machine. The program states "Sorry, I'm not an MS-DOS program" then performs an orderly exit.

Other than that, the DOSINFO technique may either be totally useless or is a solution looking for a problem. Any ideas?

Listing 3

```
/*
        DMERGE.C Ver.1.Ø
        Written in Toolworks C by William A. Wilkinson,
        15 November 1986.

        This program merges the two binary files used in the
        DOSINFO MSDOS-CP/M demonstration program.  The MS-DOS
        file should be named DMSDOS.BIN and the CP/M filename
        should be DCPM.BIN.  DMERGE will first transfer
        DMSDOS.BIN to DOSINFO.COM.  Next it will pad the
        remainder of DOSINFO.COM with nulls up to address
        Øx231Ø.  DMERGE will then read DCPM.BIN, skipping the
        first Øx231Ø bytes.  At byte number Øx2311 (Øx2411 with
        the Øx1ØØ offset), it will transfer DCPM.BIN to
        DOSINFO.COM.
*/

#include <stdio.h>
#include <math.h>           /* Comment out if not using Toolworks C or */
                            /*  don't have the Mathpak option. */

main()
{
    FILE        *fpØ, *fp1, *o_file();
    int         c, i;
    long        cpmstart; /* Change to int if using Toolworks C */
                          /*  without the Mathpak. */

    cpmstart = Øx2411 - Øx1ØØ;          /* Start of CP/M code. */

    fputs("DMERGE Ver.1.Ø by Bill Wilkinson 15 November 1986\n", stdout);

    fpØ = o_file("dosinfo.com", "wb");  /* Create DOSINFO.COM. */
    fp1 = o_file("dmsdos.bin", "rb");   /* Open the MS-DOS binary file. */

    fputs("\nTransferring DMSDOS.BIN to DOSINFO.COM...", stdout);

    i = Øx1ØØ;  /* Initialize counter to standard *.COM file offset. */

    while ((c = fgetc(fp1)) != EOF) {   /* Transfer DMSDOS.BIN */
        fputc(c, fpØ);                  /*  to DOSINFO.COM. */
        i++;                            /* Point to the next empty byte.*/
    }

    fputs("\nPadding with nulls...", stdout);
    while (i < Øx2411) {                /* Pad with nulls. */
        fputc(NULL, fpØ);
        i++;
    }

    c_file(fp1);                        /* Close DMSDOS.BIN. */
    fp1 = o_file("dcpm.bin", "rb");     /* Now open CP/M binary file. */

    fputs("\nSkipping Øx231Ø bytes in DCPM.BIN...", stdout);

    fseek(fp1, cpmstart, SEEK_SET);

    fputs("\nTransferring DCPM.BIN to DOSINFO.COM...", stdout);

    while ((c = fgetc(fp1)) != EOF)     /* Transfer rest to DOSINFO.COM.*/
        fputc(c, fpØ);

    c_file(fp1);                        /* Done! Close the files. */
    c_file(fpØ);
    fputs("\n\nDone!\n\n", stdout);
    exit(Ø);                                    /* Return to MS-DOS. */
}

/*
        Open the file (fn) for read or write (type).  Return
        the file pointer (fp).  Exit to MS-DOS if can't open.
*/

FILE *o_file(fn, type)
char *fn, *type;
{
    FILE        *fp, *fopen();

    if ((fp = fopen(fn, type)) == NULL) {
        fputs("\nCannot open file!\n", stderr);
        exit(1);
    }
    return(fp);
}


/*
        Close the file pointed to by the file pointer (fp).
        Do not return anything.  Exit to MS-DOS if can't
        close.
*/

c_file(fp)
FILE *fp;
{
    int         c;

    if ((c = fclose(fp)) != NULL) {
        fputs("\nCannot close file!\n", stderr);
        exit(1);
    }
}
```

*



SPREADSHEET Corner

Part 17

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

This article will complete our Database Sales Ranking Project. With this application we have and will use many macros and a User-defined menu, capabilities that LOTUS 1-2-3 provides that prove invaluable. Unfortunately, the LOTUS Manual hardly covers the many uses of macros and menus. Fortunately, the LOTUS 1-2-3 Users have worked with the many undocumented POWER commands. These commands and features put the power into the hands of Users that are willing to put the little extra effort into their worksheet to make them professional and user-friendly. Also, they are good time savers and add to a program's useful life by allowing the worksheets to be used by non-technical personnel. This all balances the scale of the added work with the useful output! That has been the goal of this Project! The example program that I have used to demonstrate this type of worksheet would not be a good one for a useful application, but it serves as a learning experience for the readers that want to learn Advanced LOTUS 1-2-3.

The questionnaire that I had in a "SPREADSHEET Corner" has supplied me with valuable knowledge about the readers that are using the articles. I know from the response that the readers do not want long programs that require a lot of typing to construct the worksheet. An example of that was the Income Tax articles. I was sure that the readers would like that subject, but the response was: Will you sell us your program? I am not writing these articles to sell programs. I WANT to benefit by knowing that I am providing a tutorial service so that the reader will be able to get more valuable service from their computer and software. The majority of the responses showed that the LOTUS 1-2-3 software was widely used and that these readers wanted to have articles that would provide them with a challenge! I hope that this article will start that.

I also realize that the replies covered other spreadsheet software and not all readers have experience. Therefore, I will have to provide a balance of articles that are for beginners and will work with other software. I have laid out a sequence of articles with a pattern of advanced projects, followed by beginning articles that will work with most software, and follow these with articles that will demonstrate how to combine the benefits of spreadsheet and dBASE programs (the response showed that most readers have dBASEII, if they have one.) I want this sequence to bring ALL readers useful material that they can adapt to their individual needs and to close the gap between the novice and the experienced users by these articles serving as learning experiences. I will supply more details about the advanced project with step-by-step instructions to keep them within the scope of the less experienced reader.

I am sure that the final aim of the readers is to prepare worksheets in a timesaving and professional way! As always, I request your INPUT by letter. If you do not agree with my selections, let me know. If I am not making the projects easy enough, I want to know that and if I am putting too much detail in the articles I want to hear about it. This is your REMark and the material that the writers are supplying MUST fit the needs of as many readers as possible. I will admit that I cannot think of a way to work out how to fill the needs of the new readers that keep joining our ranks. How can they be brought up to where we are without being redundant? If any reader has an idea, I would like to hear about it. One thing I want to stress, if you ask a question or write a letter that requests a reply; you MUST enclose a stamped, self-addressed envelope (business-size). I will not be able to provide a reply without this. I get dozens of letters and I cannot cover this cost. My time comes FREE! I usually get a reply out within a week.

Also, you readers using Multiplan most likely know that a new Revision 2 is now out. I understand that it will provide many of the LOTUS features. I do not have a copy. This is also a problem for my budget. I cannot buy all the different software and their revisions. If the Vendors of this software will supply me with a copy, I will be most happy to test and write a Review about it. If any readers are using the Multiplan Rev 2.0, I would like to get a report from them as to how well they like the revision. At this time, if any readers are buying a new spreadsheet program, I would HAVE to suggest LOTUS 1-2-3 which I paid for myself! In the coming months, it looks like 1-2-3 will have many clones out at lower prices and how good they will be, only time and testing will tell. If you do not have a database program, I would suggest Ashton-Tate's dBASEII for the H/Z-100 and dBASEIII for the PC compatible computers. Again, I have purchased my own dBASEII Rev.2.43 and I have done some testing of dBASEIII on my son's IBM-PC and I was very impressed. I guess it is time that I get to the current subject — OUR PROJECT.

At this time, all readers should have a collection of macros working with the Database Sales Ranking worksheet. To execute the macros, the User MUST know which macro does what and what its name is! Macro names are not descriptive as to what they do and if your memory is as bad as mine where I am working on many different things at about the same time, this can be a BIG problem! One purpose of this Project is to provide a worksheet that can be turned over to someone who would not know any details about the Project and maybe even less about spreadsheets and computers! When this person turns on the computer they would not know anything about the macros, database, or range names. We must make the program display in plain English a choice of what procedures will perform the functions that this Database Sales Ranking System can do! The answer to this is the NEED to create a menu macro! Did I hear anyone say what it that? It sounds hard, but it will not be all that difficult if we take it in nice easy small steps!

LOTUS 1-2-3 has the /XM command that will let us create a User-defined menu! That sure sounds like something we can use. This command will let the USER choose the macro names and assign these names to our existing macros. 1-2-3 provides for up to eight commands in one menu. Our Project will fit that requirement. This User-Defined menu will look and work like the 1-2-3 menus that you have all been using. How many readers have ever wondered how 1-2-3 provided these menus? Well, you will soon know. The User's Menu will display the command names on the second line of the upper Control Panel and a Descriptive Phrase (designed by the User) explaining the purpose of the chosen command displayed on the third line. Stay with me and this will all keep getting clearer as we proceed. The menu will highlight the command that you choose by moving the highlight with the cursor movement keys from one command to the other.

Figure 1

```
19-Nov-85          (SALES IN THOUSANDS OF DOLLARS)          |
                                                            |
                   SPREADSHEET Corner COMPANY               |
                   *OFFICE SALES RANKING PLAN*              |
                                                       YTD  |
OFFICE      NAME      1STQTR 2NDQTR 3RDQTR 4THQTR YTD RANK|
----------------------------------------------------------|
ATLANTA     ABBOTT,T.  178    188                 366    1 |
BOSTON      WILSON,E.  188    173                 361    2 |
LOS ANGELES RICE,L.    195    161                 356    3 |
PORTLAND    NEWMAN,D.  169    178                 347    4 |
OMAHA       JOHNSON,E. 178    163                 341    5 |
CHICAGO     MILNER,H.  162    171                 333    6 |
```

Figures 1, 2, & 3 show how we located the macros on our worksheet and they MUST be LOCATED this way. We must include all the macros in a macro range! This Macro Range includes one column for each macro with at least 3 rows for the macro. The first column of the menu range, for example, relates to the first macro command. This macro uppermost cell contains Name which MUST start with an uppercase letter. The second row cell in this column is a label containing the Descriptive Phrase which explains the purpose of the command for User-friendly purposes! The third row cell and any other cells under it store the macro that will execute when the command is chosen.

Now, it should be getting clearer as to why we put the macros side-by-side in consecutive columns — why they all started in row 6 of each column. I was providing for these macros to be used in this menu. Again, this shows why the Preparation Form is necessary when you are creating a project worksheet. It is easy to move things on a small worksheet and keep everything in the proper place, but as the complexity increases, the preparation form start is a must. As you try some application that you may have, you will soon learn the HARD WAY that to skip this phase of programming will only cause you problems! That doesn't mean you will not make changes, but you will be able to keep track of the changes easier. I want you to see this for yourself.

Row 4 is used for the command name and Row 5 will contain the descriptive phrase. If you will check with your computer, you will be able to prove why you have a limit of eight commands per menu. Luckily, we will have a place for future expansion. This could be important for the user that will be adapting this procedure to his NEED!

With the Menu Range designed, the User Must tell 1-2-3 that a menu exists and where the commands will be located. REMEMBER, 1-2-3 Ranges MUST be named. We will do this step down a few paragraphs. Let's list our macros, what column they will be in, and what descriptive phrase we will use:

\MACRO	COLUMN	DESCRIPTIVE PHRASE
\A	J	Rank Offices by year-to-date sales
\B	K	Rank Offices by quarter's sales
\C	L	Sort Database alphabetically by Name
\D	M	Sort Database alphabetically by Office & Name
\E	N	Send Data Print file to Printer
--	0	Exit from Database program (Will be added later)

Now, we can clear up one other question many readers may have had! I choose macro names (command names) for this menu. We must make sure that no two or more commands begin with the same first uppercase letter. This avoids ambiguity when a command is chosen by typing its first letter. If you did have more than one command using the same first letter, 1-2-3 would always execute the first command it gets to with the given letter and this could be the wrong one! I will list the names for the commands that we will have in the menu:

MACRO	COMMAND NAME
\A	YTD
\B	QTR
\C	NAME
\D	OFFICE
\E	PRINT
--	EXIT (Will be added later)

With the above defined information, we can create the menu:

1. Use the command Names to complete the Row 4 for the Menu Range. For example, see the label YTD in cell K4 and QTR in cell L4.
2. Use the list of Descriptive Phrases for entry into Row 5. Use capital letters sparingly to make the phrase look professional. If you can think of better descriptions use yours. An example, "Rank Offices by Quarter Sales" in cell K5.

Figure 2

```
OFFICE                                        PRINT                                 EXIT
Sort Database alphabetically by Office & Name Send Data Print File to printer       Exit from Database program
/DSPOFFICE~A~SNAME~A~G                        {Home}/PPCR                           /REJ1~/XQ
/XG\M~                                        R.{End}{Down}{End}{Right}{End}{Down}~
                                              OML1Ø~MR69~QAGPQ
                                              /XG\M~
```

Figure 2 shows the menu macro called \M in cell J2. This macro uses a new /XM command. The syntax is /XMlocation~, where location is the top-left corner of the Menu Range — in this case, cell K4. LOTUS 1-2-3 requires only the location of the upper-left corner for a range and it knows where the rest of the menu is located, because by syntax that the macros forming the menu must be next to each other. In place of cell J3 in the /XM command, I named cell J2 with MENU1. Using a range name handles any case where MENU1 should get relocated from cell J2. We have discussed this before. Here is the procedure:

1. Move the cursor to cell J2, type /RNCMENU1 and press Return.

2. Move cursor to cell J2, type '/XMMENU1, press Return, type '/RNC\M, and press Return twice.

Are you watching your column widths? Test this macro. Hold down the ALT key and press the M key. As soon as you get the M key pressed, the User-defined menu should appear on the second line of the Upper Control Panel as follows:

```
    YTD   QTR   NAME   OFFICE   PRINT
```

The YTD should be highlighted and the Mode indicator at the top right should read — CMD MENU — because 1-2-3 is now executing the menu under the control of the \M menu macro. The bottom Control Panel line should display — Rank Office by Year-to-Date Sales — which we stored in the cell J4. Do you find this is true with your work? Move the highlight pointer to the right and the next command — QTR — appears and the bottom line will display — Rank Office by Quarter Sales. When the User presses the Return key, the program executes the macro that pertains to the highlighted command. (Typing the first capital letter of the menu command will work the same way.) This menu works the same way as your 1-2-3 menus have worked for you.

Let's try the menu again. Select QTR by pressing the Q key, respond by typing 1, and press Return. The Result should display the first-quarter sales in descending order. Do you find this to be true? The menu will disappear when the macro has completed its execution and you will find the READY Mode indicator will be back. If we want to do another Sort, the \M menu macro must be started again. I think that this could and should be improved on! We will solve this with a /XG command added to the bottom cell of EACH MACRO!

1. In each cell J8, K10, L6, M6, and N7, type '/XG\M~.

Now, as the selected macro completes its execution, the above /XG command will return the program to the menu. The term for this procedure is usually called — USER INTERFACE!

Another problem comes up! What if we start executing a selected macro, how would you ABORT the procedure and/or the menu? There are three ways that I can think of right now. You may think of some others. First, you could press the ESC key. Second, hold the CTRL key down and press the Scroll Lck key. Third, use the 1-2-3 method of adding an EXIT command to the menu! This is the one that I like. The EXIT command is added to the Menu Range and we will proceed as follows:

1. Move the cursor to cell O3 which is adjacent to the PRINT command. Type the label — EXIT — into cell O3.
2. Move the cursor down to cell O4 with the Down Arrow key and type — Exit from the Database Ranking program — which will be the Description Phrase.
3. Move the cursor with the Down Arrow key to cell O5, type '/XQ, and press Return.
4. Good programming dictates that before leaving a program always clean up the program. In our worksheet, we have data in a cell — the number we inputted with the QTR prompt. Thus, we should add to our step 3 above a command to Range Erase this cell. Therefore, replace the '/XQ with '/REJ1~/XQ.

The /XQ command tells a macro to quit execution! In other words, it stops the execution of the current macro. In our case, the \M menu macro would be stopped and our program would return to the READY Mode! This /XQ command is very useful. We will find many places in the future where it will help us. Let's test this by invoking the \M menu macro and select Exit command by highlighting the command. **Note!** Typing the letter E will not work and will get us into trouble, because it is the second E first letter in the menu. The 1-2-3 program will think that we have selected the first E, which is PRINT. This is NOT all bad, because it gives us a degree of protection from leaving the menu by error when we really did not want to leave the menu.

This Project has automated five tasks in our Ranking Sales System by incorporating the tasks into macros. Then, we made the macros easier to use by integrating them into a menu. We have a Project that will execute any of the tasks by having the User hold down the ALT key and pressing the M key. The Project will nearly run itself. BUT, WE ARE NOT DONE! How would you add the third-quarter results or fourth-quarter results to our worksheet? If you answered, add another macro to the menu — YOU ARE RIGHT! I am going to try something new to "SPREADSHEET Corner"! I know that there are a lot of smart, experienced readers out in REMark land. So, I am going to request that the readers take on the task af solving the problem! I would like to request the readers that can solve the problem to send me their solution by sending their worksheet. I will put a report of the results in a future article after a reasonable amount of time has passed for busy readers to work on the project. Here is a chance to see what YOU CAN DO by yourselves and I will not embarrass anyone by giving names if they make a mistake!

Back to our Project. I am sure that the readers will agree that this type of worksheet can be used by a nontechnical operator with very little training. The menu system makes the Project easier and faster for both trained and untrained persons. It makes it possible to come back many months later and still be able to use the program without retraining. It eliminates the need to train persons to use the more complex non-everyday commands; such as, Data Sort, Data Fill, etc.

The macros that we are using for this project are not very sophisticated when you stop and think of what could be done. The User can create macros as intricate as the imagination permits; BUT one has to balance the effort to write the macros vs how often the program will be used and how many different persons may be using the program. This simple worksheet is not a good example from that standpoint, BUT it is designed as a learning experience that the readers can learn and then adapt to their needs. There are still a lot of macros and menus that we will explain in future Projects. I will be getting into really advanced 1-2-3 techniques as we go.

Figure 3

```
OFFICE                                         PRINT                                   EXIT
Sort Database alphabetically by Office & Name Send Data Print File to printer         Exit from Database program
/DSPOFFICE~A~SNAME~A~G                         {Home}/PPCR                             /REJ1~/XQ
/XG\M~                                         R.{End}{Down}{End}{Right}{End}{Down}~
                                               OML1Ø~MR69~QAGPQ
                                               /XG\M~
```

I am going to add a few more ideas to this Project. How many readers have thought of some? How many readers thought we could and should use Cell Protection? This worksheet is a natural for cell protection! When 1-2-3 is first booted up and loaded, each cell is invisibly surrounded by what the LOTUS 1-2-3 Manual calls an "electronic fence" whose power is initially turned off. If this fence were turned on, it would protect every cell's contents from alterations. The contents of a protected cell cannot be replaced or edited! I am sure that the reader can see how this safeguarded data and formulas would be advantageous. A careless or inexperienced User could ruin the worksheet. Obviously, it would not be practical to protect every cell because this would prevent the worksheet from performing a useful task. However, once you have a worksheet operating correctly, it would be wise to protect the cells that would not need to be changed during the use of the worksheet. To protect this Project worksheet, type

```
/Worksheet Global Protection Enabled
```

This command is done from the Ready Mode. It is a "toggle" command that can be Enabled or Disabled as desired. Test this command by enabling a cell and see if you can change that cell or edit that cell. Then, change it back to Disabled. There is such a thing as over-protection! There is no benefit in protecting the entire worksheet. Protection can offer advantage only because it can coexist with nonprotection. This means that a portion of the worksheet, in ranges, can be protected and that the rest of the ranges would be unprotected! The procedure works like this:

1. Move the cursor to the upper-left corner of the Range you do NOT want Protected. Type /Range Unprotect. At this point, the Range can be expanded by "painting" the Range or by specifying the corners of the Range. If you will look at Figure 4, it will show with a U the cells that are unprotected. I will leave you to check Figure 4 and unprotect the cells that are shown with the U (Obviously, the cells that have or will get future data.).

Notice that the Range Unprotect command will take effect even while the Global Protection is Enabled. Therefore, if you want to change a protected cell at any time while protection is enabled for that cell, there is no need to use Worksheet Global Protection Disable — instead use Range Unprotect for that cell — then make the change and do a Range Protect to get back to where you started. Protection is ENTIRELY under your control — IT IS NOT PERMANENT!

Unless you are using LOTUS 1-2-3 Revision 1.0 (not 1A), you can add a special file name to this Project. If you SAVE the worksheet using the AUTO123 file name, then 1-2-3 will automatically load this worksheet whenever the computer and 1-2-3 is booted and loaded, the worksheet we saved as AUTO123 will automatically load into memory. There is one special requirement — the disk containing the AUTO123 must be the default data disk (usually B:). With AUTO123 you do not need to do the /File Retrieve command to load the worksheet. This is the procedure if the worksheet is in current memory:

1. Type /File Save AUTO123 and press Return.
2. Type /Quit to exit from 1-2-3 to the LOTUS Access Menu.
3. To automatically reload the worksheet, select the 1-2-3 Option from the LOTUS Access Menu.
4. LOTUS 1-2-3 will automatically load the AUTO123 file worksheet and display it.

There is one more step further that we should take to automate the Project! LOTUS 1-2-3 has a feature called Automatic Macro Execution! This feature allows the user to create another macro that will automatically execute the Menu Macro when the AUTO123 file is loaded. This macro is created the same way as we have done many times in this Project. The only difference is the macro's name. The macro that you want to execute automatically MUST have the name ALT-O (\O)! If you have added the AUTO123 special worksheet filename, you can also execute the \M Menu Macro by giving this macro a second name \O(not zero), as well as \M!

1. Move the cursor to the \M Range Name, type /RNC\O and press Return Twice.

The AUTO123 file and the \O macro features have now created a completely self-contained Project for the Rank Sort Sales file. I strongly advise you to use a freshly formatted disk for this project! The hold down ALT key and press O key will not work from the keyboard. This macro name is only for the automatic execution. You will notice that I did not eliminate the \M Menu Macro. You can have two names for a macro in 1-2-3 so this Menu Macro will have two names — \M and \O. When the AUTO123 and \O features are combined with a program using a menu and the /X commands, the automatic execution makes the Project extremely user-friendly.

Remember one thing about the automatic macro, it CANNOT be executed by using ALT-O from the keyboard! If you need to execute the Menu Macro from the keyboard the only way that will work is the ALT-M. The one Menu Macro has become, in effect, two — macro that executes automatically and one macro that is operated from the keyboard! Also, **Note!** There can only be ONE AUTO123 on a disk and there can only be one \O automatic macro per worksheet. Now you can see why I suggested that you put this Project on a new, separate disk! There is no reasonable limit as to how large this \O Menu Macro can be and it can have an almost unlimited number of steps and macros included. One thing to watch, eight macros per menu, but one menu can be used to call a second menu. You have seen LOTUS 1-2-3 do this many times.

Test your new disk with the Automatically Executed Program by booting up the computer with 1-2-3 included. Did your worksheet, named AUTO123, load? Did your Menu Macro, named \O, execute? How do you like what LOTUS has supplied us with — POWER! Use these features with some pet program that you use many times and you will go MUCH further with your many spreadsheet worksheets. By the way, do you know that you have just become a 1-2-3 PROGRAMMER! I have heard many times that some critics say we do not know anything about programming. I do not think that you readers have found it very hard after using my little worksheet. Have you?

We have not begun to use all of these powerful features, but we will get to them in future "SPREADSHEET Corner" articles. I do want to cover some projects for the readers that do not use LOTUS 1-2-3 and I want to cover some projects for dBASE because the combination of spreadsheets and databases go together like bread and butter! I think that the reader will agree that we have COME-A-LONG-WAY! Happy Spreadsheeting.

Figure 4

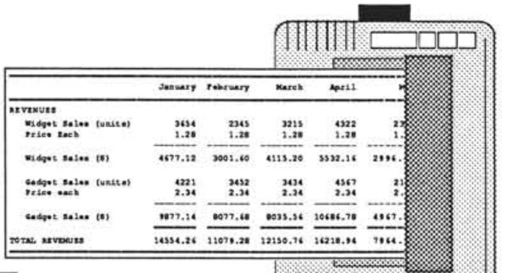
```
A1: (D1) U @TODAY
C1: '(SALES IN THOUSANDS OF DOLLARS)
I1: '¦
K1: '\M
I2: '¦
J2: '\0
K2: '/XMMENU1~
C3: 'SPREADSHEET Corner COMPANY
I3: '¦
J3: '
C4: '*OFFICE SALES RANKING PLAN*
I4: '¦
K4: 'YTD
L4: 'QTR
M4: 'NAME
N4: 'OFFICE
O4: 'PRINT
P4: 'EXIT
H5: ^YTD
I5: '¦
K5: 'Rank Offices by year-to-date sales
L5: 'Rank Offices by quarter's sales
M5: 'Sort Database alphabetically by Name
N5: 'Sort Database alphabetically by Office & Name
O5: 'Send Data Print File to printer
P5: 'Exit from Database program
A6: ^OFFICE
B6: ^NAME
C6: ^1STQTR
D6: ^2NDQTR
E6: ^3RDQTR
F6: ^4THQTR
G6: ^YTD
H6: ^RANK
I6: '¦
K6: '/DSPYTD~D~
L6: '/XNEnter Desired Quarter(1,2,3,or4):~QTRN~
M6: '/DSPNAME~A~G
N6: '/DSPOFFICE~A~SNAME~A~G
O6: '{Home}/PPCR
P6: '/REJ1~/XQ
A7: \-
B7: \-
C7: \-
D7: \-
E7: \-
F7: \-
G7: \-
H7: \-
I7: '¦
K7: 'SNAME~A~G
L7: '/XI(QTRN=1)~/DSP1STQTR~D~SNAME~A~G/XGCONTX~
M7: '/XG\M~
N7: '/XG\M~
O7: 'R.{End}{Down}{End}{Right}{End}{Down}~
A8: 'ATLANTA
B8: 'ABBOTT,T.
C8: U 178
D8: U 188
G8: U @SUM(F8..C8)
H8: U 1
I8: '¦
K8: '/DF~1~~~
L8: '/XI(QTRN=2)~/DSP2NDQTR~D~SNAME~A~G/XGCONTX~
O8: 'OML1Ø~MR69~QAGPQ
A9: 'BOSTON
B9: 'WILSON,E.
C9: U 188
D9: U 173
G9: U @SUM(F9..C9)
H9: U 2
I9: '¦
K9: '/XG\M~
L9: '/XI(QTRN=3)~/DSP3RDQTR~D~SNAME~A~G/XGCONTX~
O9: '/XG\M~
A1Ø: 'LOS ANGELES
B1Ø: 'RICE,L.
C1Ø: U 195
D1Ø: U 161
G1Ø: U @SUM(F1Ø..C1Ø)
H1Ø: U 3
I1Ø: '¦
L1Ø: '/XI(QTRN=4)~/DSP4THQTR~D~SNAME~A~G/XGCONTX~
A11: 'PORTLAND
B11: 'NEWMAN,D.
C11: U 169
D11: U 178
G11: U @SUM(F11..C11)
H11: U 4
I11: '¦
L11: '/XG\M~
A12: 'OMAHA
B12: 'JOHNSON,E.
C12: U 178
D12: U 163
G12: U @SUM(F12..C12)
H12: U 5
I12: '¦
A13: 'CHICAGO
B13: 'MILNER,H.
C13: U 162
D13: U 171
G13: U @SUM(F13..C13)
H13: U 6
I13: '¦
```

*

Now REMBRANDT brings great graphics to Heath CP/M users!

TWIST & SHOUT!

Sideways & Banner Printing Utility for Dot-Matrix Printers



```
                          January  February     March     April     M
REVENUES
  Widget Sales (units)       3654      2345      3215      4322     23
  Price Each                 1.28      1.28      1.28      1.28     1.
                          -------  --------  --------  --------  -----
  Widget Sales ($)        4677.12   3001.60   4115.20   5532.16  2996.

  Gadget Sales (units)       4221      3452      3434      4567     21
  Price each                 2.34      2.34      2.34      2.34     2.
                          -------  --------  --------  --------  -----
  Gadget Sales ($)        9877.14   8077.68   8035.56  10686.78  4967.
                          =======  ========  ========  ========  =====
TOTAL REVENUES           14554.26  11079.28  12150.76  16218.94  7964.
```

For *any* CP/M or MS/DOS computer (IBM compatibility is not required), just...

$34.95

***Special Offer:* one MS/DOS and one CP/M version for only...**

$49.95

TWIST & SHOUT! is two great pieces of software at one low price! TWIST lets you print out your spreadsheets (or any file) sideways on your dot-matrix printer. No more cutting and pasting to put together a fragmented printout. SHOUT prints giant banners in letters from two to eight inches high using three special type styles. Make banners, posters and signs with ease. Printers: Anadex, Apple, C. Itoh, Centronics, Data Products, DEC, Epson, Hewlett Packard, IBM, IDS, MPI, NEC, Mannesmann Tally, Okidata, Smith Corona, Toshiba and more! Write for latest list.

Solve Computer Incompatibility!

MEDIA MASTER lets your Z-100 computer or IBM compatible (PC, XT or AT) read, write and format up to 140 disk formats. Exchange information and programs with other computers for just **$39.95**.

MEDIA MASTER PLUS
all of the file transfer capabilities of MEDIA MASTER *plus* it gives your IBM compatible the ability to actually *run* CP/M software. All for just **$59.95**.

ACCELERATE 8/16 includes MEDIA MASTER, an upgraded emulator *and* a special easy-to-install NEC V20 chip which lets your IBM compatible run CP/M software at maximum speed. V20 chip and software is just **$99.95**.

Call or write for full compatibility chart

MILESTONE

Business Project Planner

Originally $295 — Now just $99.95!

Never blow another deadline! Whether you're opening a branch office, hiring a sales force or launching a new product, nothing boosts the bottom line like being on time and within budget.

Now plan and execute complex tasks with ease. You first input manpower, costs and time requirements for a series of tasks and MILESTONE does the rest — creates a detailed workplan, identifies critical paths and analyzes alternatives.

INFOWORLD Magazine rated it's documentation GOOD and the program EXCELLENT in performance, ease-of-use and error handling. MILESTONE is available for *all* CP/M, MS/DOS, ZDOS & PC/DOS machines.

RemBrandt

Complete Business Graphics Toolkit™

Finally there's an easy an fun way to create graphics on your H/Z-89, H/Z-90, H/Z-100 *(CP/M only)* computer or any H/Z-19 equipped machine.

No extra hardware required! It works with a standard unmodified machine yet also supports the TMSI SuperSet ROM, and the Font19 Character ROM.

Freehand drawing: You can easily draw lines, boxes, circles and write on the screen in large characters. Full block operations are also supported — move, delete, fill, copy and more! Your graphic creations can be saved to disk and recalled at any time for further editing. Layout forms, design logos, draw diagrams and pictures. It's easy and fun to use.

Business graphics: REMBRANDT lets you create horizontal and vertical bar charts, pie charts and xy plots (scatter graphs). Use hand-entered data or read numerical data from virtually any source including dBase II, SuperCalc, MBasic, Wordstar and ASCII files.

Slide shows: Sequence your graphics on-screen using eleven cinematic special effects like wipes, fades and spirals. Produce electronic 'slide shows' without any programming.

Print your graphics: Print your graphic screens on most dot-matrix and daisy wheel printers. Interface with all word processors so that your reports can include charts, graphs or any graphic creation — intermixed with your text!

Compatible: It even reads, displays and prints *Ed-A-Sketch* files!

Affordable: Even with all of this power, REMBRANDT is available for an amazingly low price of... ***$39.95***

REMBRANDT runs on H/Z-89's, 90's, 100's and H/Z-19 equipped machines.

TECHNOLOGIES, INC.
22458 Ventura Blvd., Suite E
Woodland Hills, CA 91364

We accept **VISA, MASTERCARD** and **AMERICAN EXPRESS**

Order by mail or call our 24 hour toll free order line from anywhere in the US or Canada:

800-628-2828 (Extension 918)

Technical questions, orders: 818-716-1655 (9-5 PST)

Add $4 per order postage/handling. Overseas, add $12. US funds only. CA residents add 6% tax (LA County 6.5%)

Networking

Fred W. Kent
1057 Lake Road
Conneaut, OH 44030

This article will not be for every Huggie out there, but may help to enlighten those of you who have occasion to operate more than one computer in the same general location. My friend and part-time employer in the videotape business operates four IBM computers and being in need of a fifth, finally heeded my suggestion and availed himself of a Heath 158, which we put together for him in eight hours.

Since this particular computer was to aid in his busiest store, he desired a network program in order that his tape data files base themselves all from one master database. (We have some six thousand tapes in this store.) On weekends and holidays, lines are backed up out the door of this place. The particular application was to be a second handling facility for videotapes ins and outs.

A call to the IBM store was disappointing as the salesman glibly rattled off the items needed to create a network. Among these were a file server, two network programs, several interfaces and other special equipment such as hayes modems. When the salesman got to the bottom line, the total price came to a sum higher than the cost of a new IBM XT, in the neighborhood of $2400.00.

Upon digesting this information, I came to the conclusion that I should be able to produce a program capable of doing this in the BASIC language and so stated to the boss. Now the fun began as I said that if he would purchase a ten dollar patch cord, I thought I could write a suitable program, without any idea as to how to do it. Not to worry!

One of the smartest things I have done in years was to buy my 12 year old grandson a used H-89 computer and turn him loose with it. In six months, this kid learned more about computers than I have learned in ten years. I gave him a call and invited him to spend the weekend with grandpa and to bring his computer with him. His H-89 limps along on one internal hard sector drive, so he loves to come here and get turned loose on my trusty Z-100 color job with a winchester 10 meg. drive.

We started on a Friday evening with only a general idea of the direction we would take to produce the desired program and explored a number of ideas. We decided to bypass use of any modems as they tied up two telephone lines and were subject to line noises that could alter bits in the transmission of important data. We therefore rigged up a null modem cable between my serial port B and his serial port and also decided to transmit all data in ASCII form which would obviate the differences between my MSDOS and his CP/M operating systems.

By Saturday evening we had the general form of the programming necessary. All it cost me was a 16K memory upgrade for his H-89. This kid still works cheap. In the end, we did not have to write an elaborate program at all. It proved only to require a couple of subroutines added to the tape store master program.

The scheme finally arrived at then was to initialize two computers connected between their serial ports by a null cable with identical operating programs and databases installed. As each transaction at the checkout desk is completed on either computer, the essential updates necessary to update the database on the alternate computer is then transmitted into a temporary sequential file on the alternate which is then closed and reopened for input and the data fed into the alternate at the proper place in its master program to update the following files:

```
MASTRTPE .DAT
DAYLOG   .DAT
TAPEIN   .DAT
TAPEOUT  .DAT
TOTLIZER .DAT
```

When processing of these files is completed in the alternate computer, all files in both computers match. The temporary sequential file is then deleted and both computers return to their idle state which is a menu display readying both for the next transaction.

Problems which arose:

1. For reasons we were not able to ascertain, UART baud rates higher than 1200 baud were erratic. However, since out transmissions are quite short, averaging about 30 bytes per transaction, this rate is quite satisfactory. The average time for completion of processing in the receiving computer is less than two seconds.
2. Although the transmitting computer may send its data anytime the data is ready to go, the receiving computer, which must cut to a subroutine for processing received data, must hold the data received in the UART buffer until the receiving computer comes to its idle or menu section. In this menu section, special function keys which are used for menu selection and the serial

port are continuously polled. The menu display upon finding information available at its serial port blanks and displays the fact that data is being processed. The function "ON COM1: GOTO" was not suitable as it could interrupt the receiving computer while its operator was entering data causing a loss of input or a trashed transmission.

3. On rare occasions a database may have erroneous information and need to be updated. We had to carefully instruct the clerks to be sure and update both computers in such events.

4. On the IBM computer, we had to initialize the port to ignore the MS or modem status line. Even so, the IBM computer apparently sends and receives a recognition signal upon startup. We had to examine the first byte transmitted and kill the output file which proved to have nothing resembling data in it. The program now tests the first byte received for valid data. Oddly, this problem never appeared on my Z-100 and is peculiar to the IBM only.

5. We found that the line opening the serial port would crash if the other machine was not connected and operating. We therefore try to turn on both computers within a second of each other so that both may be convinced that there is truly another serial port on its output cable.

6. This scheme has the rather unique outcome of having its own backup at all times since the databases are maintained identical continuously and quite literally does not normally need a floppy disk backup performed daily. However, because of the occurrences stated in Item No. 5 above, it seems obvious that if one of the computers went down for repair, the programmer would have to lock out the subroutines to open the UART port and send data until such time as the repairs were completed and the second computer back on line. At this time, it would be a simple matter to copy all databases to a floppy disk and upgrade the other computer to the existing state.

7. We always use the BASCOM compiler on completed program files. This has several positive advantages, one being that the program executes in the neighborhood of forty times faster than BASIC. Another, is that clerks are prone to typing the wrong keys at the wrong times. In a BASIC program which is loaded but not running, we all know that any line numbers typed and entered will erase that particular line and trash the program until reloaded. Likewise, a BASIC program may be too easily altered.

8. Compiling this program for the Heath was ducksoup. However, using IBM BASCOM, we found that the compiled program would crash with an error message that the COM port was unavailable when compiling with the /O option to make a stand alone .EXE file. Readers take note that IBM linkers must link your PROGRAM.OBJ+IBMCOM.OBJ in order for the COM port to function. IBM carries their COM port support function in a separate file labelled IBMCOM.OBJ. It took us two days to figure this out as the BASCOM guide books are a little sketchy in places.

Finally, we have this program up and running and have had literally no difficulty thus far. During closeout routines, the two daylogs must match along with the store cash register. If this occurs, we know we have had successful communications between the two computers and need check no further for accuracy.

Listed below are the essential subroutines to create such a network. Total cost about $12.00 for a null modem cable compared to thousands for a so-called network system. This has obvious limitations but can work well in an environment where one needs more than one computer running identical files and where the stations can be very close together. We are using a 10 foot null modem cable.

Below, I submit a model program for you to follow:

```
1Ø '                    -SKELETON PROGRAM-
2Ø ' PURPOSE: Demonstrate record transfer by null modem
    cable
3Ø '
4Ø '
5Ø '             ---Initialize & open COM1:---
6Ø '
7Ø '
8Ø COLOR 2:CLS:KEY OFF
9Ø FALSE=Ø:TRUE= NOT FALSE
1ØØ XOFF$=CHR$(19): XON$=CHR$(17):' Set xon-xoff protocol
11Ø OPEN "COM1:3ØØ,N,8,,CS,DS,CD" AS #3:
    ' Open UART for send-receive
                                   Cancel status reports
12Ø DATA Send data,Exit
13Ø '
14Ø '
15Ø '          ---Main program & Menu display---
16Ø '
17Ø '
18Ø CLS:RESTORE
19Ø L=6:C=23:LOCATE L,C
2ØØ FOR Z=1 TO 2:READ D$:LOCATE ,C:PRINT Z;" . . . ";
    D$:NEXT
21Ø PRINT:LOCATE ,C+3:PRINT "Choice";:
    '     Select from menu
22Ø A$=INKEY$:IF A$="" THEN 24Ø ELSE X=VAL(A$):
    'No -- Go check for incoming data
23Ø ON X GOSUB 31Ø,1Ø6Ø:'  Yes -- Jump to subroutine
24Ø IF LOC(3)>Ø THEN GOSUB 57Ø:
    '     Anything received from alternate?
25Ø GOTO 22Ø:'  No -- Keep polling keyboard
26Ø '
27Ø '
28Ø '             ---Data entry section---
29Ø '
3ØØ '
31Ø CLS:PRINT,"--ENTER DATA TO BE PROCESSED & TRANSMITTED--"
32Ø LOCATE 25,2Ø:PRINT"Enter a <RETURN> if end of data":
    LOCATE 3,1
33Ø OUTSTR$="":'   Initialize OUTSTR$ to null value
34Ø LINE INPUT "Enter data : ";R$:' Get a string of data
35Ø '
36Ø '
37Ø ' -------------------------------------------------
38Ø '
39Ø '
4ØØ '
41Ø '    ---Process datafiles here for local entry---
42Ø '
43Ø '
44Ø ' --------------------------------------------------
45Ø '
46Ø '
47Ø IF R$="" THEN 48Ø ELSE OUTSTR$=OUTSTR$+R$+",":GOTO 34Ø:
    ' Get more data
48Ø Q=LEN(OUTSTR$)-1:OUTSTR$=LEFT$(OUTSTR$,Q):
    '   Get rid of last comma
49Ø '
5ØØ '
51Ø ' ---OUTSTR$ now contains all data to be transmitted---
52Ø '
53Ø '
54Ø GOTO 76Ø:'   ---Jump to upload section---
55Ø '
56Ø '
57Ø '                 ---Download section---
58Ø '
```

```
590 '
600 OPEN "O",2,"FILE.TMP":'   Temporary file to receive data
610 IF LOC(3)=0 THEN 690:'  No more bytes coming?
620 IF LOC(3)>82 THEN PAUSE=TRUE: PRINT #3,XOFF$;:
    'Send pause if buffer full
630 A$=INPUT$(LOC(3),#3):
    '  Get a byte from alternate computer
640 PRINT #2,A$;:'         and send to temporary file
650 IF LOC(3)>0 THEN 620:'  Get more bytes
660 IF PAUSE THEN PAUSE=FALSE:PRINT #3,XON$;:
    'Resume reception of data
670 FOR I%=1 TO 200:NEXT:'  Delay before looping back
680 GOTO 610:'   Loop for more bytes
690 CLOSE #2:CLS:LOCATE 25,10:PRINT "Download complete*";
700 GOTO 870:'  Jump to data viewscreen and/or processing
       section
710 '
720 '
730 '          ---End of download section---
740 '
750 '
760 '               ---Upload section---
770 '
780 CLS:PRINT,"---Uploading data to alternate computer---
790 PRINT #3,OUTSTR$:'Transmit data to alternate computer
800 PRINT:PRINT OUTSTR$
810 PRINT:PRINT"---The above is being uploaded to the
       alternate computer---"
820 PRINT:PRINT, "---Upload completed---"
830 FOR Z=1 TO 1000:NEXT:' Delay to read transmitted string
840 RETURN 150:'    ---Return to menu---
850 '
860 '
870 '            ---Process received data---
880 '
890 '
900 '
910 OPEN "I",2,"FILE.TMP":' /  Open temporary file for read
920 LINE INPUT #2,W$:'       /
930 PRINT W$:'               /      ---Read transmitted record---
940 CLOSE 2:'               /
950 '
960 '--------------------------------------------------
970 '
980 '
990 '           ---Process record received---
1000 '
1010 '
1020 '--------------------------------------------------
1030 '
1040 PRINT:PRINT "Hit any key to continue ";:F$=INPUT$(1):
     RETURN 150:' to menu
1050 '
1060 '
1070 CLOSE:CLS:END:'  ---End of program   Exit to BASIC---
1080 '
```

*

*** UPGRADE ACCESSORIES FOR Z-100 *** SERIES COMPUTERS

ZMF100A by FBE Research. A modification package which allows 256K chips to be used on the old-style motherboard to reach 768K. Simple assembly with no soldering or trace cutting. Compatible with Easy PC and Gemini Emulator. $60.00 alone or $148.50 with 27 256K RAM chips included.

SmartWatch by FBE Research. If you don't have a clock for your Z-100, get this one. More details under Z-150 upgrade listings $44.00

GEMINI EMULATOR BOARD. Makes the Z-100 compatible with the IBM PC library of programs $432.00

UCI EASY PC. IBM PC Emulator. Makes your Z-100 IBM Software Compatible. Full 8 MEG operation, color graphics and audio compatible. Retail $699.000, Payload $477.00

UCI EASY 87. Add an 8087 Numeric Coprocessor. $69.00 for the board without an 8087 Chip. With 5 MEG 8087 $197.00 or with 8 MEG 8087 installed $234.00

UCI MEMORY UPGRADE CARD. We recommend this one highly. The board has sockets for up to 2 MEG of RAM. With no RAM installed $328.00. With RAM installed and fully tested, 512K $387.00, One MEG $446.00, Two MEG $564.00 Add $35.00 for EasyDrive RAM Drive Software if desired.

UCI RAMSAVER. Maintains power on UCI MEMORY CARD RAM when computer is off. Save your programs in RAM with your computer off. Payload $177.00

UCI EASY-I/O. S-100 board that provides IBM PC communications port compatibility with your EasyPC. Easy I/O-1, One Serial Port $91.00. Easy I/O-2, Two Serial Ports, One Game Port, Clock-Calendar $127.00

UCI EasyWin. Winchester Drive Systems at reasonable prices. Complete Hard Disk Systems for mounting inside your Z-100. Systems complete with Seagate Drives, 21 MEG $745.00, 31 MEG $865.00. System without Dirve $277.00

CDR Z-100 SPEED MODULE. Run your Z-100 Computer at 7.5 MHz. Installs easily with no soldering. Externally switchable between Speed and Normal mode. Payload $48.00

*** ZENITH SOFTWARE FOR THE *** Z-100 SERIES COMPUTERS

Zenith packages with software, manuals and registration cards for the original Z-100 computer series (not for the IBM compatibles).

PART NUMBER	DESCRIPTION	LIST PRICE	SALE PRICE
MS-463-1	Z-Basic (16 bit) interpreter	$175.00	$24.00
MS-463-7	Multiplan	$195.00	$24.00
CB-463-11	Z-Chart	$150.00	$15.00
CD-463-2	Condor File Manager	$299.00	$24.00
PK-100-4	All 4 listed above	$819.00	$62.00
MS-253-1	Microsoft BASIC-80 (8-bit)	$175.00	$24.00
OS-53-2	CP/M-85 (8 bit)	$150.00	$24.00
OS-63-4	Z-DOS	$150.00	$35.00
CB-463-9	PECON Peachtree to Condor	$99.00	$15.00
RS-463-1	Peachtree General Ledger	$399.00	$38.00
RS-463-5	Peachtree Inventory Management	$499.00	$38.00
RS-463-75	PeachText 5000	$395.00	$77.00
SC-463-1	SuperCalc	$195.00	$38.00
WI-463-1	Remote Batch Terminal Emulator	$899.00	$28.00

*** CHIP SPECIALS ***

The finest RAM available and at PAYLOAD prices. Order one to one thousand chips and add only $2.00 for shipping.

64K Dynamic RAM, 150 ns	$1.35 each
256K Dynamic RAM, '50 ns	$3.28 each
256K Dynamic RAM, i20 ns	$3.87 each

V-20 CHIPS. High Speed NEC V-20-8 8088 replacement. These run at up to 8 MEG and are said to increase CPU speed 10-30%. Payload $14.75

8087 MATH COPROCESSOR CHIPS. Speeds and improves numeric processing. 5 MEG 8087-3 . . . . $129.00, 8 MEG 8087-2 . . . . $165.00

*** UPGRADE ACCESSORIES FOR Z-150/160 *** SERIES COMPUTERS

SmartWatch from FBE Research. Installs in ROM Socket on CPU Board in Zenith computer series Z-100/150/158/160. This tiny jewel of a product contains a ten year battery and keeps your computer informed of both time and date at each boot-up. Complete instructions and software included $44.00

MEMORY KIT #150-256-18. Includes a ZPAL chip which allows use of 256K RAM chips included (18 pieces 256K 150 ns RAM chips). Kit increases 128k memory to 640K or 256K memory to 704K. All chips plug into your existing Zenith Memory Board. Unbelievable but true $87.00

Winchester Hard Disk Drive Internal Set-up. Includes Winchester drive, controller/interface card, cables and all hardware. With 20 MEG (formated) drive $478.00. May be installed in Z-148 using an Expansion Card sold below.

PTZ-148 Expansion Card for Z-148. Includes 2 expansion slots plus a clock/calendar. $129.00

EVERCOM INTERNAL MODEM. Fully Hayes compatible 1200/300 baud with powerful BitCom software included $163.00

*** ZENITH SOFTWARE FOR THE *** Z-150/160 SERIES COMPUTERS

PART NUMBER	DESCRIPTION	LIST PRICE	SALE PRICE
RS-463-75	PeachText 5000	$395.00	$77.00
BP-5063-71	BPI Series Self-Training	$69.00	$25.00
BP-5063-8	BPI Personal Accounting	$195.00	$55.00

*** UPGRADE ACCESSORIES FOR H/Z-89 *** COMPUTERS

Magnolia Microsystems Double Density Controller. Soft-sectored disk controller. Supports up to four each 5.25" and 8" disk drives. Complete with cables, installation instructions and CP/M . . . $294.00

INTERNAL DUAL DRIVE SETUPS. Includes two half height double sided disk drives and all hardware and connectors required to mount inside your H-89. Steel mounting shield/case included.
MITSUBISHI MF501 Setup, 48 TPI, 6 MS seek, 320K $279.00
MITSUBISHI M4853 Setup, 96 TPI, 3 MS seek, 640K disks . . $284.00

*** HALF HEIGHT FLOPPY *** DISK DRIVES

MITSUBISHI M2896	8"	48 TPI DS/DD	1.2 MEG	$375.00
MITSUBISHI M501	5.25"	48 TPI DS/DD	320K/360K	$110.00
MITSUBISHI M504	5.25"	96 TPI DS/DD	360K/1.2 MEG	$177.00
MITSUBISHI M4853	5.25"	96 TPI DS/DD	640K	$130.00

*** SEAGATE HARD DISK DRIVES ***

ST-225 20 MEG Winchester Hard Disk	$393.00
With Western Digital Controller & Cables	$478.00
ST-238 30 MEG, Requires RLL type controller	$424.00
With Adaptec RLL Controller & Cables	$528.00
ST-4026 20 MEG High Speed for Z-200	$588.00
ST-4038 30 MEG High Speed for Z-200	$669.00
ST-4051 40 MEG High Speed for Z-200	$859.00
ST-4096 80 MEG High Speed with Software	$1425.00

MasterCard

PAYLOAD
COMPUTER SERVICES

VISA

15718 SYLVAN LAKE, HOUSTON, TEXAS 77062
PHONE (713) 486-0687

Please **MAIL** or **PHONE** your order today and expect prompt service. **MASTERCARD** and **VISA** gladly accepted with no additional charge. All hardware carries a 90 or more day warranty. Add $5.00 to all prepaid orders for handling and shipping, we pay the balance. Texas Residents please add 6.125% sales tax. We accept purchase orders from schools, government and approved accounts.

Sines And Cosines And 2-D Graphics On The Z-100

Part I

Thomas J. Vaden
5765 Grand Avenue
Riverside, CA 92504

Introduction

Computer graphics is a powerful tool used in the production of brilliant video displays, computer art and animation, and charts and graphs for business and engineering applications. Other applications of computer graphics include schematic and architectural drawing, flight simulation, motion picture special effects, and computer aided design. Computer graphics opens a diverse spectrum of capability ranging from visualization of complex concepts in science and mathematics to entertainment in the world of abstract art and animation.

The ability to create complex two-dimensional and three-dimensional drawings is an exciting extension of the home computer. However, many essential graphics capabilities such as location, rotation, and scaling are often shrouded in mystery. Algebraic and trigonometric identities are often combined with complex matrix operations and geometric manipulations to impose a seemingly formidable blockade to learning computer graphics.

The purpose of this article is to take the reader along on an exploratory, informative, and hopefully enjoyable excursion into the world of computer graphics. This article introduces simplified mathematical programming techniques used to produce two-dimensional computer graphics. Relationships between the graphs of functions and transformations necessary to display these graphs on the screen are discussed. Manipulations of screen displays are accomplished through the application of fundamental trigonometric identities. Specifically, location, scaling and rotation of simple two-dimensional figures are presented and programs are developed to demonstrate these concepts on the Zenith computer. Although this article is written for the Z-100, the concepts discussed are generally applicable to any system.

This article is written for two types of users: The practitioner who needs a basic understanding of the design of graphs, functions and charts for mathematical, engineering or business applications, and the artist who needs to express himself in a new and exciting media.

Characteristics Of Zenith Screen Display

The size of the screen display is measured in picture elements (pixels) or points. The Zenith Z-100 screen measures 640 pixels across horizontally by 225 pixels down vertically. Each character on a screen occupies a rectangular grid 8 pixels across by 9 pixels down. There are 80 characters across the screen and 25 characters down. The relationship between the character size and the display size is important when intermixing characters and graphics on the same screen.

Plotting a point on a screen requires that we specify the location of the point as a pair of (x,y) coordinates. The x coordinate gives the horizontal distance across the screen and the y coordinate gives the vertical distance down. Point (0,0) is located at the top left corner of the screen display, point (639,0) at the top right, point (0,224) at the bottom left, and point (639,224) at the bottom right. The center of the screen is located at approximately the point (320,112).

Converting A Function To Screen Display

In a graph of a mathematical function, the positive direction is to the right on the x-axis and upwards on the y-axis. However, on a standard display screen, as the y-value increases, the point to be graphed is located further down the screen (in the negative direction mathematically). Therefore, to convert to the correct screen display, we must negate the y-value of the function we wish to plot, otherwise the function will be inverted (turned upside down). For example, to plot SIN(x), we must set

```
    y = -SIN(x).
```

Next, we wish to locate the x-axis (horizontal axis) of the SIN(x) function in the center of the screen and the y-axis (vertical axis) on the left side. This is equivalent to locating a starting point for the graph on the left side of the screen (half-way down) at the point where the x-value equals zero and the y-value equals 112. In Cartesian coordinate notation, this is written as the point (x,y) = (0,112). Figure 1 illustrates the location of several points on a full screen display of the SIN(x) function.

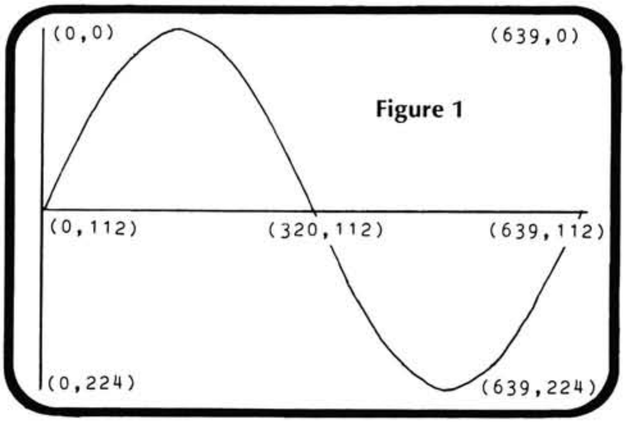


Figure 1

Locating the x-axis in the center of the screen requires that each point in the display area be shifted vertically. Since the center is to be located at y = 112 pixels, the function to be plotted is modified by adding 112 to the y coordinate.

```
y = 112 - SIN(x)
```

If we graph this function, we obtain a sine wave with a maximum height of one pixel (almost a straight line across the screen). To convert this sine wave to full screen height, we should multiply by a scaling factor SY. To find the scaling factor, divide the height of the display area by the height of the function, and round downwards to the next lower integer. Since the full screen height of the display area on the Zenith monitor is 225 pixels and the maximum height of the sine function is 2, the conversion factor SY = 225/2 = 112 rounded downwards. The function now becomes

```
y = 112 - SY * SIN(x)
```

In a similar way, to find a scaling or conversion factor which represents a full screen width display, we divide the width of the screen (640 pixels) by the range of x-values. For example, to display the sine wave as x goes from zero to 2*PI, the conversion factor SX = 640/(2*PI) where PI = 3.14159.

```
x = SX * x
```

In general, we can set

```
SX = DX/RX
SY = DY/RY

x = LX + SX * x
y = LY - SY * f(x)

LX,LY are screen location coordinates
SX,SY are scaling/conversion factors
DX,DY are maximum lengths (display area)
RX,RY are maximum ranges (function values)
```

For example, to center the graph of f(x) = SIN(x) in the upper right quadrant of the screen (see Figure 2):

```
SX = DX/RX = 32Ø/(2*PI)
SY = DY/RY = 112/2

x = 32Ø + SX * x
y =  56 - SY * SIN(x)
```

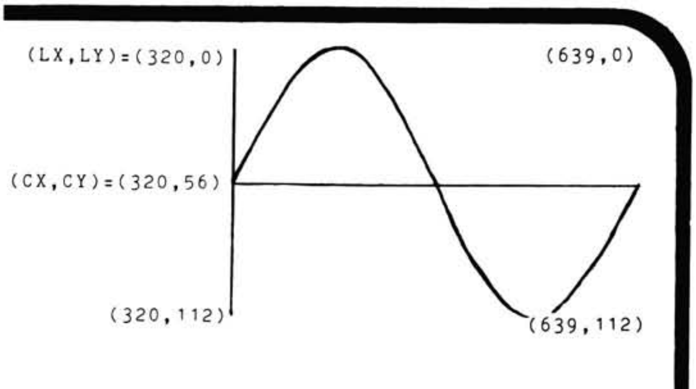


Figure 2

Plotting A Sine Function

In BASIC, the graph of a mathematical function may be generated by placing the function within a FOR...NEXT loop. The purpose of the loop is to generate several x-values for the function y = f(x). The start and end points for the loop correspond to the lower and upper limits of the function along the x-axis, and the resolution of the graph is determined by the number of steps within the loop. The function f(x) may represent any mathematical equation such as a straight line, a circle, or a special function like COS(x), EXP(x).

In our first example, we will graph the function y = SIN(x). The x-values and their corresponding y-values locate the coordinates for each point in the function. The start and end points for the loop define the range of x-values which, in turn, determines the range of y-values. For instance, in plotting a sine function, the x-value may be any real number within the limits/range defined by the user, whereas the y-values range from a minimum of -1 to a maximum of +1 units. These ranges determine the boundaries of the space occupied by the function. Both the x-values and y-values are then modified within the loop to correspond to screen display values. These modified values determine the location and boundaries of the graph on the screen display. The modified coordinates are used as arguments in the LINE command, which plots the function in pixels.

Program 1-1 demonstrates the graphing of the sine function for an arbitrary location on the screen. As a useful exercise, the reader should try several changes to program 1-1, such as inputting various display sizes (DX,DY) and locations (LX,LY), or changing the start point (S1) or end point (S2). For instance, if S1 = 0 and S2 = 6*PI, we would obtain three complete cycles of the sine wave within the display area selected. Also, we could change the function SIN(x) to COS(x), or modify the program to draw several graphs on the same screen.

Program 1-1 is set to display a sine wave which has two full cycles. The range of x equals 4*PI (RX = 4*PI). The size of the display area is 160 pixels across by 56 pixels down, which occupies about one sixteenth of the full screen area. The upper left corner of the display area is located at the point (LX,LY) = (320,112) in the center of the screen. The x-axis is centered in the display area by the formula CY = DY/2 + LY. The y-axis is located on the left border of the display area.

It is instructive to experiment with the number of steps (NS). The number of steps determines the size of the step (ST) and the resolution of the graph. For instance, if NS = 10, then there are only ten steps or lines drawn on the graph and the resolution of the graph is poor. On the other hand, if we use the maximum number of steps possible (equivalent to the maximum number of pixels which can be placed horizontally in the display area, DX), then we have very fine resolution. However, to obtain this fine resolution, we must sacrifice computational speed. A nice balance between speed and resolution for a full screen display is 60 steps for a sine function. Temporarily changing the LINE command to a PSET command will prove helpful in determining the best resolution for a specific function or application.

Program 1-1

```
10  REM - PROGRAM 1-1, SINE WAVE
20 '*****************************'
30 '*****    THOMAS J.VADEN  *****'
40 '*****    COPYRIGHT 1985  *****'
50 '*****************************'
100 REM - SET DISPLAY BOUNDARIES
110 CLS
120 DX=160                   'SIZE OF DISPLAY AREA
130 DY=55
140 LX=320                   'UPPER LEFT COORDINATES
150 LY=112
200 REM - SET AXIS
220 CY=DY/2+LY               'CENTERS Y
240 LINE (LX,CY)-(LX+DX,CY)  'DRAWS X-AXIS
250 LINE (LX,LY)-(LX,LY+DY)  'DRAWS Y-AXIS
300 REM - INITIALIZE GRAPH
310 PI=3.14159
320 S1=-2*PI                 'START POINT IN RADIANS
330 S2=2*PI                  'END POINT IN RADIANS
340 RX=S2-S1                 'RANGE OF X-VALUE
350 RY=2                     'RANGE OF Y-VALUE
360 SX=DX/RX                 'SCALE X-VALUE
370 SY=DY/RY                 'SCALE Y-VALUE
380 NS=60                    'NO. OF STEPS IN PLOT
390 ST=RX/NS                 'STEP INCREASE
1000 REM - DRAWS SINE CURVE
1010 PRESET(LX,CY-SY*SIN(S1))  'LOCATES FIRST POINT
1020 FOR I = S1 TO S2 STEP ST
1030 Y=SIN(I)
1040 X= LX + SX*(I-S1)
1050 Y= CY - SY*Y
1060 LINE-(X,Y)
1099 NEXT I
```

Aspect Ratio/Distortion

In the next few sections, we will draw some simple shapes, such as circles and squares, and place these shapes at various locations on the screen. The first attempt to draw these shapes usually results in squares that are distorted into rectangles and circles that are stretched into ellipses. The incremental distances along the x-axis on a screen are significantly less than the corresponding distances along the y-axis.

We would like to control this distortion whenever we are attempting to draw "regular" polygons such as squares, pentagons, hexagons, and other figures with specific relationships between angles, lengths of sides, etc. so that the graphs appear proportional on the screen display. We must measure the ratio of vertical pixels to horizontal pixels necessary to produce equal lengths along the x-axis and y-axis, and use this ratio to correct for screen distortion.

The aspect ratio for the Z-100 is discussed in the Z-BASIC (Z-DOS) manuals. However, the discussion is somewhat confusing. Page 8.10 gives the aspect ratio as 0.4843 or 7/16, however 7/16 equals 0.4375, not 0.4843. Also, page 10.17 gives the aspect ratio as 0.4844 or 31/64. A careful measurement of the lengths of the x and y distances indicates that the standard green monitor for the Z-120 has an aspect ratio of 7/16 and the ZVM-135 color monitor has an aspect ratio of 31/64.

An aspect ratio of 7/16 means that seven pixels down is equivalent in length to 16 pixels across the screen. A display area 160 pixels across by 70 pixels down forms a square. Given a horizontal length of 100 pixels, the equivalent vertical length is found by multiplying the horizontal length by the aspect ratio:

```
DX = 100
DY = 100*7/16 = 44 pixels
```

We should use the correct aspect ratio for the specific monitor in use in order to prevent distortion. Any distortion will be especially noticeable when rotating figures in a two-dimensional plane. To determine the aspect ratio for other monitors, draw a line 100 pixels across the screen and compare its length to a line drawn 100 pixels down the center of the screen.

Plotting Circles

The CIRCLE command in ZBASIC provides an easy method to graph a circle. However, we will follow a more instructive approach and expand the concepts for graphing simple mathematical equations (program 1-1) to include parametric equations (program 1-2) which use trigonometric identities to graph a circle. In a parametric equation, both the x and y coordinates of a point on a graph are functions of an independent variable, i.e., x = f(I), y = f(I). These concepts are further developed in programs of increasing complexity and eventually used for rotation of shapes in a two-dimensional plane.

Circles, ellipses and other polar functions may be easily plotted on a computer using sines and cosines. A straightforward approach is to generate values for an angle between 0 and 2*PI radians and plug these values into the equations for the x and y coordinates of a circle. Location and scaling transformations convert these values to screen values, and the aspect ratio makes the circle look like a circle.

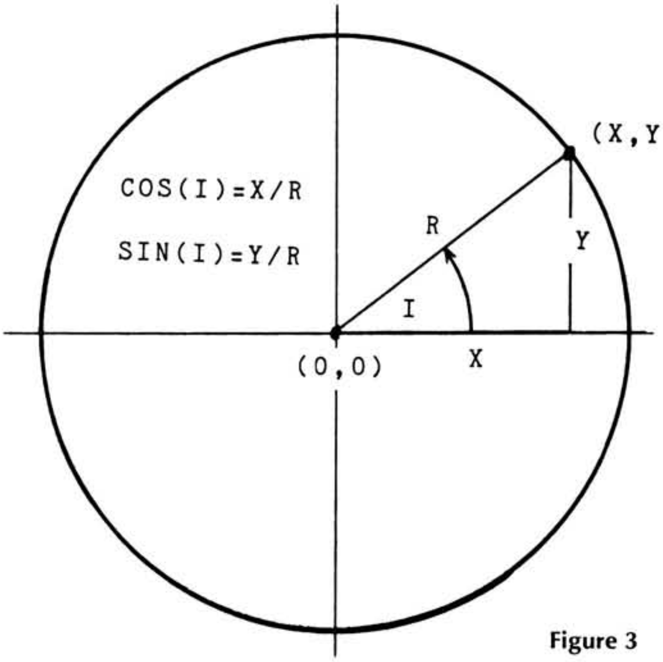


Figure 3

When drawing a circle, by convention, the counterclockwise direction is considered positive. The direction of rotation will prove useful when we discuss rotations of mathematical functions, miscellaneous shapes and sketches in two-dimensional space.

The equations for a point on a circle with center at (0,0) and radius R are obtained using trigonometric identities (see Figure 3).

```
COS(I) = x/R
SIN(I) = y/R
```

Without any loss in generality, we can set the radius of the circle equal to one (R = 1) and negate the y-value to achieve counterclockwise rotation.

```
x = COS(I)
y = -SIN(I)
```

Locating the center of the circle at the point CX,CY and scaling the x and y coordinates, we find that

```
x = CX + SX * COS(I)
y = CY - SY * SIN(I)
```

In program 1-2, we specify the horizontal length of the display area (DX) and set the vertical length (DY) equal to the product of the Aspect Ratio times DX (DY = 0.4375*DX). The display area is now a perfect square. We then specify the location (LX,LY) of the upper left corner of the screen display. This sets the boundaries of the display area. We locate the center (CX,CY) by adding half the length of the display area to the location of the upper left corner, and draw the x-axis and y-axis through the center. Since the radius of the circle was set equal to one, the maximum range (RX,RY) in the x or y direction is two. The scaling factors (SX,SY) are used to plot the function exactly within the display boundaries by converting function values to display values. The PRESET command is used to initialize the first point on the graph. The circle is then drawn in a counterclockwise (positive) direction using the LINE command within a FOR...NEXT loop.

Program 1-2

```
10 ' PROGRAM 1-2, CIRCLE
20 '******************************'
30 '*****    THOMAS J.VADEN  *****'
40 '*****    COPYRIGHT 1985  *****'
50 '******************************'
100 REM - SET DISPLAY BOUNDARIES
110 CLS
120 DX=400                      'SIZE OF DISPLAY AREA
130 DY=.4375*DX                 'CORRECTS FOR DISTORTION
140 LX=120                      'UPPER LEFT COORDINATES
150 LY=15
200 REM - SET AXIS
210 CX=DX/2+LX                  'CENTERS X
220 CY=DY/2+LY                  'CENTERS Y
240 LINE (LX,CY)-(LX+DX,CY)     'DRAWS X-AXIS
250 LINE (CX,LY)-(CX,LY+DY)     'DRAWS Y-AXIS
300 REM - INITIALIZE GRAPH
310 PI=3.14159
320 S1=0                        'START POINT IN RADIANS
330 S2=2*PI                     'END POINT IN RADIANS
340 RX=2                        'RANGE OF X-VALUE
350 RY=2                        'RANGE OF Y-VALUE
360 SX=DX/RX                    'SCALE X-VALUE
370 SY=DY/RY                    'SCALE Y-VALUE
380 NS=60                       'NO. OF STEPS IN PLOT
390 ST=(S2-S1)/NS               'STEP INCREASE
400 X1 = CX + SX*COS(S1)        'INITIALIZE X VALUE
410 Y1 = CY - SY*SIN(S1)        'INITIALIZE Y VALUE
1000 REM - DRAWS CIRCLE
1010 PRESET(X1,Y1)
1020 FOR I = S1 TO S2+ST/2 STEP ST
1030 X = CX + SX*COS(I)
1040 Y = CY - SY*SIN(I)
1060 LINE-(X,Y)
1099 NEXT I
```

Plotting Regular Polygons

The sides of a regular polygon have equal length. Plotting a regular polygon is a matter of specifying the correct number of steps (NS) in program 1-2, since the number of steps is equivalent to the number of sides to be plotted. If the number of sides is set equal to three, then an equilateral triangle will be plotted. If the number of sides is set equal to four, then a square will be plotted.

The diagonal of the square equals the diameter of the circumscribed circle which is set equal to the length of the display area. The square appears to be a diamond because of its orientation on the screen. The orientation can be changed by specifying a new start point (S1), which rotates the square. If the start point is changed to PI/4, then the sides of the square will be correctly oriented, parallel to the axes.

Rotation Of Regular Polygons

We can rotate a regular polygon about its center (CX,CY) by modifying program 1-2 to increment the start point (S1) and setting the end point (S2) equal to S1+2*PI. This converts the start point into an angle of rotation. An alternate procedure is to insert the angle of rotation into the sine and cosine formulas, which increments the polar coordinates of the vertices of the polygons. Program 1-3 is designed to rotate regular polygons about their centers without distortion. Specifically, this program rotates a square about its center by incrementing the corners of the square through several rotations. Several interesting patterns are formed by removing the CLS from the FOR...NEXT loop and changing the aspect ratio or the number of rotations (NR). Although program 1-3 may be modified to create numerous patterns, it is very limited in use. If the aspect ratio or the scaling factors are modified to stretch a square into a rectangle, the angular relationships of the rectangle will be distorted (sheared) as the rectangle is rotated around its center. Also, the vertices of the polygons are located on a circle with fixed center and radius. Thus, the polygons created cannot be rotated about arbitrary points on the screen. Finally, the calculation of sines and cosines within FOR...NEXT loops increases computational time and slows the image display down noticeably.

Program 1-3

```
10 ' PROGRAM 1-3, ROTATES REGULAR POLYGONS
20 '******************************'
30 '*****    THOMAS J.VADEN  *****'
40 '*****    COPYRIGHT 1985  *****'
50 '******************************'
100 REM - SET DISPLAY BOUNDARIES
110 CLS
120 DX=200                      'SIZE OF DISPLAY AREA
130 DY=.4375*DX                 'CORRECTS FOR DISTORTION
210 CX=320                      'CENTER COORDINATES
220 CY=112
300 REM - INITIALIZE GRAPH OF POLYGON
310 PI=3.14159
320 S1=0                        'START POINT
360 SX=DX/2                     'SCALE X-VALUE
370 SY=DY/2                     'SCALE Y-VALUE
380 NS=4                        'NUMBER OF SIDES
390 ST=2*PI/NS                  'STEP INCREASE
400 REM - INITIALIZE ROTATIONS
410 N1=PI/4                     'START POINT
430 NR=23                       'NO. OF ROTATIONS
440 AR=2*PI/NR                  'ANGLE OF ROTATION
500 REM - ROTATE POLYGON
```

```
510 FOR K% = 0 TO NR
520 A = N1+K%*AR
530 X1 = CX + SX*COS(S1+A)
540 Y1 = CY - SY*SIN(S1+A)
550 FOR DELAY = 1 TO 80  : NEXT DELAY
599 CLS
1000 PRESET (X1,Y1)
1010 FOR I% = 0 TO NS
1020 S=S1+I%*ST
1030 X= CX + SX*COS(S+A)
1040 Y= CY - SY*SIN(S+A)
1060 LINE-(X,Y)
1099 NEXT I%
1199 NEXT K%
```

Conclusion

Up to this point, we introduced a number of steps necessary to convert two dimensional graphs of simple mathematical functions, such as SIN(x), to a screen display. We defined the functions so that we could specify the exact location and size of the display area. We also discussed ways to eliminate distortion of graphs due to differences between vertical and horizontal scaling of screen display areas. Then we plotted a circle using parametric equations and extended this concept to the rotation of regular polygons with vertices located on the circumference of a circle.

In Part II, we will define relationships between characters and graphics in order to intermix graphical displays with legends and titles. We will rotate points and shapes defined in the polar coordinate system about arbitrary centers, without distortion or shear, and extend this concept to the rectangular coordinate system. We will then use rotations to create artistic patterns from an assortment of shapes plotted in the rectangular coordinate system.

Heath/Zenith Related Products

Jim Buszkiewicz
HUG Managing Editor

Through the efforts of a number of people, there is now a map database which can be used on most microcomputers which can read standard MSDOS 360k 5" floppy disks. This database was extracted and compressed from the **World Data Bank II** files which are distributed by the National Technical Information Service (NTIS). Since this database has been adapted to work on microcomputers and is not in the same format (9 track tape) as that used by NTIS, we have verbal permission from NTIS to place it in the public domain. This Micro WDB II contains some 178,000 points and provides world wide coverage for coast lines, country boundaries, state boundaries, islands, rivers, and lakes. Each line type is provided at five levels of detail.

Because the total database takes a little over one megabyte of disk storage, each line type has been extracted into its own file so that the database can be distributed in the standard MSDOS 5" floppy disk format. Several basic utilities, with Pascal source code, are provided to assist in customizing the files to individual needs. The distribution disks also contain basic documentation which describes the files and two programs, again with Pascal source code, which display the map data on Z-100s, Z-150s, as well as CGA and EGA based systems. Please note that the purpose of these disks is to make map data more readily available. They do not provide a tutorial on map generation and projections or writing graphics programs. The included programs do, however, provide a good starting point for building map displays.

We are asking $10.00 to defray the cost of shipping, handling, and the five (5) disks which are required to hold all of the data, documentation, and programs. After receipt of the database, it, and the programs, may be freely copied, placed in club libraries, etc. but not resold. To order, send $10.00 to Micro Doc, 3108 Jackson Street, Bellevue, NE 68005. If you have questions please call Fred Pospeschil, (402) 291-0795 (7-10 PM Central).

Burton Systems Software has announced a new product called TLIB. TLIB is a super-fast source code revision control system and librarian for PC-DOS and MS-DOS computer. Like the famous Unix utilities SCCS and RCS, it maintains compact, annotated libraries containing all the past and current versions of a program. This allows regression in case a new bug is introduced, and easy determination of what changed from version to version. TLIB works with Local Area Networks, and it handles synchronized control of multiple related source files. A copy of Landon Dyer's excellent public domain MAKE utility is also included; it features macros and full source code for both MS-DOS and VAX/VMS versions. TLIB is not copy-protected. TLIB is available for $99.95 plus $3 shipping and handling from: Burton Systems Software, P.O. Box 4156, Cary, NC 27511-4156, (919) 469-3068.

Now available from **InterContinental Microsystems**, is 'The Z-100 Node'. This system allows Z-100 users for the first time, obtain a protocol conversion node to link those computers with other PCs. It allows Z-100 users a fast, easy way of networking Z-100s into local area networks.

"ICM saw a real need among Z-100 users for a device that would allow their busses to network with IBM-PC compatible machines", said Ernest Wassmann, director of marketing for Inter-Continental Microsystems. "Government offices that purchased Z-100 systems can now eliminate communications cul-de-sacs in their network technology".

The Z-100 node includes LANS100 and TurboDOS operating software, and is designed with PC configuration hardware. Suggested retail price is $595. For more information, contact Ernest Wassmann, InterContinental Microsystems, 4015 Leaverton Court, Anaheim, CA 92807, (714) 630-0964.

Continued from Page 9

drive software to expect the bank switch port at a different address so no conflict occurs (you have to buy the new RAMdrive software).

If you are not using their RAMdrive, and just want to use MS-DOS 3.10, you can do one of two things: change the port address on the RAM board so it does not conflict (the ports the Z205 initialization code uses are 98H through 9FH), or patch out the offending initialization code.

I published the following patches in the December 1985 issue of the Al Lynch HUG (Tampa) newsletter. The patches remove the BIOS access to ports 98H through 9FH.

BIOS Version 2.22

```
Address    Old    New

 2DØE      EE     9Ø
 3ØC4      EE     9Ø
```

BIOS version 3.00

```
 2DDC      EE    9Ø
 3194      EE    9Ø
```

I don't have patches for other versions, but the code fragments to look for are:

```
2DD6 BA98ØØ     MOV        DX,ØØ98
2DD9 B9Ø8ØØ     MOV        CX,ØØØ8
2DDC EE         OUT        DX,AL
2DDD 42         INC        DX
2DDE E2FC       LOOP       2DDC

318C BA98ØØ     MOV        DX,ØØ98
318F B9Ø8ØØ     MOV        CX,ØØØ8
3192 ØC4Ø       OR AL,4Ø
3194 EE         OUT        DX,AL
3195 42         INC        DX
3196 E2FC       LOOP       3194
```

Change the OUT DX,AL instruction to NOP.

A comment on a different subject. Some MS-DOS 2.0 (and above) users have apparently misread the CONFIG.SYS documentation about the DEVICE statement. There is no (apparent) restriction on the number of DEVICE statements which can exist in your CONFIGUR.SYS file (except memory limitations). I typically use two or three DEVICE statements (tape driver, RAM driver, ANSI driver).

Sincerely,

Cliff Kimmery
2045 Sunset Grove Lane
Clearwater, FL 33575

Windows For The Z-100

Dear HUG:

I recently purchased "WINDOWS" for my Z-100 computer and I must say that I have seen colors on the Z-100 that I have never seen before. What's best is that this program allows some IBM compatibility . . . or so I thought. When I am using "Windows Write" all the screen display works correctly. When I try to print a file to my Panasonic 1091 printer the program works correctly . . . the first time. On the second printing of any file (without resetting the computer) the system locks up. Drive B: is usually spinning when I must reboot. What is really odd is that MS Paint works perfectly. My computer has 768K installed and full video RAM.

By the way, Pat Swayne's patch that modifies the number of lines on screen in WordStar for PC-compatibles, (see PC compatibles, November REMark) also works for the Z-100 while in an interlace mode.

Can anyone help with Windows printing problem?

Sincerely,

Sam Amantia
4608 Cypress Court
Tallahassee, FL 32303

Remote Keyboard Adapter

Dear HUG:

This letter is to let HUG members know of my new address. I will continue to support my H/Z-100 Remote Keyboard Adapter. Anyone with questions about the Adapter can contact me at:

Tim Ross
133 S. Chevy Chase Drive, Apt. 122
Glendale, CA 91205

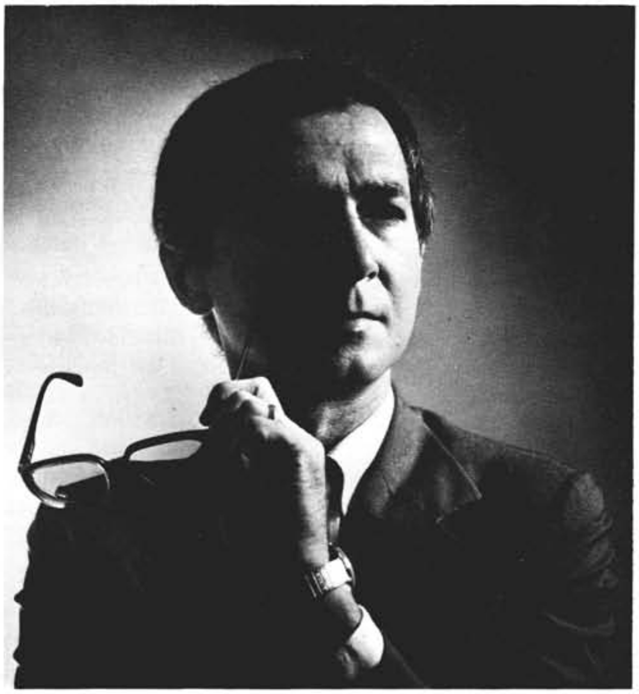
C. Itoh 8510A Printer Drivers Under MS-DOS 3.1

Dear HUG:

I am asking for assistance from anyone concerning the C. Itoh 8510A dot matrix printer and the transient command PSC in MS-DOS 3.1. The DOS manual describes the PSC command as a transient DOS command who's purpose is "Outputs all graphics and special characters from the screen to the printer". It then goes on to list the seven printers supported, of which mine is not on the list. I have called the Heath Operating systems help line (1-616-982-3860) and they said that "No, a PSC file does not exist for the C. Itoh 8510A at this time, although we are working on a number of other printer drivers which should be out sometime soon." When I asked if they had a program to allow the user to generate a printer driver (Microsoft Word and many other word processors supply such utilities), I was told "No, we don't, but gee! That's a neat idea!" When I asked about getting the source code for the existing printer drivers so I could write my own utility I was told that Heath was under contract with Microsoft not to give out that source code. When I called Microsoft, I was told I could BUY a license to look at the source code, but the price they quoted was astronomical. All I want to do is print out graphics characters and screens to my dot matrix printer which supports both features! The printer simply uses a different sequence of escape and control characters than the Epson MX-80 to accomplish these tasks! Does anybody know of an existing printer driver or printer driver generator? There are at least 10,000 8510A's in America and some of them must be hooked up to IBM PCs and compatibles like the H-148 and H-158. Any help at all will be greatly appreciated.

Sincerely,

Kurt A. Vogel
4153 Bacon
Berkley, MI 49072



Mainstream Computing

Joseph Katz
103 South Edisto Avenue
Columbia, SC 29205

This month we start, appropriately, with the first computer products in Heath's new *Starter Kits*, then go on to Boca Research's inexpensive new i/o (input/output) boards and the way you can use one of Zoom Telephonics' *Zoom/MODEM PC* internal modems with such boards to have three of four possible COM ports on your compatible the easy way. If there are any themes here this month they're ports and TSR programs, those useful nuisances that pop up — sometimes — into another program to give your computer simulated multitasking capabilities. Since I didn't plan things this way, you'll have to keep your wits about you to see those threads. I think they're here, though, because I've already read this column. It's a pretty good column, I think. If you want a second opinion, I think it's useful too.

Heath's *Starter Kits*

I've been curious about the computer products in that new line of *Starter Kits* Heath announced last August. It's an attractive concept: useful electronic equipment in easy-to-build kits for $50 or less. The reality could be even more attractive because there are many tools and gadgets to make computing easier, more effective, or more fun, but manufacturers tend to put premium prices on things for computers. Since brief catalog descriptions of electronic stuff rarely mean much to me, I wondered about the real meaning of "easy to build" and also about whether it would be translated into useful products or those that are little more than simple doodads, geegaws, and knickknacks. When the first three computer-related *Starter Kits* arrived a few weeks ago, therefore, I tore into them. Not to worry. What I found was useful electronic equipment in easy-to-build kits priced at $50 or less. You might think of them as *Turbo Heathkits*.

The comparison is to Borland International's marketing approach for programming tools. My guess is that the Starter Kits are at least partly intended as lures for big-ticket Heathkits such as computers and television sets. Most people today who buy microcomputers, especially those who buy compatibles, shop for them as appliances. Relatively few microcomputer buyers nowadays think about building their own electronic equipment from Heathkits. Many of us do cross the threshold and become interested in tinkering, though.

Look, for example, at the phenomenal sales of *Turbo Pascal*. What Borland did was package a useful programming language so it was less intimidating to novices, then price the package so it became more an impulse purchase than an investment. Suddenly the woods were full of people learning to program in *Turbo Pascal*. They weren't necessarily trying to write their own *WordStar* or *SuperCalc*, but who really cared? They were learning enough to pierce what had been a mystery and tinkering enough to produce some pretty good utility programs. They became "amateurs" in the best sense of that word: they worked for the love of it and utility was merely a byproduct.

Heath's idea, if I'm on the right track, seems to be to provide the same kind of people with kits that are hardware equivalents of *Turbo Pascal*. Let them see how easy and satisfying it is to build those little Heathkits and many of them ought to move on eventually to big Heathkits. They're the products on which Heath undoubtedly makes the real money.

It's missionary work of a kind that logically extends the direction Heath always has pursued most successfully. Heathkits have been and are better than the competition because they are engineered to be satisfying. Instead of parts thrown loosely into a carton with some miscellaneous sheets of paper, Heathkits are produced rationally. Logical groups of components are segregated into their own containers: these new kits, for example, go the final mile by fastening things like resistors and capacitors to strips of tape in the order you use them. A chart identifies each of those "Taped Components" by both electrical value and section of the circuit board for which it is destined. The board itself is sectioned off for sensible assembly, and the assembly instructions take excellent advantage of this keying.

Those instructions take nothing for granted: if you need to solder, they give a short course in soldering — including the way to position the soldering pen; when you need to insert a part in the board, you're told just how to position the part for best fit; and when you need to use a wire jumper, you'll find the wire color-coded and a ruler printed on the bottom of each relevant page of the instructions so you won't make a mistake in measuring. It's a "paint-by-numbers" approach that assumes absolutely no experience and very little intelligence, which is just the right approach for starter kits. When a kit has parts so small I crave a magnifying glass, the kit usually includes a magnifying glass. No wonder the kits come with Heath's well-known fail-safe guarantee.

You can do these *Starter Kits* as monkey work and succeed every time, but it's possible to learn from them too. As with most kinds of learning, what you come away with depends on what you bring to it and the effort you expend to fit the unknown into the known. Don't expect much passive learning, except possibly the merely manual skills needed for kitbuilding. A novice should acquire some polish in that area just by building these Starter Kits according to their instructions. For the rest, there's a section in each of the instruction manuals I've seen that explains the theory of operation for the kit. You'll need at least a little basic knowledge of electronics to follow along. My wish is that Heath go further in this area, perhaps by explaining relevant basics for those of us who don't already know them, but I'm not sure if what I want can be done usefully within the confines of a manual that already covers assembly and operating instructions.

At any rate, the only thing needed for Heath to have a line of *Turbo Heathkits* was the right price point. That magic number seems to be $49.95, the original price of *Turbo Pascal*. It's now the maximum price in Heath's *Starter Kits*.

Maybe I'm having pipe dreams, trying to project my own great idea onto Heath. But it really is a good idea. Let's say I'm right. You then might wonder whether these *Starter Kits* are plants rather than baits: whether they're deceptively easy to build in order to mislead you into building the big Heathkits. In fact, though, there's nothing at all deceptive about the three *Starter Kits* I've received. They're excellent value and in no way should they lead you to underestimate the difficulty of building the only two big Heathkits I've had the opportunity to try. If you are going to be misled by these three kits, it'll be in the opposite direction. I found it much easier and faster to build my H-241 computer (the Heathkit version of Zenith's Z-241 AT-compatible) than to build *The AC Monitor* in the *Starter Kits*. That *Starter Kit* was about as hard for me as my H-158 (the Heathkit version of Zenith's Z-158 XT-compatible) and much harder than my H-241 (the Heathkit version of Zenith's Z-241 AT-compatible). Your experience may differ. I think not.

This first batch of computer-related *Starter Kits* was *The RS-232 Line Protector* (SK-202) at $14.95, *The Traffic Cop* (SK-204) at $39.95, and *The AC Monitor* also at $39.95. That's their hardness scale too. Where utility is concerned, I give them a different ranking: *The Traffic Cop*, *The AC Monitor*, and *The RS-232 Line Protector*. Your conclusions will depend on what you need or think you need.

The RS-232 Line Protector

I've recently become concerned about surges, spikes, and other nasties that can distress or even kill computer equipment. Some of those undesirables come through the AC line that powers the equipment, some from the telephone line through which a modem communicates, and some through the cable lines that connect one piece of equipment to another. I'm not paranoid: everything really is trying to zap my computers.

The RS-232 Line Protector is supposed to isolate a serial device — modem, printer, or whatever — and computer from that last source of damage. It's sixteen diodes and a couple of jumpers on a circuit board. There's a male DB-25 connector on one end, a female on the other, and a grounding wire coming off the center. You plug one connector of *The RS-232 Line Protector* into your whatever, run a cable from the other connector to your computer, and attach the grounding wire to a valid ground.

We're expecting arrival of an Apple *LaserWriter* any minute. It's a laser printer that connects to a serial port. As soon as that thing rolls through the door, *The RS-232 Line Protector* goes on it. For $14.95 and fifteen to thirty minutes of effort, it's cheap insurance on expensive equipment.

The Traffic Cop

The name "traffic cop" may not tell you that this kit makes up into a simple kind of breakout box. Since that explanation may not mean much more to you, I'll amplify it a little.

Technicians use breakout boxes to monitor the signals between two serial devices supposed to communicate according to the RS-232 "standard." You need to know which wires in a cable should carry which signals or else you can't make the devices work together properly. Of course the reason for a standard in the first place is so you won't have to know anything of the kind. Sometimes, if you're lucky, you don't: you just plug things together.

Most times, though, you need to know which wires do what. Then you ought to be able to assume you can run a wire in the cable from Pin 7 of your computer's RS-232 output to Pin 7 of your modem's RS-232 input and be sure you've connected the Signal Ground at each end. In fact Pin 7 and Pin 1 (the Protective Ground) are among the few things you can assume as following this standard on microcomputers. Because the RS-232 standard was devised prior to microcomputers as we know them, each manufacturer has to translate it for his own equipment — sometimes differently for different models of the same equipment. Breakout boxes "break out" the signals so you can see what's happening and change it until the devices work. Then you build a cable to reflect the wiring that made things work. And then you replace the breakout box with the cable so you can get real work done at last. A breakout box is a tool for figuring out how to build a connecting cable between two serial devices.

Although there are twenty-five possible signals in the RS-232 standard, most serial devices you'll consider for your microcomputer use no more than eight. Of course it's cheaper to build a tool for monitoring eight wires than twenty-five, so there are "junior" breakout boxes like *The Traffic Cop*. It has a male DB-25 connector on one end and a female on the other: you plug a cable from the computer into one and a cable to the modem, printer, or whatever you're trying to connect into the other. There are three pairs of LEDs on *The Traffic Cop* top panel and a slide switch associated with each pair. Set the switches until you can get as many of the LEDs as possible to glow, then try getting the computer to communicate with whatever it is you're trying to connect. Slide each switch until the whatever-it-is starts working. Then look at *The Traffic Cop* top panel and you'll see a simple sketch next to each LED showing how to connect the equivalent set of wires in your cable. (Of course you follow the sketch associated with the glowing LED in each pair.) *Voila!*

Ironically, *The Traffic Cop* arrived the very day I bought a full-fledged breakout box. Even with my frequent-flyer discount at our local electronics distributors the darned thing cost me $145. I was not disposed towards glee at finding a $39.95 equivalent among the day's parcels when I returned home with my new purchase.

Some kind of breakout box had become a necessary medical expense because several days earlier I had bought one of Radio Shack's sale-priced *DT-100* terminals and was going crazy trying to get it working on my H-241 with SCO *XENIX-286*. If you think it's rough trying to interface an RS-232 device from one manufacturer to a computer from another manufacturer, try doing it with XENIX from a third manufacturer. I suppose these are all nice people individually, but put them into the same room and I don't imagine they'll communicate well with one another.

To be fair, I faced a few special complications. I knew Radio Shack often puts its own house label on products from other vendors, but I've worked with various terminals so I easily recognized the *DT-100* as some kind of Wyse guy, which was why I bought it. A few things worried me, though. Although Radio Shack sells its own compatibles, this terminal always is advertised for the *Model 6000*, which is about as uncompatible as one can imagine. Could Radio Shack know something I didn't? Nah. (What I found out later from a friend, who is a technician for a well-known computer vendor, but who must remain nameless or else said vendor will frown upon him, is that the *DT-100* was a Wyse 60 terminal with Radio Shack's own ROM. Argh.) A further complication is that the *DT-100* has not one but two RS-232 interfaces, for DTE (Data Terminal Equipment) as well as DCE (Data Communications Equipment). To muddy the waters slightly — slightly! — the documentation for the *DT-100* is written in one or several foreign tongues masquerading as one oriental language mistranslated into incomprehensible English. I wasn't upset much because what I could figure out made me think that no one involved in writing the documentation believed for a moment that anyone who bought the *DT-100* would be stupid enough to try hooking it to an H/Z-241 with SCO *XENIX-286* anyway. You know by now that I think SCO XENIX is absolutely super on Heath/Zenith computers, but my admiration does not extend to its manuals. The best I can say is that they are excellent XENIX manuals, which is the equivalent of my saying that I'd prefer being run over by a Ferrari than by a Chevrolet. Great stuff if you know XENIX and have the leisure to travel through several looseleaf binders in search of what interests you, they are not much help with the kind of problem I was facing: after I accumulated various mumblings about */dev/tty*, I figured out that no one involved in writing the documentation believed for a moment that anyone who bought SCO XENIX-286 would be stupid enough to try hooking a Radio Shack *DT-100* to an H/Z-241 anyway. And there I stood with my piccolo.

I kept fiddling with a homebrew gadget that was a cable terminating in scads of alligator clips on each end. You may hear some day that I spent a while in a corner of my office making mewling little sounds, and that is true. But it happened only after I went around in enough circles to suspect that maybe my *DT-100* — which I had bought new in a sealed carton — was defective and that all the hours I put into trying to connect the thing had been wasted. When *The Traffic Cop* came I built it as therapy, then introduced it into The Great *DT-100* Escapade to see if the LEDs would glow. In less than ten minutes I had determined the proper cabling. So far the Radio Shack *DT-100* is a nice terminal for the H-241 and SCO *XENIX-286*. Would that all three had been delivered to me already talking to one another.

The Traffic Cop really is a handy tool when you have to connect recalcitrant RS-232 devices — which means nearly all RS-232 devices. For most such devices we encounter, *The Traffic Cop* is just as reliable as, and much easier to use than, a full-fledged breakout box. It took me about an hour and one-half to build *The Traffic Cop* itself.

The AC Monitor

It's a cute little brown box that plugs into an AC electrical circuit and monitors the power, and before I tried it out I would have thought *The AC Monitor* about as useless as the *SYS REQ* key on an AT-compatible's keyboard. Well, I was not right.

On the back of the box is a standard two-prong AC plug, and on the front of the box are eleven red LEDs and a pushbutton switch. Nine red LEDs constantly monitor line voltage in five-volt increments from 95 to 130 volts. One of the remaining two red LEDs kicks on if there is a voltage "spike" (a surge 200 volts or above) and the other kicks on if there is a voltage "fault" (either a drop below 90 volts or a complete power outage). The pushbutton is used to reset the spike and fault indicators after either condition has been detected.

I was too insulated by almost-ideal surroundings to have anticipated much use for *The AC Monitor*. A couple of years ago we had a contractor erect a small building designed for me to work with bunches of computer equipment and write my little heart out. My office was electrified with computers in mind. So who — in my egocentric view — needed *The AC Monitor*? Then Janet reminded me of an unmodernized circuit remaining in the house itself. When we had an old CP/M machine on it, the thing would spontaneously reboot from time to time. No one could find anything wrong with that computer, so we simply moved it to one of the circuits we had been able to modernize. Everything worked fine. That was years ago and I had forgotten. I plugged *The AC Monitor* into that circuit and immediately the 115-volt indicator began to flicker. When I checked again after a few minutes, the spike LED was on. I reset and sat down to watch. In a few minutes it popped on again. My thinking now is that before I plug any of my computers into a foreign circuit I'll check it for a while with *The AC Monitor*. There are professional line monitors selling for bunches of money and I'm sure they do a proportionately more sophisticated job than *The AC Monitor*. I think of it too as cheap insurance.

The AC Monitor was the toughest of the three kits for me to build: assembly took six and one-half hours spread over two evenings, exactly the same time it took me to build my H-158 computer and more than four times as long as it took me to build my H-241. The reasons are that *The AC Monitor* is many parts crammed onto a 2 × 3-inch circuit board and my eyes are of an age when they are reluctant to change focus. If your eyes are newer than mine and less set in their ways, you should have an easier time with this kit.

Boca Research's 1MB Add-on

I said I would let you know about the 1MB piggyback that brings Boca Research's *BocaRAM* up to 2MB of expanded memory. One came a few minutes ago, while I had the top off my H-241, so I turned off the computer, extracted the *BocaRAM*, and installed the daughter board: it snaps into a socket and over a stud on the *BocaRAM*; there's only one way to do it and it's so obvious there's no way to go wrong. Then I replaced the *BocaRAM* and turned on the computer and had 2MB of expanded memory. I don't know what more to say about a board that installs so easily and works properly the first time.

On second thought there is something more to say. H/Z-248s seem fussy about add-on boards of any kind. I wouldn't call it a "problem": the machine runs fast enough at 8MHz with no wait states that many third-party boards can't keep up with it. Don't think you can buy just any AT-compatible board, shove it into a 248, and expect happiness ever after. You might not even get to the chuppah. Boca Research is among the earliest vendors to know the score. If you have an H/Z-241, for example, you'll want the standard *TophAT* board to fill out conventional memory on your machine to 640KB. If you have an H/Z-248, though, the standard *TophAT* board (which has 150NS RAM chips) won't work. I'm waiting for the 12MHz *TophAT*, which has 120NS RAM chips and is supposed to work on the 248. As for the *BocaRAM* extended memory board, it will work on either machine, the 248 as well as the 241. Of course I've tried it. It works as advertised. So does the piggyback board on both machines.

What I've been thinking recently is that it's hard to write about well-designed products that are simple to install and use, that work just the way they're supposed to work, and are sold at bargain prices. It's a thought, not a complaint.

Boca Research's New Input/Output Boards

While I was on a winning streak and had the computer open anyway, I decided to install Boca Research's *BocaIO/AT*. It provides AT compatibles with one more serial port, one more parallel port, and the base for an optional game port. (The game port option was not included, so I can't say anything about it.) Standard AT compatibles like the H-241 have one serial port and one parallel port. Add the *BocaIO/AT* and you have two serial ports and two parallel ports.

Installing the *BocaIO/AT* takes about the same degree of skill and amount of time as any of the Boca Research boards I've seen so far: not much of either. It's a short board, about 4-1/2 inches long, with an 8-bit edge connector that will slide into any vacant slot. You slide it into any vacant slot after you've turned off the computer. You turn on the computer. It reboots. You then run a setup program *Iofig*, which uses one of those natty bar menus from which you make selections with the arrow keys on your keyboard. It lets you configure the *Bocalo/AT* serial port as COM1 or COM2, its parallel port as PRT1 or PRT2 (*i.e.*, LPT1 or LPT2) and its game port as enabled or disabled. You also can disable the serial port, the parallel port, or both.

I disabled the game port (because I didn't have the optional doohickey that connects to it), set the serial port to COM2, and set the parallel port to LPT2:. Yes, I know I told you a few months ago that I had installed a *Zoom/MODEM PC-2400* internal modem as COM2 on the H-241. I've just changed it. You'll see in a bit why I think nothing of changing the modem's configuration.

At any rate, the internal modem is now COM3 and therefore my H-241 now has three serial ports and two parallel ports. Yes, I have a use for all those ports. Yes, all those ports are usable too. And, again yes, in a short time I'll have them all in use: as soon as my Apple *LaserWriter* arrives, it gets connected to either COM1 or COM2 because it's a serial printer. In the next section of this saga I'll explain how I got three working COM ports and can add one more should the right hardware arrive. A number of faithful readers, and a few testy ones, started writing since I mentioned it could be done last August and continue to write asking how to do it. Hold on for a few paragraphs and you'll get an easy way. Right now, though, I want to be tidy and finish this part of the saga.

Pay attention. After you've set the *BocaIO/AT*'s configuration from the *Iofig* menu, you have three ways to exit it. Use the second option ("Quit & Discard Changes") if you're just fooling around or have become confused. That way nothing you've done has any affect at all. It's the back door. If you're experimenting with setups to see what you really want, use the third option ("Exit without Storing"). That way the changes you've made are effected in RAM but not stored in the board itself. Only when you are sure that everything is set up properly should you choose the first option ("Exit and Store Changes").

The reason is that the first option stores the configuration in an EEPROM (Electrically Eraseable Programmable Read Only Memory) so the board is ready without further attention each time you boot the computer. The manual says so. What it doesn't say is that there is a limit to the number of times the EEPROM can be changed. It's a most generous limit — 1,000 times — so you shouldn't care unless you fiddle and faddle away with *Iofig*. A message when your cursor is on the first option tells you the limit, but fiddlers and faddlers tend not to read such things. You also should know that the manual is for the *BocaIO/XT* as well as the *BocaIO/AT*. The *BocaIO/XT* is a version of the board for XT compatibles such as the H/Z-151 and H/Z-158. Unlike the *BocaIO/AT*, the *BocaIO/XT* includes a clock/calendar. Don't get confused: the manual starts by explaining that there are two versions of the board. From then on it assumes you were paying attention, so if you rush through — as you have every reason to do with such an easily-installed board — you might get lost looking for the non-existent clock/calendar on the *BocaIO/AT* and its equally non-existent driver. I have the *BocaIO/XT* too and I'll install it in our H-158 as soon as I can, then report on it. The only reason I didn't do so immediately is that Janet said she had forgotten how the 158 looked when it had its case together. I can take a hint.

And I can review the manual. The one that came with each i/o board was defective — not the boards themselves, only the manuals. What happened was that some signatures were not "perfected" (printed on both sides), so we had blank pages 6 and 19 in one manual and blank pages 4 and 21 in the other. A complete manual has no blank pages among pp. 1-25. I've informed Boca Research about the problem. (It was a pre-Christmas phone call that did not spread cheer despite my opening, "Ho, ho, ho! Your manual crashed! Have a Merry Christmas and a Happy New Year!") I suppose the problem was corrected before you even knew it existed. Check your manual anyway. If it's incomplete, write Boca Research for a replacement. You probably can install the board from my instructions, though: as I've said, it's simple.

Four Serial Ports With The *Zoom/MODEM PC*

Heath/Zenith compatibles, like other PC, XT, and AT compatible microcomputers, come standard with one serial port that DOS addresses as COM1. You can add a second serial port easily just by inserting a board like *BocaIO/XT* (for the XT compatibles) or *BocaIO/AT* (for the AT compatibles) into the backplane of your computer and using Boca Research's *Iofig* program to tell DOS to address the new serial port as COM2. All the commercial and public domain programs I've used know how to work with either COM1 or COM2: select the port from a menu or the program's installation routine.

You can add even more serial ports, but not so easily. In his *Programmer's Guide to the IBM PC* (p. 210), Peter Norton says that IBM's original design for the PC allowed "up to seven serial ports to be added." (I think he means a total of nine serial ports, but maybe he means only a total of seven. The statement strikes me as ambiguous.) The difficulty in adding ports after COM2 is that these must use non-standard addresses. There is another difficulty if you want to use more than four COM ports: DOS has room for only

A magazine for Kit Hobbyists

Kit Builders Journal

Special Edition

$2.50

Volume 0, Issue 0 • January 1987

P/N 000-0000 Issue 0

INTRODUCING ... KBJ

The Kit Builders Journal, The Magazine For The Kit Building Hobbiest

- *The reader will be able to enjoy independent reviews on both electronic and non-electronic kits.*
- ***KBJ** will feature a wide variety of kit building information for Heath products as well as other do-it yourself projects.*
- *The **K**it **B**uilders **J**ournal is available through subscription only and will be delivered every other month, starting in January of 1987.*
- *The Journal will include how to sections devoted to building and using products, tips from Heath Technical Consultants on correcting problems or improving performance, small construction projects, modifications, questions/answers and other valuable information relating to this fun hobby.*

***For subscribers only!** KBJ will be offering super discount prices on selected Heath products in each issue. Just one product purchase during the year could save you several times the KBJ subscription price.*

Subscribe now and get your Kit Builders Journal! Order KBJ–2000–NM and get six big issues for only $9.95. Use your Visa, Mastercard or Heath Revolving Charge. Call TOLL FREE 1–800–253–0570. In Alaska and Michigan call 616–982–3411.

For outside U.S. orders send your name and postal address to Heath Company P.O. Box 1288, Benton Harbor, MI 49022. Rates in U.S. Funds for Canada, APO/FPO and all others: $17.95.

four addresses in the table starting at 0000:0000. The address for COM1 starts there, the address for COM2 follows, and them come blanks where it is possible to patch in addresses for two more COM ports. So COM1 and COM2 are extremely easy on the hardness scale, COM3 and COM4 are extremely hard, and I am not going to try COM5 or above unless somebody pays me to do it.

I have three working serial ports on my AT-compatible H-241 and soon will have the same number on my XT-compatible H-158: COM1 on each computer is the RS-232 port on the i/o board that came in the machine, COM2 is or will be an RS-232 port on a Boca Research i/o board, and COM3 is the RJ-11 jack on Zoom Telephonics *Zoom/MODEM PC-2400* in my H-241 and will be the RJ-11 jack on the *Zoom/MODEM PC-1200* in my H-248. All three COM ports are usable.

They are usable because none of that hardware conflicts with any of the other hardware, and because I try to select my software to work with my hardware. If you don't do the same, you will waste a lot of time and probably a lot of money unless you have the skill and knowledge to tinker things into compatibility. Remember that although the addresses for COM1 and COM2 are standard, there are no standard addresses for any other COM ports. If you want more than two COM ports, you'll have to get at least some boards on which you can change the addressing, software you can adapt to recognize that addressing, and the knowledge to program your hardware and software so that what you do does not conflict with anything else in the system. Prepare to pay someone else to do that if you can't: it's skilled work.

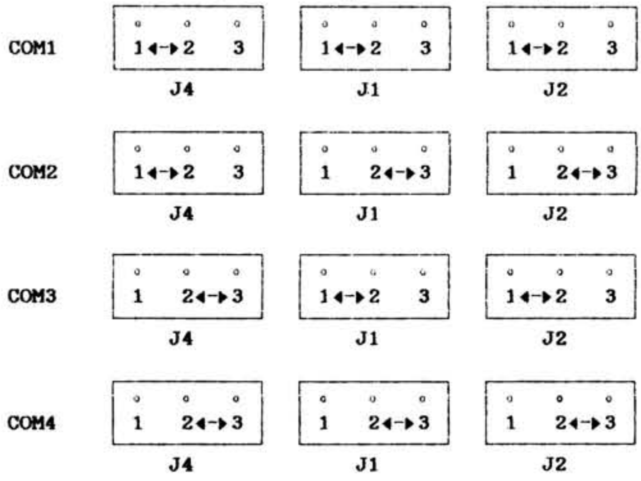
The hardware key to COM3 or COM4 in my system is the *Zoom/MODEM PC*. It is an internal modem that can be configured for addressing as COM1, COM2, COM3, or COM4. (If you use an external modem, you'll have to find an i/o board that allows such flexibility.) The configuration is easily done with three jumper banks labelled "J1," "J2," and "J4." Banks J1 and J2 work together to select between COM1 and COM2 or COM3 and COM4. Bank J4 determines which pair of COM ports J1 and J2 select between. If you have the jumper on J4 over its Pin 1 and Pin 2, then J1 and J2 select between COM1 and COM2. If you have the jumper on J4 over its Pin 2 and Pin 3, then J1 and J2 select between COM3 and COM4. You must set J1 and J2 both the same way. That relationship and all possible settings are shown in Section A of the charts to this column, "Setting the Zoom/MODEM PC." (The charts were made with Analytics International's *MonoGrafx*, which I've talked about before and can't praise enough.) Note that the pin numbering in the illustrations is for reference only: the numbers are not printed on the circuit board. The charts assume that you hold the board so its speaker is in the upper left-hand corner.

Since true compatibles are shipped with one serial port that is addressed as COM1, the *Zoom/MODEM PC* is jumpered at the factory for installation as COM2. You can change the *Zoom/MODEM PC* to COM3 or COM4 by moving the jumpers as I've indicated in the illustrations. Do not change it to COM1, because then there is a conflict with the existing COM1. Be extremely careful to avoid such conflicts whenever you add a board. For example, since the *BocaIO* boards can be configured as COM1 or COM2, I configure them as COM2. The *Zoom/MODEM PC* then can be COM3 or COM4. It doesn't matter which. Murphy's Law rules my office, so I've selected COM3 because it's easier for me to remember the sequence 1-2-3 than 1-2-4.

Once the hardware is properly configured, you'll need software to work with it. As I've said, all the commercial and public domain communications software I've seen can use COM1 or COM2. One of the many things I like about the *Zoom/MODEM PC* — and one of the many reasons I've settled on it — is that all the communications software I've tried that can recognize COM3 and COM4 will recognize those ports on the *Zoom/MODEM PC*. *Crosstalk XVI* version 3.61 need only be told the port to use: COM3 or COM4. *HyperACCESS* version 3.10 needs that information followed by the interrupt level: "3 , 4" for COM3; "4 , 3" for COM4. *Mirror* 3.6 requires the port number, its address, and its interrupt vector: "3 03E8 30" for COM3; "4 02E8 2C" for COM4.

SETTING THE ZOOM/MODEM PC

A. Jumper settings for four COM ports



B. Addressing COM3 and COM4

	ADDRESS	VECTOR	INT
COM3	Ø3E8H	3ØH	4
COM4	Ø2E8H	2CH	3

If you use any other communications program, you're on your own. Those communications programs I've mentioned and *HUGMCP* are all I'm using right now. Each has its own special charms for me, and I am happy with them all. Since *HUGMCP* comes with source code, you can configure a version to use COM3 and COM4 instead of COM1 and COM2 if you have the programming skills and tools. You'll find the addresses, interrupt vectors, and interrupt levels you need in Section B of the charts accompanying this column. You'll also need a relatively-recent version of Microsoft's *Masm* and *Link*. If you can't do the work or don't have the tools, I know you'll be delighted with the software I've mentioned that does support COM3 and COM4 on the *Zoom/MODEM PC*.

Alpha Software's *Keyworks*

I think it was Mike Edwards or Kevin Hauser who suggested on CompuServe's HUG SIG that I take a look at Alpha Software's *Keyworks* as a substitute for Borland International's *SuperKey*. Mike or Kevin or whichever of the merry pranksters there it was who made that suggestion was quite right. I've been using *Keyworks* instead of *SuperKey* for several months. The only times I use *SuperKey* now are when I need its predefined macros for using *SideKick* as a simulated printing tape calculator.

Keyworks does so much more than *SideKick* that it's really a keyboard enhancer and not merely a way to have one keystroke do the work of several. In addition to doing that very nicely, it will do things like set up a printer or modem, encrypt and decrypt files, control the "lock" keys (CAP, SCROLL, and NUM), do a screen dump (the equivalent of pressing the PRT SC key), and blank the monitor after a specified period of keyboard inactivity.

Once you realize all the things *Keyworks* will do, it should strike you as an ideal way to set up the computer so a novice could operate it from menus. That's easy: *Keyworks* itself is operated through popup menus and provides sensible ways to design your own for just about any style or kind of operation. You can even make help screens available at the press of a key.

Keyworks is a well-designed package with plenty of sample macros that suggest its possible uses with various kinds of applications programs including word processing, spreadsheets, database managers, and the Microsoft C compiler. One thing I particularly like is that there seem to be fewer conflicts with other TSR (Terminate and Stay Resident) programs when I use *Keyworks*. If I can figure out how to do a *Keyworks* macro to use *SideKick* as a printing tape calculator, I probably won't use the current version of *SuperKey* anymore.

TurboPower Software's *TSRCOM*

TSR (Terminate and Stay Resident) programs are useful nuisances. They help simulate multitasking on microcomputers that otherwise can't do it, but TSRs can create such awful problems that many knowledgeable microcomputer users won't have anything to do with them. I'm fascinated by TSRs and use those I've learned to trust, but since I know the potentially-vicious bite to these things I've developed a dependence on Kim Kokkonen's TurboPower Software toolkit for handling TSRs. Don't use any TSR until you get Kokkonen's *TSRCOM* and use it first.

Take a "good" TSR like Writing Consultants' *Word Finder*, which is an available-on-demand thesaurus. (The latest version of *Word Finder*, incidentally, claims to have 220,000 synonyms.) Executing any TSR merely loads it into RAM (Random Access Memory), where it attaches to one or more interrupts (triggers built into MS-DOS for controlling each part of the hardware) to lurk. *Word Finder*, for example, watches the keyboard interrupt (INT 16) for when you press the CTRL key and F6 key simultaneously. That combination is the so-called "hot key." In the meantime you go about business as usual, which in the case of *Word Finder* usually is word processing. So you execute a word processing program such as *WordStar* or *XyWrite III* and, because it's a normal transient program (not a TSR), it runs immediately on execution and you start mincing words. Press CTRL-F6 when you've typed one for which you'd like to consider substitutes, and *Word Finder* overlaps your *WordStar* display with a list of alternatives from which you can select with another keypress.

Word Finder is a likeable TSR because once installed for the word processing program with which it should work, it actually does work and, in my experience, it works well. Some combinations of TSR and transient clash. Such a clash can mean little more than the TSR stealing an interrupt so it doesn't get to the transient. You're fortunate then because the only problem is that things don't work right. But the clash also can mean an unresolved battle that completely paralyzes the system. It locks and you have to reboot, usually (unless you have installed a reset button on the computer) by switching the machine off and then on again. I've heard that sometimes the brawl gets out of hand and trashes a disk, but I haven't experienced any such disgraceful behavior. I do worry, though, when trying out a new TSR or combination, and I save my data files more frequently than usual. And, as I began by saying, I always use *TSRCOM*.

It's free. You'll find it on bulletin board systems and on CompuServe's IBM Software SIG, usually listed as something like "TSRCOM.ARC": I'm sorry but, like much public domain software, the package name will vary according to the whim of whoever uploaded it to the system. The latest version I know is version 2.10, in an archive containing seven files: TSR.DOC, MARK.COM, RELEASE.COM, RAMFREE.COM, MAPMEM.COM, FMARK.COM, and TEST.MRK. The DOC file is of course an instruction manual, and you should read it carefully for the package's advanced features.

Basically, though, you run *Mark* before executing any TSR to mark the place where it begins. Then, when you want to unload the TSR from RAM, you run *Release* to free memory up to that mark. Anything after the mark gets unloaded. If you live dangerously by using more than one TSR at a time, it's an awfully good idea to run a mark sandwich: run Mark, run the first TSR, run *Mark* again, run the second TSR, and so on. When you run *Release* in such a case, it chops its way backwards: run *Release* to unload everything after the last mark, run *Release* again to unload everything after the preceding mark, and so on. Each mark takes up 1584 bytes of RAM. If you can't spare even so little, *Fmark* reduces consumption to 150 bytes by using a disk file you specify to hold information *Mark* keeps in RAM. TEST.MRK is an example of such a file.

The rest of the package is extremely valuable utilities for programmers and other tinkers. *Ramfree* says quickly how much RAM is free right then. *Mapmem* shows a memory map with the names of everything in RAM, how much each thing is using, and which interrupts they attach. *Eatmem* wastes memory — intentionally — for times when available RAM must be reduced. I use it mainly to test the lower limits under which software will operate. If you don't require intimacy with your system, you shouldn't need these things. Nevertheless, if you use any kind of TSR you must have *Mark* and *Release* from *TSRCOM*.

It's a remarkable programming achievement. One reason why Kokkonen dedicated it to the public domain is to demonstrate TurboPower Software's abilities. He has. They evidently are considerable.

An Apple For The Teacher

Our Apple *LaserWriter* printer arrived right before the end of 1986, while I was writing you this, and just under the wire for the capital investment credit and other benefits abolished under the new tax law. We're excited. I'm ecstatic. If you've been keeping even one eye on laser printers, you'll know that at the current list price of $5,000 the Apple *LaserWriter* costs more than most laser printers for personal computers.

So how come? The reason is *PostScript*, Adobe's page description language. It's available on a range of output devices including those that do typeset quality. Typesetting on a microcomputer is one of my major interests and has been since my first sight of a microcomputer. That took imagination back then: the first one I saw was a Radio Shack TRS-80 Model 1. At any rate, I'm not much interested in using a laser printer only as a quick and quiet letter-quality printer: I already have a slow and noisy daisy wheel that I love.

My idea is to use this laser printer two ways: first, to make camera-ready copy for medium-quality jobs in which near-typesetting will do; second, to make proofs of files that can drive a real typesetting machine for high-quality jobs in which only real typesetting will do.

The software I have that can do both right now are the two ports of *TeX* available for MS-DOS microcomputers: *PC-TeX* from Personal TeX and *Micro_TeX* from Addison-Wesley. You heard me right. I said "MS-DOS microcomputers," *not* "IBM-compatible microcomputers." Of course these two implementations work — and work splendidly — on compatibles such as the H/Z-158 and H/Z-241, but they also work — and work splendidly — on so-called "generic" MS-DOS microcomputers like the H/Z-100. I'll have much more to say about *TeX* as time goes by. There's enough to say about *TeX*, and *PostScript*, and the Apple *LaserWriter* to warrant several articles. I'll do them. And, with your permission, I'll be saying more things here in my pilgrim's progress towards digital nirvana.

You're too polite to ask the question direct, but I'll answer it anyway. No, I don't feel the least bit awkward about having an Apple *LaserWriter* connected to my Heath and Zenith computers. I like fine things and I don't like silly computer wars. Both inclinations helped steer me towards Heath and Zenith equipment, and both keep me here. I do feel that the *LaserWriter* costs more than it should, but since John Sculley did not have the advantage of consulting with me I have to forgive his gaffe. With the present hot competition among makers of laser printers, including *PostScript* laser printers, I expect Apple to do things with the price of the *LaserWriter* around the time you read this. I bought when I did because our tax situation made it worthwhile to buy when I did. I was interested in a QMS *PS-800*, the *LaserWriter*'s direct competition, but at the last minute QMS played a game with me that stood my hair on end and immediately decided me to back away. The *LaserWriter* it is then. I don't think anyone at Apple seriously believed it would wind up at the other end of a Heath/Zenith computer so I had to work out the interface by myself. I'll tell you how, later.

Right now I'm busily typesetting everything in the world. Hugh Kenner wrote some questions about C programming right at the time I had run out of stationery. A couple of minutes with *TeX* and ArborText's *DVILASER/PS* driver for *PostScript* printers gave me my letterhead. I was tempted to typeset the entire letter (not a difficult job once you get the hang of it) but decided against looking completely mad. So I "settled" on using Microsoft's *Word* (which has a couple of nifty *PostScript* drivers) and wound up with a letter that looked better than it read. It actually looked too good. I can't imagine what a student will think about getting typeset comments about a paper. Well, I guess I'll find out. As I say, I'm busily typesetting *everything*. Both Personal TeX and Addison-Wesley carry lines of drivers, macro programs, and other software that complement *TeX* for microcomputers. They even have "preview" programs for viewing what you've set before printing it. The one I've seen strikes me more as showing images and shadows of divine things than WYSIWYG (What You See Is What You Get), but with *TeX* that's probably acceptable: it gives you much, much finer typesetting than can possibly be displayed on current personal computer monitors. I can't wait to see what other software is available now, and what will be coming soon, for this marvellous combination of hardware.

Isn't life exciting?

Products Discussed

The RS-232 Line Protector (SK-202)	$14.95
The Traffic Cop (SK-204)	$39.95
The AC Monitor (SK-211)	$39.95

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

BocaIO/AT	$119.00
BocaIO/XT	$129.00
BocaRAM 1MB Add-on	$345.00

Boca Research, Inc.
6401 Congress Avenue
Boca Raton, FL 33431
(305) 997-6227

MonoGrafx (Version 3.0)	$99.95

Analytics International, Inc.
1365 Massachusetts Avenue
Arlington, MA 02174
(617) 641-0400

Zoom/MODEM PC 1200 ST	$299.00
Zoom/MODEM PC 1200 XL	$349.00
Zoom/MODEM PC 2400 ST	$499.00
Zoom/MODEM PC 2400 XL	$549.00

Zoom Telephonics, Inc.
207 South Street
Boston, MA 02111
(617) 423-1072

Keyworks
Alpha Software Corporation
30 B Street
Burlington, MA 01803
(617) 229-2924

Crosstalk XVI (Version 3.61)	$195.00

Microstuf, Inc.
1000 Holcomb Woods Parkway
Roswell, GA 30076
(404) 998-7798

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

HyperACCESS (Version 3.10)	$149

Hilgraeve, Inc.
P.O. Box 941
Monroe, MI 48161
(313) 243-0526

Word Finder (Version 3.3)	$79.95

Writing Consultants
Techniplex Center
300 Main Street
East Rochester, NY 14445
(716) 377-0130

TSRCOM (Version 2.1) public domain (Free if downloaded from a BBS. I have no other availability information, but suggest you accompany any request for it from the publisher with a check for $25.)
TurboPower Software
3109 Scotts Valley Drive #122
Scotts Valley, CA 95066
(408) 448-8608

PC-TeX (Version 1.50)	$249.00

Personal TeX, Inc.
20 Sunnyside Avenue, Suite H
Mill Valley, CA 94941
(415) 388-8853

Micro TeX (Version 1.50A) $295.00
DVILASER/PS $300.00
Addison-Wesley Publishing Company
Reading, MA 01867
(617) 944-6795

*

In the December 1986 Issue of REMark, in the article "Paper Bits: A Review Of Softstrips On Zenith Computers", the phone number of Cauzin Systems, Inc. was wrong. The correct number should be (203) 573-0150.

Nashua.
The leading name in coating technologies since 1849
Nashua
Professional Magnetic Media
MD2D
Ten 5¼ Diskettes
TDK
SUPER AVILYN
HS
120
VHS
FREE

WHAT A DEAL!

[illegible] ua MD2D High [illegible] ided double- [illegible] y 5¼" diskettes at the special price of \$39⁹⁵ *and* receive a TDK VHS T120-HS High Standard video tape FREE – a \$7⁹⁵ Value.

Remember: all Nashua diskettes carry a lifetime replacement warranty!

Offer good at all Heath/Zenith Computers & Electronics Stores.

"It's how we make them that makes them best."

Heath/Zenith Users' Group

Hilltop Road
Saint Joseph, Michigan 49085