REMark®

Volume 8, Issue 3 • March 1987
P/N 885-2086 Issue 86
Official magazine for users of HEATH ZENITH computer equipment.
$2.50

More Goodies For Your Lightpen
See Pages 8 & 41

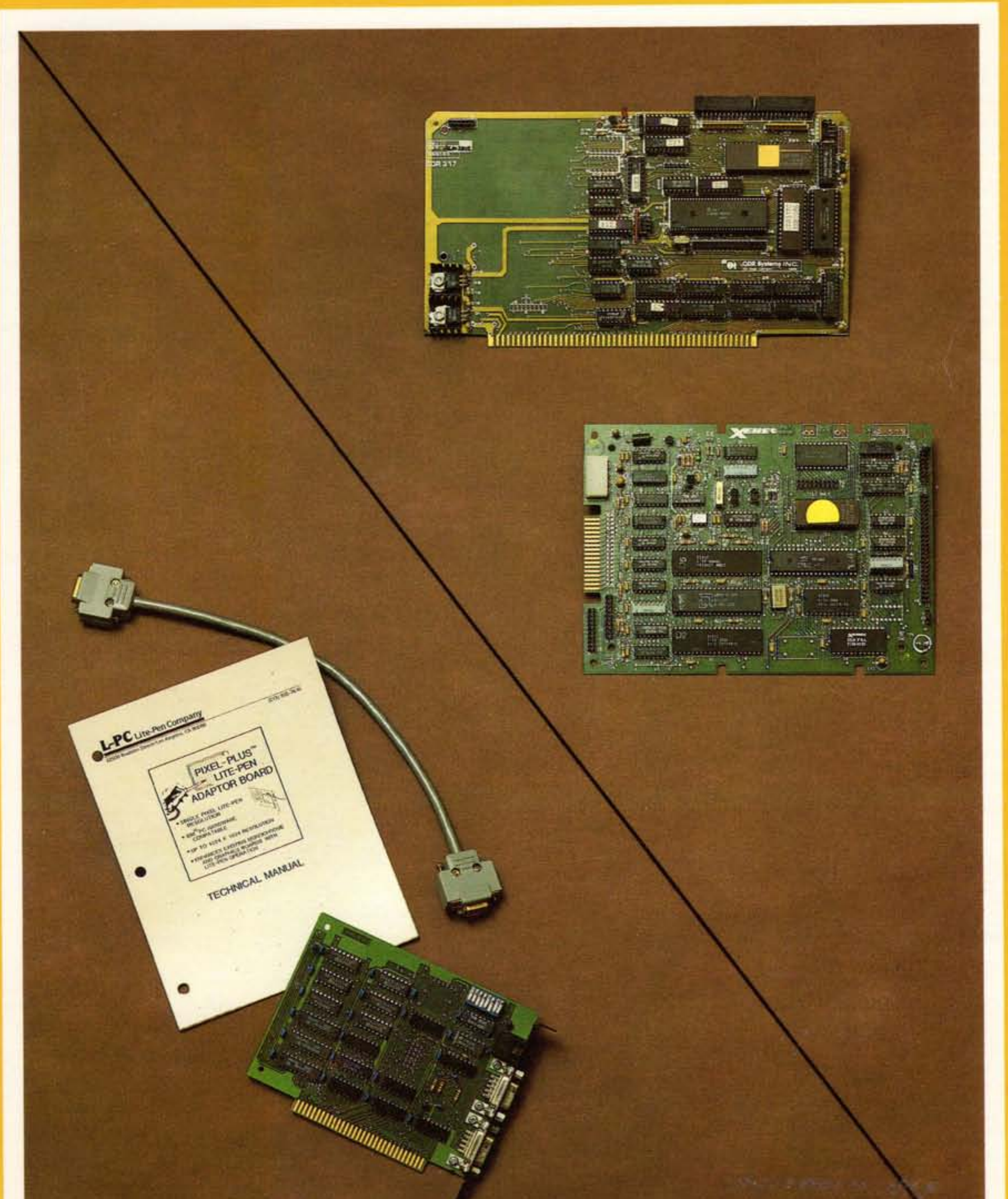


Photo by Dan Wilson, Heath Graphics Design

QUIKDATA, INC.

1977 ONE DECADE OF SERVICE 1987

TO THE HEATH/ZENITH COMPUTER WORLD

That's right! We've been here for **10 YEARS** serving the Heath/Zenith computer world. We started with the H8, and have continued on through the H89/90, H/Z100 series, the H/Z PC computers, PC laptops, and the Z240 -AT series.

If you're a newcomer, you may not have heard of Quikdata. This is our first advertisement in almost two years. We depend mainly on word of mouth to spread our business, because that's the best advertising you can get - it also helps us keep our prices to a rock bottom low, while providing you with full service and support. We felt it was time to get our name out again for those of you who may have the misfortune of not knowing of **the mail order service who cares!**

What made us a leader in supplying H/Z users in the past 10 years? * Lowest Prices * 24 hour shipping on 99% of all orders * We are familiar with the products we sell * Most every item sold is used in-house * We stand behind everything we sell * Software and hardware consultation * Systems put together the way you want * On-site repair facilities * We **stock** what is in our catalog * Support the H/Z line of computers * We treat you as an individual * Full stock of repair and replacement parts * H-SCOOP newsletter to help you get the most from your system

MEMORY AND STORAGE DEVICES

M41256 150ns 256K DRAM CHIPS	$2.79
M41256F 120ns 256k CHIPS	$2.95
M41256T 100ns 256K CHIPS	$3.49

Z150W20 21 MEG WINCHESTER FOR PC'S. This unit includes Segate ST-225 21 meg half-height winchester drive, OMTI controller, cables, software, mounting hardware where applicable. Complete kit with PREPped winchester ready to install and go.
Z150W20 $419

Z150MP memory mapping PAL for use in H/Z150/160 systems to allow up to 704K of system RAM on main memory board. Requires 18 M41256 memory chips plus up to three banks (27) of the 64K M4164 chips. $19

MEGARAM-150 for the H/Z160 computers allows up to 704K system memory and 512K RAM drive memory. Software included. Uses up to 5 banks (45) of the M41256 memory chips.
MEGARAM $43

BARE WINCHESTER DRIVES

ST-225 half height 21 meg 80ms	$319
ST-4026 full voice coil 21meg 37ms	$595
ST-4038 full voice coil 32meg 37ms	$639
ST-4051 full voice coil 40meg 37ms	$749

FLOPPY DRIVES

SA-455 half-ht DS DD 360K	$129
TM100-2 Tandon full DS DD 360K	$129
SIEM-R SS Siemens refurbs	$ 59
TM100-4R Tandon DS DD 96TPI refurb	$ 95
M4853 Mitsubishi half-ht 96 TPI	$139

COMPUTERS

We carry the full line of Zenith computers, but the prices are too low to list here! If you are interested in a system, call or write for a complete system quote. Tell us what you want in the system and we'll be glad to quote you a price. To the basic systems we can add additional memory, drives, winchesters, whatever you need. All systems come with MS-DOS 3.x operating system.

H8/H89 STUFF

H37 SOFT SECTOR CONTROLLER BOARD	$195
WH8-37 H8 SOFT SECTOR CONTROLLER	$295
VIDEO BLANKER	$ 21
AUTO REPEAT (kit $33), assembled	$ 43
16K H89 MEMORY EXPANSION BOARD	$ 49
AS-IS H89 CPU & TLB BOARDS	$ 39
REFURB WORKING H89 CPU/TLB BOARDS	$ 79

21 MEG WINCHESTER SUBSYSTEM (requires HDOS/CP/M software package for $95 and host adapter card. H8 uses WH8-37 and H89 can use Z67 or ADD320).

H20	$649
ADD320 SASI winchester/3 serial ports	$145

We also have keyboards, H89 power supplies and parts, transformers, video boards, cables, H17 disk controllers, serial I/O boards, memory chips, diskettes, various H8 boards and much much more.

PANASONIC PRINTERS

KXP-1081i 80 column 120 cps F/T parallel	$259
KXP-1092 80 column 180cps F/T parallel	$395
KXP-1592 132 column 180cps F/T parallel	$495
KXP-3151 daisy Diablo 630 compatible	$469

DISKETTES

3740/2D	8" DS DD SOFT	$29
3740/1	8" SS SD SOFT	$25
204/2D	5" DT DS DD SOFT	$37
207/2D	5" DT DS DD HARD	$37
104/2D	5" ST DS DD SOFT	$25
107/2D	5" ST DS DD HARD	$28
5H-11 Sentinel	5" ST SS SD 10-HOLE HARD	$19
BROWN	UHR for -AT's	$37
UHRII	Dysan UHR for AT's	$47
X35	Xidex/Dysan ds 3 1/2"	$22
352D	Dysan high end 3 1/2"	$35

We carry a full line of modems, disk drive and winchester drive cabinets and cables, software, hardware, surge and spike suppressors, line conditioners, backup power supplies, serial printer interfaces, print buffers, and more.

Call or write for a **free no obligation catalog** of our product line and/or a sample issue of our H-SCOOP H/Z newsletter. VISA and MC accepted. All prices are CASH prices - PO's add 5% and pick up 3% S&H. Charge cards pick up 3% S&H. Cash orders over $100 shipping on us. All orders under $100 pay $4 S&H.

QUIKDATA, INC.

2618 Penn Cir
Sheboygan, WI 53081
(414) 452-4172

YOU'VE GOT THE BEST!

NOW GET THE REST... H-SCOOP

If you're reading this advertisement, you probably own one or more Heath/Zenith microcomputers. The Heath/Zenith computers are the best buy in price/performance systems available. Now that you've got the best, get the rest. You need to keep up with the latest information as it applies to your H/Z system to help you **get the most from your system**. That's where **H-SCOOP** comes in.

H-SCOOP, the independent double density newsletter for H/Z computer support is probably the best asset you and your system can enjoy. You see, we've been supporting only H/Z systems since before Zenith was ever part of the whole scenario. In fact, we've been in the H/Z support business ever since the first H8's came rolling off the assembly line - 1977. That gives us one decade of first hand experience - more than anyone else! The first issue of the newsletter was mailed FIRST CLASS MAIL April 1980; it has continued each and every month like clockwork ever since.

Why **H-SCOOP**? Because **H-SCOOP** is the **only** H/Z related publication that **dares** to be different! Since we are not affiliated with Heath or Zenith, we **tell it like it is**. Keep in touch with the latest developments from Heath, Zenith and other independent vendors. The latest in hardware and software. The best prices. Know the vendors you can feel confident to deal with - and those to stay away from. Keep in touch with those having similar interests as yourself, and know who is doing what with their systems. Have a problem or need some advice? Use the REQUEST column. Want to sell or buy something? CLASSIFIEDS gets results. Reports. Reviews. Good guys & Bad guys - who are they? Who can you trust to deal with? Tips, ideas, problems, solutions - all part of **H-SCOOP**.

But let's be fair and let the subscribers speak. The following are unsolicited letters from some of our subscribers:

"As always, thanks for making this kind of information available. Your service is invaluable, not only for those of us overseas, but to anyone who wants to get the greatest service from his or her Heath/Zenith. Keep it up!" D. Shoemaker., APO NY.

"...Thanks very much for compiling and publishing such a useful, straightforward newsletter... I have certainly picked up enough information to repay the subscription price several times over." P. Herman, Canada.

"I would like to thank you for all that you do, it certainly is helpful to have someone like you in business that can provide our needs and at the same time give the necessary support. You certainly make my life better and easier thanks to your help. It is greatly appreciated." J. Waldron, Ohio.

"Your newsletter is refreshing and has been quite helpful to me as an owner of a Z-150 PC. I will continue to look forward to receiving it in the future." D. Howey, MD, Indiana.

"Please enter my subscription for your famous newsletter, H-SCOOP". C. Pitonza, New York.

"Enclosed is my renewal. My main difficulty with H-SCOOP is telling between very good, excellent and very excellent." A. French, LA.

"I'd like you to know how much I don't take what you do for granted. I very much appreciate your printing efforts..." E. Nitzke, MD, Iowa.

"Of all of the publications that I receive concerning computers I guess I enjoy yours most of all. First of all it's understandable, second, it tells it like it is..." Atty M. Aukerman, Indiana.

"As a satisfied customer of your products and subscriber of your newsletter I feel a word of appreciation and thanks is well overdue. I have experienced the degree of quality that keeps any human being content in being a member of the H/Z family. Thanks for your energy and sincere effort!" F. Scott, Maryland.

"Thank you for the wonderful and interesting reading in H-SCOOP for the past year. To say the least, it has been very enlightening and informative... Please keep up the outstanding work as all Heath/Zenith users can use your fine product." J. Minsterl, Virginia.

"I find the newsletter very informative and one of the best news letters for the Heath/Zenith user that I have seen." M. Stueber, Michigan.

"Because of an article you wrote some issues back, I was able to get my printer ... to come up in both HDOS and CP/M. Your advice on software and Good guys and Bad guys has been very helpful." L Becker, AL.

"The newsletter has made a steady improvement over the years. You contribute a lot to the usefulness of my computer and stop me from reinventing the wheel over and over again." T. Tverre, Norway.

"I have enjoyed the newsletter very much, especially because of your style of brevity and telling it like you see it. I get more out of one issue of your newsletter... than in the combined issues of ..." R. Tibbetts, CA.

"The newsletter, called H-SCOOP has enough gossip and opinionated evaluations to be worthwhile to anyone who uses an IBM PC or compatible. It is a must for Heath/Zenith users." John Dvorak, May 27, 1985 issue of InfoWorld

Seen enough? It's your move! Test ride it for a year. Rates: $24/year US and Canada, first class mail, $32 overseas AIRMAIL. Back issues available. **Special back issue deal:** Start the current year with the January '87 issue (expires with the December '87 issue) and receive all available back issues dated before January '87 for an additional $20 (US prices). VISA & MC accepted.

H-SCOOP

2618 Penn Cir
Sheboygan, WI 53081
(414) 452-4172

Heath/Zenith Users' Group

REMark®

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

Official magazine for users of HEATH ZENITH computer equipment.

Volume 8, Issue 3 • March 1987

Index of Advertisers

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

Reader Service No.		Page No.
126	Al Davis	73
101	American Cryptronics	58
102	Analytical Products	23,28
127	Bersearch Information Services	51
128	Cometsoft Technology	51
104	FBE Research	58
105	First Capitol Computer	70
129	Generic Computer Products	79
130	H-Scoop	3
106	HUG PBBS	10
107	Paul F. Herman	39
108	Hogware	79
109	Hughes Development Systems	57
111	KEA Systems	58
113	Kit Builders Journal	16
130	Quickdata, Inc.	2
116	PMI	40
117	Payload Computer Services	52
132	RAM Technology	82
118	Rybs Electronics	33
119	S&K Technology, Inc.	60
120	Santa Cruz Operations	6
121	Scottie Systems	60
122	Secured Computer Systems	51
123	Software Appl. of Wichita	39
105	Software Wizardry	42
124	Spectre Technologies Inc.	34
134	Systems Peripherals Consult.	28
131	Veritechnology Elec. Corp.	24
133	Barry A. Watzman	84

On The Cover: *Do you have an H/Z-100? Thinking of installing a hard disk system? Check out the CDR317 Hard Disk Controller. For a review of this package see Page 13. For PC users, the Pixel-Plus(tm) will provide superior resolution and accuracy with computer graphics and enables you to use your lightpen one pixel at a time. See Page 41 for details.*

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 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 applications available for all popular UNIX/XENIX systems.

SCO
THE SANTA CRUZ OPERATION

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

Rotating HUGCON

If you're having trouble deciding which HUG Conference to attend this year, your mind may have already been made up for you. From all indications, the International HUG Conference in Chicago this year may be the ONLY conference to attend! For various reasons, the West Coast HUG Conference, and the Midwest HUG Conference will NOT take place this year. The Capitol Heath Users' Group has indicated that "there may not be a CHUGCON 87" due to a lack of volunteers.

Putting on a conference for a group such as ours is an exciting and personally rewarding experience. Literally thousands of tasks must be finished before the final day, and usually a hundred are forgotten about. These 'weekend shows' can sometimes demand more time and effort than the average person has to give. But, somehow, each conference always turns out for the best.

I've always felt that holding the National HUGCON in the same place every year was somewhat unfair to those people living at either end of the country. So, I would like to propose the following idea. I feel we should have a 'rotating HUGCON'. As an example, consider the following. This year, have HUGCON in Chicago as already planned. In 1988, have it on the east coast. In 1989, the west coast. In 1990, hold it in the south somewhere. Then finally in 1991, we'd be back in the midwest. By moving HUGCON in this manner, I think we would give everyone around the country a more reasonable chance to attend one our national conferences. In addition, I think the possibility of 'burn-out' would get eliminated since it would be 3 to 4 years before we would be back in the same general location. We'd like to involve the local HUG groups in these 'rotating' conferences. Since they're essentially where 'the action' would be, they would be of great assistance in relaying information regarding conference sites, local users to give seminars, hotel information, and general assistance the days of the conference. I really would like to hear your thoughts on this idea and the thoughts and ideas from the different HUG groups around the country. Bring it up at your next meeting. Drop me a line, or if you like, give me a call.

At this point, I would also like to dispel a bit of confusion. Rumor has it that REMark is geared only toward H/Z-89 and H/Z-100 users. Not true! We have taken extra steps to assure that the PC compatible line of computers gets its fair share of coverage in every issue. Over 1986, 35% of the computer hardware articles were PC related, 49% were H/Z-100, and 16% were H-8/H/Z-89. Remember, REMark also reflects your input. If you think there's a topic we should cover, why not write about it and submit it yourself? Researching a topic can be a tremendous learning, as well as rewarding experience. Along with first hand knowledge about your topic, you'll earn Heath/Zenith software products or free HUG BUCKs. For more information about submitting articles, check out the January issue of REMark.

Jim

Jim Buszkiew

What Else Can I Do With My Light Pen?

Robert F. Doolittle, Ph.D.
RD SOFTWARE
1290 Monument Street
Pacific Palisades, CA 90272

About The Author

Dr. Robert F. Doolittle has been a dyed-in-the-wool hacker since 1976. At that time he was still working as a high-energy astro-physicist at TRW Systems in Los Angeles. The lure of computer programming finally took over and in 1983, after 25 years with TRW, he took early retirement to pursue his computer interests full time. He has been writing and selling CP/M utilities for Z-80 systems since 1979 under his own business name, RD SOFTWARE. In addition, he currently works part time at ASHTON-TATE as a systems analyst.

Introduction

In August 1986, HUG released my software calculator simulation called MSDOS CALC (HUG P/N 885-8043-37) for the Z-100. It is described in the September 1986 issue of REMark magazine. The calculator buttons are pushed on the screen by a light pen. Included with this software package is a coupon good for a $114.98 discount off the standard retail price of a light pen from Lite-Pen Company in Los Angeles. A number of customers who bought CALC and redeemed the coupon for the pen have since written or called me asking the question raised in the title of this article: 'Your program works fine, but what else can I do with my light pen'. In this article I will attempt to present some answers to that question by discussing a light pen driver which may be called from any high level language.

Among the many superior features of the H/Z-100 (not PC) series of computers is a hardware port for a light pen and low level software drivers in the BIOS (IO.SYS) for generating light pen position and status, returned via two BIOS interrupts to the user.

These interrupts (INT 54H and INT 5BH) are documented in the MS-DOS Programmer's Utility Pack. Any applications program which intends to accept a light pen input must redirect these interrupts to a user written interrupt service routine. The exact nature of the service routine will depend on the application. I will present a light pen driver in this article which can then be called from a BASIC applications program. The changes required to call these routines from other high-level languages are minor and will be pointed out where appropriate.

A light pen operates by detecting light from your CRT. It is important to remember that nothing will happen if your pen is not over a point on the screen which is lighted. The CRT controller chip in your computer will store in its internal registers information which is essentially the time difference from the start of a video frame to the point where the pen signal occurs. This, along with other CRT controller data and software, makes it possible to reconstruct either screen character position or pixel and scan line information. Your screen normally accommodates 640 horizontal pixels (0-639) and 225 vertical scan lines (0-224). A character is 8 pixels wide by 9 scan lines vertically. The screen is thus 80 characters wide by 25 rows. A complete description of the Video Logic Board in your computer is given in Volume I of the Z-100 Technical Manual.

Interrupt 54H

Whenever a light pen signal (trigger) occurs, the BIOS generates interrupt 54H. When this happens the BX register in the 8088 CPU is set to the character position corresponding to the pen location at that time. This is a number from zero to 1999. If this number is divided by 80, the quotient will be the character row and the remainder will be the character column. Here we label rows 0-24 and columns 0-79. In addition, this interrupt also provides, in the AL register, the pixel and scan line information. The most significant four bits of the 8-bit byte in AL gives the scan line within the character (0-8). The least significant three bits gives the pixel within that scan line (0-7). (Remember, characters are 8 pixels wide by 9 scan lines tall.) It should be pointed out that the horizontal accuracy is not very precise, especially at the extreme left and right of the screen. Some constant correction is usually required anyway and attempts to use the pen at these edge locations should be avoided.

Interrupt 5BH

This is called the Special Light Pen Interrupt in the Programmer's Utility Pack, but is actually the status of your light pen switch. The light pen from the Lite-Pen Company, which is sold for CALC, has the switch spring-loaded in the tip. The switch is closed by pressing the pen tip against the screen. Some light pens have a push button switch mounted on the body but they both serve the same purpose. This interrupt is generated when a switch transition occurs. The status is in CPU register AL. If AL equals 0, the pen was lifted off the screen (switch opened). If AL equals 1, the pen was pushed on the screen (switch closed). If this status were not available, the pen would continuously generate an output which could not be controlled by the user as a single shot event. Sometimes this mode may be desirable. It is possible to use the pen in this mode as a real pen to draw continuous pictures on the screen. The status is still used but in a manner such that the pen stays active as long as the switch is closed. Remember that the pen must still see light, so that it is not possible to draw white on black for example. You must have some kind of full screen background to draw or paint in this mode. We will discuss only the single shot mode, as used with CALC, in the remainder of this article.

The driver presented here has three entry points. They are called INIT, RESTR, and DOINT. INIT initializes (re-vectors) the two interrupts to our interrupt service routines. RESTR restores the two interrupts to their previous values. DOINT is the procedure called by the applications program to obtain pen data. See Listing 1 for the complete driver code. I call this file DRIVER.ASM. This driver also includes the interrupt service routines themselves. All of this is in a single file which, when assembled, can be linked with a compiled BASIC program or other language compilers. Minor modifications will probably be necessary depending on the calling conventions of other high level languages. Procedures will also be discussed for using this driver with BASIC running as an interpreter. Once the driver is installed and the application is running, a call is made to INIT. Then whenever the pen position is desired, a call is made to DOINT. Finally, before your application exits, a call must be made to RESTR.

The INIT And RESTR Procedures

Our BIOS provides an entry point for redirecting and restoring interrupts 50H-5BH. It is called BIOS__INTFUNC. The entry point is in the BIOS segment, 40H, and its offset is 45H. This address is defined as a double word in the driver at BINTF. To use the entry point, the ES register is set to the segment of the interrupt service routine and BX points to its offset. Register AL is loaded with the interrupt number. If we are redirecting, AH=0 and if we are restoring AH=1. Then a far call is made to BIOS__INTFUNC. The interrupt service routines must exit with a far return. The actual interrupt return is handled by the BIOS.

INT54 Service Routine

Procedure INT54 is somewhat application dependent. Here we have ignored the pixel and scan line information in AL. If needed, it could also be returned. Also, the position information in BX could be returned directly and used for whatever purpose by the application. Our example does the row and column calculation at this point and makes an empirical correction to the position. At the same time this 'correction' factor normalizes the row numbers to go from 1 to 25 and the column numbers from 1 to 80. It then stores these data as a 16-bit word in the variable POS. The high order byte is the row number and the low order byte is the column number. This service routine is entered whenever the pen 'sees' light and independent of the switch position.

INT5B Service Routine

This procedure is not necessarily application dependent. It simply stores the previous status at PSTO and the current status at PST. The application dependency is in DOINT which looks at these variables as described below.

Both interrupt service routines do their thing automatically and you need not be further concerned with them. They are also independent of the high level language in which the application is written. Again, the application and language dependency is in DOINT.

The DOINT Procedure

This procedure is the one actually called to return pen position data to your application. Your BASIC program would have a statement such as CALL DOINT(A) where, upon return, A would contain the row and column position at the point of the pen switch press. A BASIC statement such as X=A MOD 256 and Y=A\256 would store the row in Y and the column in X. It is important that A be declared to be an integer with either a DEFINT statement or by naming it A%.

If you prefer to get the row and column directly, then a two argument call should be made (CALL DOINT(X,Y)). Listing 2 shows how DOINT should be rewritten for this case.

DOINT starts by saving the base pointer (BP). This is not necessary for BASIC, but it is for most other languages. Other registers may also have to be saved when making assembly language calls from other languages. After saving BP, the stack frame is made accessible by moving the stack pointer to the base pointer. The pointer to the variable is now at BP+6. This pointer is moved to the SI register which now serves as a pointer to the integer variable itself in BASIC's data segment. Next BASIC's data segment (DS) is saved and the data segment register is loaded with our code segment. This makes our data directly accessible from our code segment. The next instruction tests the current pen status and returns -1 to the program if the pen is up. The next few instructions check to see if this was the first hit since the pen switch was closed. If not, a -1 is also returned. If it is a valid hit (first hit since switch was closed), the position word at POS is loaded to AX and returned to BASIC. This is done by first restoring BASIC's data segment register and then executing the instruction MOV [SI],AX. Finally, the base pointer is restored and we return to BASIC with a RET 2 instruction since BASIC requires that we pop the stored pointer off the stack. Again, other language implementations may differ.

Installing The Driver

1. With compiled BASIC or other compiled languages.

   For BASIC, simply compile your application. Other languages may require some kind of EXTERNAL statement for the three driver entry points. Next assemble the driver to create an .OBJ file. LINK these two to create the final .EXE file. The PUBLIC statement in the driver will tell the linker how to do this. Suppose your BASIC program is called PEN.BAS. Then the following commands would be given:

   ```
   BASCOM PEN;
   MASM DRIVER;
   LINK PEN DRIVER;
   ```

2. With the BASIC interpreter

   There are many ways of loading an assembly language program to be used when BASIC is running as an interpreter. I shall present my preferred method. As before, assemble the driver but this time ask the assembler to also create an .LST file. Look at the .LST file to find the entry point offsets for INIT, RESTR, and DOINT. Make a note of these. Next link it to get an .EXE file. Then run EXE2BIN to get a .BIN file.

"Now I Can Handle More Of You!"

Hi, my name is HUGPBBS, I know that many of my friends have tried to reach me, only to hear that annoying buzz... buzz... buzz... busy signal. It's hard to believe that I've become so popular in so short a period of time. Well, now I'm multi-user, and can respond to more of you! My human figured out a way to clone me, and now I'm not sure which one of me is the original! We're all still available 24 hours a day, now with over 20 megabytes of free software for downloading. Most of my software is for the PC Compatible computer, however I do have a lot to offer the H-8, H/Z-89, and H/Z-100 systems. I've got games, utilities, business, scientific, home finance, real estate, point-of-sales, graphic applications, and even HDOS version 3.0 just to give you an idea. In addition to all this software, users have been using my message base to exchange ideas, get online help, and even sell things. Remember, there's no charge for my service to HUG members, and I can be as slow (300 baud) or as fast (1200, 2400 baud) as you like. I'm a little touchy as to 'word' type, so make sure you're set to 8-data bits, NO-parity, and 1-stop bit. Call me at (616) 982-3956, type a carriage return several times to get my attention, and register your human. Just leave his first and last name, his HUG ID number, and a password (up to 16 characters). Although not as interesting to talk to, your human can call mine (Jim Buszkiewicz) at (616) 982-3837 and register in English.

The entry point addresses for DRIVER.ASM, as presented in Listing 1, are INIT=0BH, RESTR=28H, and DOINT=45H, where H stands for hex. We must now load this file somewhere in memory out of BASIC's way. I usually pick a segment near the top of available RAM, say 6000H for example. This corresponds to an absolute address of about 393K. Next load DEBUG. At the DEBUG prompt type NDRIVER.BIN and return. This gives the file name to DEBUG. Next type L6000:0 and return. The driver is now loaded at segment 6000H and offset 0. Now give the R command to DEBUG and note the value in the CX register. This is the length in HEX of the driver in memory. For DRIVER.ASM it is 99H. Now give the Q command to quit DEBUG. We will assume the above entry point addresses, length and segment address for the examples below. Finally, load your BASIC interpreter and type the program:

```
10 DEF SEG=&H6000
20 BSAVE "PEN.FIL",0,&H99
```

BSAVE expects a disk file name, an offset, and a length to save. RUN the program and the driver will be saved on disk under the name PEN.FIL. Use some other name if you prefer.

Any time you have an application for the light pen you must include the following statements, with appropriate line numbers, near the top of your program:

```
10 INIT=&HB
20 RESTR=&H28
30 DOINT=&H45
40 DEF SEG=&H6000
50 BLOAD "PEN.FIL"
```

Line 40 tells BASIC what segment to use for the CALLs and the BLOAD. If you want to load the driver at some other segment than the default segment under which it was BSAVEd, you need only put that segment address in line 40 and include the parameter, 0, after the file name in the BLOAD command. The driver will then be loaded at offset 0 in the new segment. The entry point offsets will be the same.

Listing 1

```
;*****************************************************;
;                                                     ;
;          LISTING 1 -- DRIVER.ASM                    ;
;                                                     ;
; This source has three callable procedures. These are ;
; INIT, RESTORE and DOINT.                            ;
;                                                     ;
; INIT initializes the lite pen interrupts.           ;
;                                                     ;
; DOINT gets the pen data and returns it to the program. ;
;                                                     ;
; RESTR restores the pen interrupts.                  ;
;                                                     ;
; It also contains the INT 54H and INT 5BH            ;
; interrupt service routines.                         ;
;                                                     ;
;*****************************************************;
;
CODE    SEGMENT
        ASSUME   CS:CODE,DS:CODE,ES:CODE
;
        PUBLIC   INIT,DOINT,RESTR          ;Only necessary when
                                            compiling
;
START:  JMP      INIT                      ;Jump over data
;
BINTF   DD       400045H                   ;BIOS INT_FUNC
;
POS     DW       0                         ;Pen position
                                            (Hi:row, Lo:col)
PST     DB       0                         ;Current pen status
PSTO    DB       0                         ;Old pen status
;
;******************************************;
; This procedure redirects the light pen   ;
; interrupts to our service routines.      ;
; BINTF expects address of service routine ;
; in ES:BX.                                ;
;******************************************;
;
INIT    PROC    FAR
        PUSH    ES                          ;Save extra segment
        MOV     AX,CS
        MOV     ES,AX                       ;Make extra segment=
                                             code segment
        MOV     AX,54H                      ;Code to set INT 54H
        MOV     BX,OFFSET INT54             ;Point BX at INT54
        CALL    CS:BINTF                    ;Set INT 54H
        MOV     AX,5BH                      ;Code to set INT 5BH
        MOV     BX,OFFSET INT5B             ;Point BX at INT5B
        CALL    CS:BINTF                    ;Set INT 5BH
        POP     ES                          ;Restore ES
        RET                                 ;Far return
INIT    ENDP
;
;******************************************;
; This procedure restores the light pen    ;
; interrupts.                              ;
;******************************************;
;
RESTR   PROC    FAR               ;Entry point for RESTR
        PUSH    ES                ;Save extra segment
        MOV     AX,CS
        MOV     ES,AX             ;Make extra segment=
                                   code segment
        MOV     AX,154H           ;Code to reset INT 54H
        MOV     BX,OFFSET INT54   ;Point BX at INT54
        CALL    CS:BINTF          ;Reset INT 54H
        MOV     AX,15BH           ;Code to reset INT5BH
        MOV     BX,OFFSET INT5B   ;Point BX at INT5B
        CALL    CS:BINTF          ;Reset INT 5BH
        POP     ES                ;Restore ES
        RET                       ;Far return
RESTR   ENDP
;
;******************************************;
; This procedure gets lite pen input       ;
; and returns character row,col in AX.     ;
; If pen up or still down after first      ;
; hit, then returns -1.                    ;
;******************************************;
;
DOINT   PROC    FAR
        PUSH    BP                ;Save base pointer
        MOV     BP,SP             ;Get stack frame
        MOV     SI,[BP+6]         ;SI is pointer to variable
        PUSH    DS                ;Save BASIC DS
        MOV     AX,CS
        MOV     DS,AX             ;Make data segment=
                                   code segment
        MOV     AL,PST            ;Get current status
        OR      AL,AL             ;Is pen up ?
        JZ      D01               ;Yes, return invalid
        MOV     AH,PSTO           ;Get old status
        AND     AL,AH             ;Is this first time ?
        JNZ     D01               ;No, return invalid
        MOV     AX,POS            ;Get pen position
        MOV     PSTO,1            ;Make sure old status set
        JMP     D02
D01:    MOV     AX,-1             ;Return -1 for pen up or
                                   still down
;
D02     POP     DS                ;Restore BASIC DS
        MOV     [SI],AX           ;Store position or -1
        POP     BP                ;Restore BP
        RET     2                 ;Pop one pointer and far
                                   return
DOINT   ENDP
;
```

```
;***********************************************;
; These are the pen interrupt service routines  ;
; Written to return character row, col and      ;
; ignore scan line and pixel info in AL.        ;
; The position is corrected so that 1<=R<=25    ;
; and 1<=C<=80. Note that column 1 is subject   ;
; to error. You should not work near the edges  ;
; of the screen. The correction factor includes ;
; adding one more to the column. This is an     ;
; empirical fudge factor. You may want to play  ;
; with this for your particular terminal.       ;
;***********************************************;
;
INT54   PROC    FAR
        PUSH    AX                      ;Save registers used
        PUSH    DX
        MOV     AX,BX                   ;Get char position (0-1999)
                                         to AX
        MOV     DL,80                   ;80 columns
        DIV     DL                      ;Divide position by 80
        XCHG    AH,AL                   ;Row in AH (0-24), Col in
                                         AL (0-79)
        ADD     AX,102H                 ;Correction factor
        MOV     CS:POS,AX               ;Store in POS
        POP     DX                      ;Restore registers
        POP     AX
        RET                             ;Far return
INT54   ENDP
;
INT54B  PROC    FAR                     ;Special pen INT service
        PUSH    AX                      ;Save register used
        MOV     AH,CS:PST               ;Get last status
        MOV     CS:PST,AL               ;Store current status
        MOV     CS:PSTO,AH              ;Store old status
        POP     AX                      ;Restore AX
        RET                             ;Far return
INT5B   ENDP
;
CODE    ENDS
        END     START
;
```

Running Your Application

Whether you use the BASIC interpreter or are running a compiled application, the first call must be to INIT. Your application must then have a routine to get the pen data when you want it. Finally, before program exit, make a call to RESTR.

In BASIC this might be as follows:

```
5  DEFINT A,X,Y             'Declare as integers
10 INIT=&HB                 'Fill in the entry point offsets
20 RESTR=&H28
30 DOINT=&H45
40 DEF SEG=&H6000           'Segment where driver will be
                             loaded
50 BLOAD "PEN.FIL"          'Load driver
60 CALL INIT                'Redirect INT 54H and 5BH
..
..
.. GOSUB 100                'Get pen position
..                          'Use it in any way you want
..
.. CALL RESTR               'Restore INT 54H and 5BH
.. END                      'End of program
..
..
100 CALL DOINT(A)           'Call the driver
110 IF A=-1 GOTO 100        'If not valid, call again
120 RETURN                  'Return from GOSUB
```

You may want to insert statement 115 X=A MOD 256:Y=A\256 to return the column and row directly. Be sure all these variables are declared as integers.

Summary

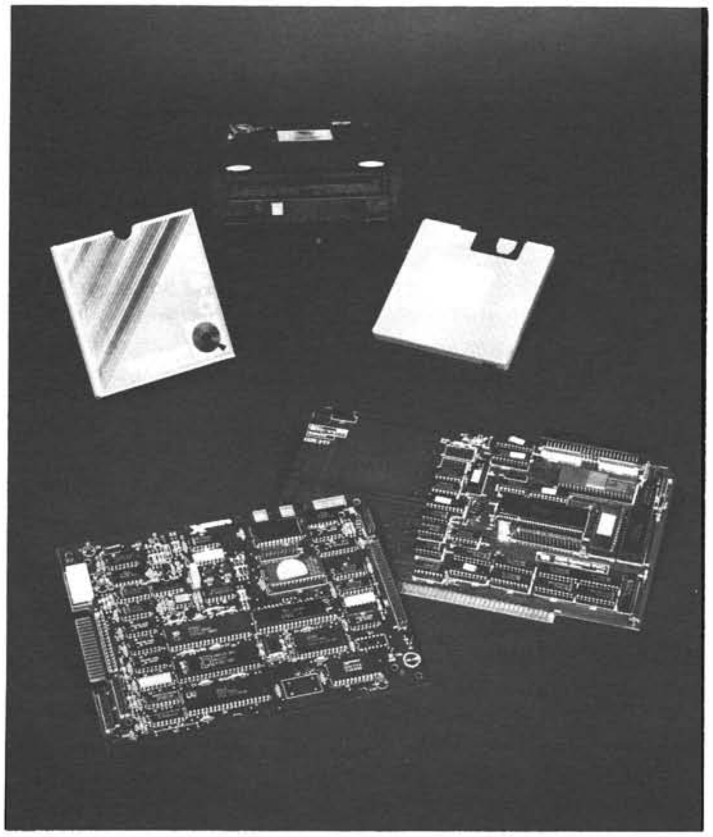
I have tried to present a fairly general, high level, light pen driver for Z-100 application programs. As you can see there is a great deal of flexibility in how certain details might be handled. My CALC program, written in C, makes calls to a similar assembly language driver to get the pen input. In this case a coded ASCII character is returned. The procedure is slightly different in BASIC for passing characters rather than integers.

So now, go to it and write some great light pen programs for HUG. I hope this article has been helpful in making it possible for you to accomplish that.

Listing 2

```
;****************************************************;
;                                                    ;
;       LISTING 2 -- DOINT for returning X,Y.        ;
;                                                    ;
; Use this procedure in DRIVER.ASM instead of DOINT in ;
; Listing 1 for the driver to return two parameters. ;
; Note that -1 is only returned in X if invalid data. ;
;                                                    ;
; Subroutine at 100 should now read:                 ;
;                                                    ;
;   100 CALL DOINT(X,Y)                              ;
;   110 IF X=-1 GOTO 100                             ;
;   120 RETURN                                       ;
;                                                    ;
;****************************************************;
;
DOINT   PROC    FAR
        PUSH    BP              ;Save base pointer
        MOV     BP,SP           ;Get stack frame
        MOV     BX,[BP+6]       ;Get pointer for Y
                                 (2nd parameter)
        MOV     SI,[BP+8]       ;Get pointer for X
                                 (1st parameter)
        PUSH    DS              ;Save BASIC DS
        MOV     AX,CS
        MOV     DS,AX           ;Make data segment=code
                                 segment
        MOV     AL,PST          ;Get current pen status
        OR      AL,AL           ;Is pen up ?
        JZ      D01             ;Yes, invalid
        MOV     AH,PSTO         ;Get old status
        AND     AL,AH           ;Is this first time ?
        JNZ     D01             ;No, invalid
        MOV     DX,POS          ;DL has X, DH has Y
        MOV     PSTO,1          ;Make sure old status set
        JMP     D02
;
D01:    POP     DS              ;Restore BASIC DS
        MOV     WORD PTR [SI],-1  ;Return -1 in X if invalid
        JMP     D03             ;Ignore Y and exit
;
D02:    POP     DS              ;Restore BASIC DS
        XOR     AH,AH           ;Zero AH
        MOV     AL,DL           ;Get X to AL
        MOV     [SI],AX         ;Store X value
        MOV     AL,DH           ;Get Y to AL
        MOV     [BX],AX         ;Store Y value
D03:    POP     BP              ;Restore BP
        RET     4               ;Pop two pointers and return
DOINT   ENDP
```

Are you reading
a borrowed copy of REMark?
Subscribe now!



A Versatile Hard Disk Alternative

Pat Swayne
HUG Software Engineer

A Review Of The CDR317 Hard Disk Controller For H/Z-100 Computers

If you have an H/Z-100 (dual processor) computer, and you have been thinking of installing a hard disk (Winchester) system in it, you would do well to consider one of CDR317 packages from CDR Systems. The CDR317 approach is more versatile than other hard disk systems because it uses the SCSI (Small Computer Systems Interface) standard that ensures compatibility with a greater number of mass storage systems, including some that haven't even been invented yet.

Product Description

The CDR317 itself is an S-100 SCSI adapter card that can be plugged into one of the card slots in your H/Z-100. It is available by itself, or in packages that also contain a controller board, a hard disk drive, or both. The CDR317 board translates control codes meant for the standard Heath/Zenith Z-217 hard disk controller into SCSI codes, so that a hard disk system built around it is bootable and fully H/Z-100 compatible. You can get drives that are SCSI compatible and use them directly with the CDR317, or you can purchase it with an additional controller board to run drives that use the ST506 interface standard (the kind of drives normally used in Heath/Zenith systems).

The optional controller board that CDR includes with the CDR317 is the Xebec 1410A controller. The CDR317 and the Xebec 1410A together duplicate the Heath/Zenith Winchester controller system in function, but will work with a larger number of drive types, including removable media drives. The CDR system will work in any H/Z-100, including those that have "old" power supplies in them that do not have proper connectors for the Heath/Zenith Winchester boards. The CDR317 card uses less power than the Heath/Zenith controller card, and gets its power directly from the S-100 bus rather than from a special connector. The Xebec card gets its power from a standard 5.25-inch drive power connector, and CDR provides a Y connector so that you can power the Xebec card and a drive from one cable.

CDR Systems had previously supplied the Omti 3100 controller board instead of the Xebec board with their system, but has since substituted the Xebec board because it supports more drives.

Assembly And Testing

The package that I received for review consisted of the CDR317 card and the Xebec card. I decided to try the cards out with a Syquest model SQ 312RD drive, which is a 10 megabyte removable media drive. This type of drive will not work with the standard Heath/Zenith controller, and I figured that it would be a good test of the CDR package.

The manual that I received with the CDR package is not the one that CDR is shipping now, but after I had assembled the unit, I received a new manual. Either manual leaves much to be desired. They do not contain anything resembling a step-by-step procedure for assembling the system, and the information that the average person would need is scattered throughout the manual among information that only the most avid hacker would be interest[illegible]in. In spite of this, I was able to figure out where everything went without too much trouble. (Aside from the installation procedure, the only thing you really need a manual for is the values to supply the PREP program once you get everything together.) CDR provides hardware that allows you to mount their two-board system similarly to the way the original Heath/Zenith two-board system is mounted. My system is the all-in-one model, and I mounted the Xebec board above the drive cage where the data separator board from a Heath/Zenith system would be mounted. The only difference is that the CDR hardware mounts the Xebec board with the solder side up.

When I had completed the mounting of the CDR boards and the Syquest drive, I was ready to power the system up and try out the new drive. The CDR documentation states that the H/Z-100 "hand prompt" will not appear until your drive has come up to speed. Since my drive used removable media, I inserted a cartridge before turning the computer on. The Syquest drive has a two-color LED on its front panel that is red when the drive is busy or while it is coming up to speed, and green otherwise. It took several seconds to turn green after I turned the power on, and the hand prompt appeared a couple of seconds later.

To enable you to initialize and partition your drive, CDR includes a copy of the Heath/Zenith Z217 Winchester Utility disk with their CDR317 packages. This copy has a label stating that it is "on loan" and that you should by a valid copy at your Heath/Zenith store when you can. However, since the H/Z-100 series has been discontinued, that may not be possible.

The Z217 disk contains a PREP program that is used to initially prepare the disk, and a PART utility that is used to divide it into partitions. Some Heath/Zenith hard disk systems are sold with the drive already PREPed, but it is likely that you will have to prep a CDR system yourself, especially if you provide your own drive. The CDR-supplied disk not only contains the standard Heath/Zenith PREP program, but also a patched version called FASTPREP, which runs faster because it skips some of the media test passes. It turned out to be a good thing that FASTPREP was there.

My first attempt at PREPing my Syquest drive resulted in an error message as soon as the media test part of the PREP program started. I tried again using other cartridges and even another Syquest drive, all without success. Then I put in a MiniScribe 2012 drive (one of the kinds Heath supplies), and it worked perfectly. I called Marc Brooks at CDR, and he suggested that I try PREPing the Syquest drive again, but that I specify a much smaller number of cylinders than the documentation indicated. (When you run PREP, you must specify certain parameters about your drive, including the number of "cylinders", or sectors per platter surface.) This time, the PREP process was completed successfully. So, on a hunch, I looked up the drive specification tables in my MS-DOS 2 manual Winchester Command Guide. The Syquest drive is not listed there, but I saw that the specifications for a drive containing the same number of heads and capacity showed 4 less cylinders than the CDR specification (in the original documentation I had) for the Syquest drive. I tried PREPing the Syquest drive again, and specified 4 fewer cylinders than the CDR documentation indicated, and the process was completed successfully. I informed Marc Brooks of my results, and he agreed that the cylinder count given in the documentation was probably in error. The new documentation confirmed this with a corrected cylinder count.

One thing you should be aware of about the CDR-Xebec system is that it formats disks using 17 sectors per track, which is the way it is done on IBM PC and compatible systems. The Heath/Zenith Z-217 controller formats with 18 sectors per track, which means that you get a bit more storage space on a disk of a given size, but that is also one of the reasons why the Heath/Zenith system will not work with certain drives. The CDR317 board makes H/Z-100 "see" 18 sectors per track somehow, so everything works normally. One problem arising from this, however, when you run PREP, you have to take the specified cylinder count, which is a hex number, and multiply it by 11 hex (17 decimal), and divide it by 12 hex (18 decimal), and use the result as the cylinder count for PREP. If I hadn't had Perks with its built-in hex-decimal calculator, I would have had problems getting my system up.

The new documentation that CDR sent listed four drive types that are not in the Heath/Zenith Winchester guide, and the values given are the ones that you would plug in directly, without having to do any conversions. It is hoped that a more complete documentation will be issued soon, containing not only the much needed step-by-step instructions, but also corrected PREP values for most popular drive types that are likely to be used with the CDR317.

Operation

When I had successfully PREPed a Syquest cartridge, I ran PART on it and divided it into two MS-DOS partitions, formatted one as a bootable system disk using MS-DOS version 3.1, and booted up on it. Everything worked normally, and it was just like using a standard Heath/Zenith H-100 Winchester system. I prepared another Syquest cartridge, also with MS-DOS on it, and tried changing cartridges with the system booted up and running. To remove the cartridge from a Syquest drive while it is running, you first press the release button in slightly. The LED on the front of the drive will begin to blink red. When it stops blinking and goes out, you can press the release button all the way, remove the cartridge, and insert another one. The LED will glow red for a while, and change to green when the drive is ready. When I changed cartridges in my system, it continued working normally, and I was able to access files on the new cartridge.

There are some problems that could arise from changing cartridges that I did not test (I had to give up the boards for magazine pictures). You could boot up on a cartridge containing two partitions, and use ASGNPART (MS-DOS 3) or ASSIGN (MS-DOS 2) to assign the second partition. Then you could change to another cartridge containing only one partition. There would probably be trouble if you tried to access the missing partition. Or, if you changed to a cartridge containing more partitions than the one you removed, would you be able to access the new partitions just by running ASGNPART again? And would it be a disaster if you removed an MS-DOS cartridge and replaced it with a CP/M cartridge?

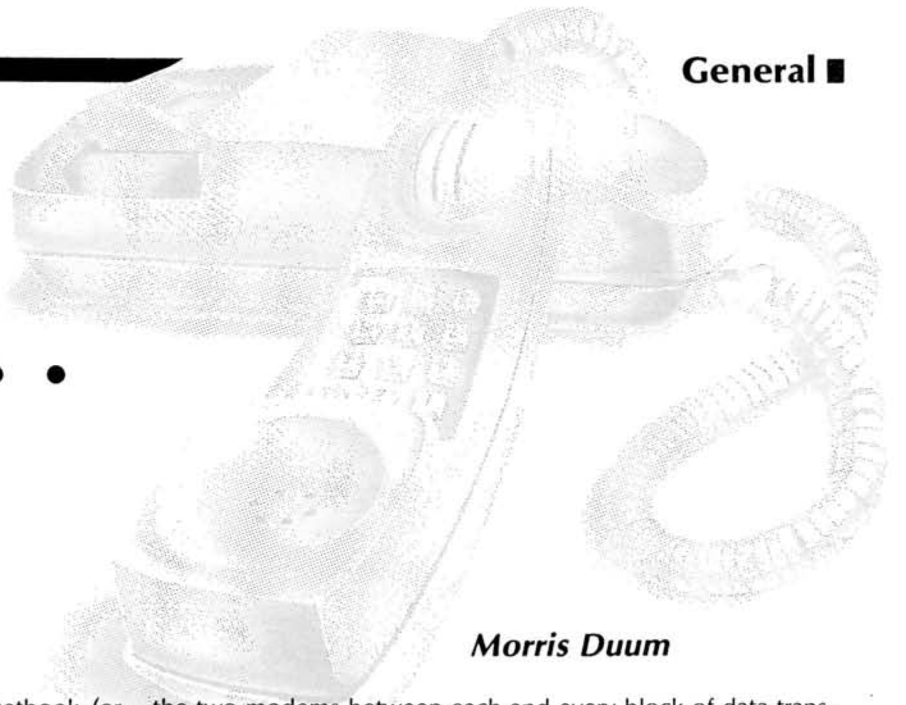
The Xebec controller supplied with the CDR package can support two drives, so you could set up a system with two Syquest drives, or one Syquest drive and one fixed hard drive. However, the Syquest drive gets so hot while it is running that it would probably not be a good idea to put a two drive system within your H/Z-100 cabinet. One or both drives could be mounted in an external cabinet with its own power supply.

Your Computer's Future With The CDR317

The SCSI interface that the CDR317 uses was designed not just for supporting hard drives, but also for other mass storage systems, including tape drives and optical disks. It is also designed for other uses besides mass storage, such as networking and driving laser printers. CDR has plans for using the CDR317 in at least some of these applications.

Continued on Page 23☛

Free Software . . . For Less!



Morris Duum

If you're a subscriber to CompuServe, your pocketbook (or charge card) is being slowly picked away at, 'bit by bit', so to speak. After becoming registered on HUGPBBS (HUG's Personal Bulletin Board System in St. Joe), I began to notice a definite difference in speeds when downloading files from that bulletin board, and CompuServe. It was then I decided to do some timing, and connect charge comparisons between the two systems. The results are quite eye-opening, and worth noting!

Typically, Compuserve charges $6.00/hr at 300 baud, and $12.00/hr at 1200 baud, for non-prime time (after 5:00 pm) connect charges. Since 300 baud is just about as bad as 110 baud, I'll ignore it for this discussion. If you don't have a local CompuServe number you can call, then you most likely will have to use a go-between called TYMNET. This is another $2.00/hr charge. Assuming you don't have to make a long distance call, your flat charge is now $14.00 per hour of connect time with CompuServe at 1200 baud. *Ed: Prices are now slightly higher.*

Now, let's assume you're calling HUGPBBS from somewhere out in California, or Washington state. AT&T non-prime time (after 5:00 pm), weekday long distance rates for 1 hour is roughly $12.67, and even less after 11:00 pm. That's less of a charge per hour than CompuServe!! The charges from Florida and Massachusetts are roughly the same.

Ok, so now by calling HUGPBBS instead of CompuServe from anywhere in the continental United States, I can save, at minimum, $1.33. So, big deal, so what's a buck and a half an hour? But wait, remember I said that there was a download speed difference? I made some timing comparisons by downloading the same file from both systems, and this is what I discovered.

The test file I used was 520 XMODEM blocks long, or 66,560 bytes (520 x 128). Ideally, this file should take 9.24 minutes to transfer at 1200 baud. This value was derived from the formula:

Time in Minutes = ((filesize in bytes / (baud/10)) / 60)

This value assumes no disk access time, no block re-transmission because of errors, and no protocol handshaking between the two modems. This ideal condition, of course, does not exist. Time must be taken to read and write the blocks to disk, errors do sometimes occur causing retrys, and handshaking does go on between the two modems between each and every block of data transferred. These conditions can be thought of as 'constants', no matter what system you're connected to.

Downloading the file from CompuServe, through TYMNET, took an actual 24.5 minutes! That's an effective baud rate of 453!! When I eliminated TYMNET, and went to CompuServe directly, the download still took 18.5 minutes, or an effective baud rate of 600!! This test was done several times, and the results were always roughly the same. To say the least, I was a 'bit' upset to be getting less than half of what I was really paying for.

After cooling off a bit, I decided to give HUGPBBS the same download timing test. The same file when downloaded from HUG, took 11.1 minutes, or an effective baud rate of 1000. Now that was more like it. More than twice as fast as CompuServe and TYMNET, and at a reduced connect time charge! It looks to me, that even under the worst conditions, connecting up with HUGPBBS can realize a savings of at least 60% over CompuServe, no matter where you live in the continental United States. Let's look at it another way. If you downloaded a 360000 byte file from HUGPBBS, it would take you exactly 1 hour, and cost $12.67 (after 5:00 pm). If you downloaded the same 360000 byte file from CompuServe, it would take you 2 hours and 10 minutes, and cost $30.33.

Agreed, HUGPBBS isn't as big and fancy, or have alot of services, but if all you want is to exchange messages with other Huggies, or download a pile of software, then HUGPBBS is the best value. In fact, your download time can be decreased even further still by running at 2400 baud. Yes, HUGPBBS will work at 2400 baud, and CompuServe won't.

Getting registered on HUGPBBS is easy. First, you have to be a Huggie (I don't mind being called a Huggie), and have your hug I.D. number handy. Next, you need to make up some sort of password (up to 16 characters). Your password can be anything you like, and hopefully something you won't forget (write it down to be safe). For example, my password is 'FLEA-BITTEN', because my dog always sleeps by my bed. Finally, that information can be left on the HUGPBBS system, with your first and last name, in a private message to the SYSOP, or you can call Jim Buszkiewicz and give him the information directly. The two telephone numbers are: (616) 982-3956 for HUGPBBS, and (616) 982-3837 for Jim Buszkiewicz. Happy modeming!

A magazine for Kit Hobbyists

Special Edition

$2.50

Kit Builders Journal

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.

Using Computers With A MODEM

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

MODEM. We've all come across that term at one time or another in the pursuit of our hobby, and most of us are aware it means MODulator-DEModulator. (A word about spelling. Since it's an acronym, I prefer to spell it "MODEM," but many publications spell it "modem." It's become such a common term that I suppose either is acceptable. You can spell it your way; this is my article.) With the price of MODEMs dropping almost daily, more and more people are becoming interested in what they are and how to use them. The MODEM is a useful piece of equipment used for communicating between data devices, usually over the phone lines. If you're like me, with a relatively well-developed system that has adequate memory (does anyone ever really have enough memory?), a good terminal and a disk drive or two, you may be ready to take another look. You may especially want to investigate the fascinating world of the electronic bulletin board or information service. Here's how to begin.

If you have an H-8 or an H-89 computer, or the Zenith version of same, then this was written especially with your problems in mind. Most of what we'll have to say will apply equally to the H/Z-100 series and the flock of IBM PC-compatibles, whether Heath/Zenith's or Someone Else's. The basic principles would also apply to any other microcomputers you might have lying about, such as the TRS-80, Commodore 64, Apple II or Timex 2068 (I just mentioned that last one to see if you were paying attention.) The only significant variations will be in regard to the actual physical connection and the software you'll be using. Since we'll have to draw the line somewhere, we'll draw it here and say that if you're not using an H-8, H-89 or H-100, you may have to do a bit of interpolating. Don't worry, it's all a lot easier than you might have been led to think. And remember, when all else fails, read the instructions!

Without going into great detail, a MODEM will allow you to access any source of digital data that meets some fairly common definitions. First, for the less expensive variety of MODEM, the data transfer rate is generally 300 baud. The audio signal on the phone line must be in accordance with Bell Telephone standards (that is, compatible with their Bell 103A MODEM.) There are faster MODEMs that don't cost all that much more money; what we say about the 300 baud equipment generally applies to the 1200 and 2400 baud MODEMs now available. And finally, the MODEMs we'll talk about are serial devices that fall into the class of Data Terminal Equipment (DTE), the same as a computer but different from a terminal or a printer, which are Data Communication Equipment (DCE.) This last point will be important to keep in mind when it comes time to interface (connect) the equipment.

Since we're concentrating on the kind of MODEM that you plug into the back of the computer, we'll talk mostly about one common type, the Cat, made by Novation. This is possibly the most frequently seen inexpensive MODEM for a number of reasons. First of all, it works well. The price is reasonable, less than $100.00. Interfacing is relatively easy, due to the EIA 25-pin DCE female socket built-in. The Cat has its own power supply, transformer-isolated from the 120 volt AC line for safety (UL approved), and finally, it is switch-selectable for originate or answer mode. More about that later.

The Cat is one of a family of inexpensive MODEMs that use accoustical couplers to connect the MODEM to the phone line. With this type, the telephone receiver or handset is placed into the rubber cup-like recepticles on the MODEM, and no other link is required. Obviously, a Princess-type phone won't work; you need an older model with a conventional handset. There are other types, known as hard-wired dial-up MODEMs, that are connected directly to the phone line, usually by plugging into the wall phone recepticle. These are generally more expensive, and may be subject to local phone company tariffs and regulations. They can be bought with the capability to transmit at up to 2400 baud, due in part to their ability to avoid the accoustical coupling and in part to more sophisticated electronics. Many such MODEMs are available that plug directly into S-100 and other busses, suitable for use with the Heath/Zenith 100 series of computers.

Because the Cat is so common, many computer manufacturers have chosen it as their standard, putting their own label on Novation's product. Heath and Radio Shack are just two examples. This also increases the number of sources you can check, since a MODEM with Radio Shack's label will work just as well with a Heath H-8, H-89 or H-100 series as it will with Radio Shack's own computers.

Regardless of the brand chosen, all MODEMs in the $100-$500 price range work on the same principle. The serial digital signal from your computer's I/O port is converted into an audio signal by the same technique used by most cassette I/O boards, Frequency Shift Keying. This is just a fancy way of saying that a "mark" or "1" will be translated into a 1,270 cycle tone, while a "space" or "0" will become a 1,070 cycle tone. Actually, there's a whole set of frequencies involved in the CAT's operation, due to the fact that it can be used either to originate a message or to receive a message, and these standard frequencies are shown in Table 1.

Transmitter Frequencies	
Originate Mode	
Mark	1,270 Hertz
Space	1,070 Hertz
Answer Mode	
Mark	2,225 Hertz
Space	2,025 Hertz
Receiver Frequencies	
Originate Mode	
Mark	2,225 Hertz
Space	2,025 Hertz
Answer Mode	
Mark	1,270 Hertz
Space	1,070 Hertz

Table 1

These are the Bell 103A audio frequencies used by voice-grade MODEMs. Note how the frequencies are assigned depending on the choice of originate/answer mode and whether the MODEM is transmitting or receiving.

The table of frequencies isn't all that important, because from a practical standpoint it won't usually make much difference whether you're in one mode or the other. The important thing is that your MODEM and the one you're talking to must be in different modes. Otherwise, no communication takes place. For instance, most of the large time-sharing systems that allow you to "dial-up" the computer from a remote site are operating in the answer mode, and your "dial-up" link will be set for the originate mode. As a matter of fact, the communication link would work as well if the modes were reversed. As the French say, "Vive la Difference!" It's the difference that's important.

The mode factor becomes important when you want to talk to a "non-standard" system, such as the one owned by your friend across town. If you both bought originate-only MODEMs, you're out of luck. Such MODEMs are available, and they're usually cheaper, so keep in mind what you want to do with it before you buy it. Keep in mind, too, that some MODEMs advertised as capable of originate and answer modes can actually be altered only by rejumpering the circuit board. Look for a MODEM that's switch-selectable. Most of the better MODEMs on the market today are dual-mode, so you normally won't have a problem, especially with the plug-in circuit cards.

Speed is another factor to consider. 300 baud MODEMs are a lot cheaper than 1200 baud versions, which are cheaper than the 300-1200 selectables, which in turn are cheaper than the 300-1200-2400 baud models just coming on the market. What you spend on the equipment, you can save on telephone connect charges. So you get what you pay for in speed, up to a point. That point is signal reliability.

The reason for the limitation on transmission speed is primarily one of design limitations, partly for equipment but mostly for the phone lines. Voice-grade lines such as we use aren't the best possible, especially since deregulation has resulted in a proliferation of local companies who often don't match Ma Bell's circuit quality. A better circuit can be provided but the cost is astronomical. However, if time is money (as it is for the Big Boys) you can do what they do and order a data-grade line. But at several kilobucks a month you may not really want a 9,600 baud circuit. There's an inevitable trade-off between speed and cost. Keep in mind that at 300 to 1200 baud, the reliability of your data transfer will be limited primarily by the quality of the phone line.

After this brief look at the theory and operation of MODEMs in general, we're ready to consider interfacing. This is the point where you'll need to dig out your system specifications and see what you have to work with. We'll assume, to begin with, that you will be interfacing a computer to the MODEM. Since the MODEM is a serial device that speaks in RS232C, so that's the standard to which we must work.

What you have to go through to connect your MODEM depends largely on what computer you have. TRS-80 and Commodore 64 users can connect directly to a specified MODEM port. Most laptop portables have the same arrangement. If you have a buss-oriented system like the H-8, H-89 or most SS-50 or S-100 machines, or if you have an IBM PC-compatible, it will be a matter of what sort of serial I/O ports your computer has available. For instance, with the H-100, there are two RS232C ports, one of which is configured for MODEM use (DTE). The H-89's three-port serial card has the same arrangement, with one port already configured for MODEM use. Most other buss-oriented systems will have similar provisions. No-buss machines like the Heath H-88 and H-89 have slightly different provisions, but in general the idea is to plug the MODEM into an RS232C port. The pins used to connect the Cat MODEM are shown in Table 2. Not all the RS232C lines are used; these are the only ones you need to connect.

MODEM Pin	Function	Description
1	Protective ground	Used to link the chassis grounds of the MODEM and the data terminal.
2	Transmitted data	Used to transfer data from the data terminal to the MODEM.
3	Received data	Used to transfer data from the MODEM to the data terminal.
4	Request to send	Indicates the data terminal is ready to transmit data.
5	Clear to send	Indicates the MODEM is ready to receive data.
6	Data set ready	Indicates the MODEM is on-line to the phone and is in the data mode.
7	Signal ground	Provides a common reference for the signals.
8	Carrier detect	Identical to Clear to Send.
9	Ringing indicator	Indicates the MODEM is receiving a ring signal from the phone. Not normally used in simple interfaces.

Table 2

Description of the pins and functions on the standard 103A-compatible MODEM. Note the duplication of some pins, and the fact that all MODEM pins are not always required.

At this point it may be necessary to determine whether or not your chosen port is configured for DTE or DTC. In most systems it will probably not matter, but in some like the H-8 when equipped with the H8-4 board having sophisticated programmable I/O features, it will be necessary to insure that the proper configuration is used. If this is a consideration, your system documentation will make that clear. Basically, any serial port can be used, it's just a matter of configuration.

This is a good place to point out the fact that it isn't necessary to have a computer to make use of a MODEM. As we saw, the MODEM translates the audio signals to RS232C-compatible digital signals. Your stand-alone terminal is capable of printing the data out, on the CRT or in hard copy (DECwriter, etc.) In fact, this is one of the most common uses of our class of MODEM. If you have access to a large time-sharing system at work or at school, you can interface a MODEM to your terminal and operate in the same manner as you could at work or at the school's computer center or terminal room. For many people, this is entirely sufficient; after all, if you have a CDC 7600 at work and you can tap into it at home, a TRS-80 might lose some of its appeal.

You might also note that the Cat-type MODEMs provide a convenient test mode for use with just a stand-alone terminal. This allows you to talk through the MODEM from the keyboard to the CRT or printer, in both full and half duplex modes. Any problems uncovered here can be corrected before you undergo the frustration of trying to communicate through a defective MODEM. In reality, the MODEM itself is highly reliable, but some of the interconnections may have come adrift or you may uncover a power supply problem.

The final obstacle to your operations will be the MODEM device driver. Lap-top computers generally have the necessary software built-in. For most systems, based as they are on a Disk Operating System (DOS) of some sort, the device driver is a program residing on the system disk that can be called up for data transfer when desired. The precise form of the MODEM device driver will depend on the DOS; there is a new version available from Lifeboat Associates called "BSTAM," intended for all CP/M-based systems. The Heath Users' Group provides two for the H-8/H-89, one especially tailored for Heath-to-Heath communications (H8COMM) and one intended for use by a Heath owner communicating with any other type machine (MCS.) There are several others tailored to the H-100 and the H-150/IBM PC-compatibles. The point here is that without the device driver the MODEM can't be used with the computer. It can, of course, be used with the terminal alone. In some cases, the device driver or MODEM program is an extra-cost item, so be sure to check that it's available. While program commands differ, Table 3 shows a representative set, and Figure 1 shows a brief session at the console using this particular program (MCS.)

Once the interfacing and device driver questions are answered, you're ready to go on the wire. Well, almost. First is the little matter of protocol. This term means different things to different individuals; the engineer thinks of the way the signals are transmitted, received and processed, but what we mean is the manner in which you and your partner on the other end are going to conduct communications. Remember, one of the MODEMs must be set to originate, the other to answer.

Next, you will need to decide whether to use half duplex or full duplex modes, and you may need to configure the device driver for a particular I/O port number and baud rate. In most cases, the device driver specifications will tell you what it requires, and in some instances you may have the option to reassemble the driver to your own specifications using the source code provided. The Heath Users' Group drivers fall into this category. It is, however, possible that you'll have to experiment. Half duplex means that in effect there is only one signal carrying wire and a ground connecting the data terminal and the MODEM (and hence the two MODEMs.) This means that signals travel in one direction at a time. Full duplex means that there are two paths for the signals, one each way, so signals can travel in both directions at once. Under normal circumstances the mode will be invisible to the user, and it won't make any difference in system performance. Figure 2 illustrates the connections.

Once you've decided how to communicate with the other terminal, you're set. If you're dealing with a time-sharing system, there will be certain formalities to observe. You will be in the originate mode and the other system will be in the answer mode. This isn't a bad rule of thumb to follow even if you have the option of reversing the modes. That way, you'll always be sure who's in what mode, because it will depend on who calls whom.

MCS Operating Instructions

Commands

BAUD — Set baud rate. "3" sets rate to 300.

BYE — Exit to HDOS.

CLEAR — After writing a file, the file isn't closed. If file is not for retention, CLEAR erases it.

CLOSE — Closes a file for retention. Either CLEAR or CLOSE must be used before moving on to another file.

CONVERSATION — Conversation mode. Enter before establishing communication. To return to mode, type CTRL-B.

COPY — Copy a file from remote site to your console and memory. Does not write to disk (WRITE does this). Files may be concatinated by answering "NO" to request for new buffer. When asked for "FUNCTION" use the remote computer's command to access a file. You must exit this mode by CTRL-B when transfer is done. Then WRITE and CLOSE.

OPEN — Open a file for write. Used with COPY, WRITE and CLOSE, but not with SEND. The name of the output file will be requested.

SEND — Transmit a file to the remote site.

WRITE — Write the file from the buffer (memory) to the disk file previously OPENed.

Table 3
A representative MODEM driver instruction set. These are the commands for MCS, a Heath Users' Group program for use with a Heath H-8 or H-89 talking to nearly any other computer, from a TRS-80 to an IBM 370.

Keep in mind the fact that the data transfer reliability will be somewhat degraded by any noise on the phone lines, especially on long distance. When you first pick up the receiver, dial the first number and listen for the background noise. If there's much at all, hang up and redial. Few things are more frustrating than to lose the carrier signal on the last sector of a 50 sector file you're transferring. Some MODEM drivers allow partial recovery, but its a hassle any way you look at it.

One final note. Keep track of how long you're on the line and how far you're calling. At 300 baud, a 4K program will take about 2-1/4 minutes to transfer, and those long-distance toll charges can creep up on you. You might stop by the local phone office for a rate calculator, a circular card device that gives the rates for different times and distances. Don't overlook the special rates offered by long-distance carrier companies. They can often save you a bundle. Look, too, for local network access to major bulletin boards. In many cities, this local access allows you to connect to a nation-wide bulletin board for the cost of a local call.

Once you try data transfer over the phone, you'll wonder why you didn't start sooner. There's no easier or faster way to exchange data. Remember that in addition to being able to communicate with others who have the same type computer, you can also download programs and data from computers vastly different in type and size. Once the MODEM drivers put the data in RS232C format, it doesn't matter whether your remote site is a TRS-80, an Apple II, an IBM 370 or a Heath H-8. You can talk to them all.

Command	**Comment**
>MCS	Call MCS
MCS 01.03.04 ENTER "HELP" FOR ASSISTANCE	Log-on message
COMMAND=CONVERSATION	Conversation mode
.OK	Ready prompt
<CTRL-B> COMMAND=SEND	Exit to command mode Send a file out
.OK ENTER FILE NAME:LIFE.ABS TRANSMITTING..... FILE HAS BEEN TRANSMITTED	Identify file
COMMAND=OPEN	Open H-8 file
.OK ENTER FILE NAME: ACCOUNTS.BAS	Name of H-8 file
COMMAND=COPY	Receive a file
.OK DO YOU WANT A NEW BUFFER? YES	Don't add to old buffer
ENTER FUNCTION:	<CR> for TRS-80
COMMAND=	After transfer, type the next command

Figure 1
Part of a console session using MCS. A Heath H-8 is communicating with a TRS-80, first sending a file to the TRS-80, then receiving one.

```
----------                                                  ----------
|        |                                                  |        |
|        |                                                  |        |
|        |------------------------------------------------->|        |
|        |                                                  |        |
|        |                                                  |        |
----------                                                  ----------

  MODEM      Data transfer one way at a time                  MODEM

                        HALF - DUPLEX

----------                                                  ----------
|        |                                                  |        |
|        |                                                  |        |
|        |------------------------------------------------->|        |
|        |<-------------------------------------------------|        |
|        |                                                  |        |
----------                                                  ----------

  MODEM        Data transfer both ways at once                MODEM

                        FULL - DUPLEX
```

Figure 2
The difference between half and full duplex.

C__POWER

Part 1

John P. Lewis
6 Sexton Cove Road
Key Largo, FL 33037

This article is directed to the "Huggies" running CP/M on their H/Z-89s and/or H/Z-100s. Most of the information will also apply to HDOS users, but since I am not familiar with the Heath DOS, I can't point out the places in the code which would have to be altered to enable the user to "run" the following program or use the subroutines listed.

By way of an explanation of the following, I had been searching since October 1983 for a high level language that was portable, easy to learn, and very flexible. In other words, powerful. I have used BASIC and Fortran with good results, but found some of their routines too confining. I tried assembly language, but found that to be too time consuming although very powerful. About a year ago, I discovered C/80. I was immediately impressed with the speed and power of the compiled programs. The more I learned about "C", the more I liked it. Some of the expressions are a bit foreign to a hacker who started with BASIC, but relatively easy to learn.

One of the features of "C" which makes it so easy to use is the ability to write and include subroutines which are peculiar to the Heath environment. These subroutines can become part of the user's library and, henceforth, called from your program to locate the cursor, clear the screen, input characters without echo or a multitude of other jobs which you deem appropriate.

Again, let me emphasize that I'm using C/80 from Software Toolworks. I purchased this gem about a year ago and consider it to be the best investment I ever made in software. For $49.95 you can get a copy which even includes the source code for many of the library functions. This does not include the mathpak which would be needed for math functions involving floating point or double precision. Only integer and unsigned math is supported in the basic package. An expenditure of an additional $29.95 for the mathpak will put you in business with a full featured compiler, capable of 32 bit math.

Brian Polk wrote a series of articles for Huggies on the "C" language which were directed at the new user. This series was a great help to me in getting started in the "C" language, not to mention a bit of an inspiration. This series took the reader through some of the basic functions and file handling routines. The following program will show the reader how to build a custom library of "C" functions for his own use. Building more versatility into the language with each addition. By the way, as an illustration of C__power, this article is being written using a text editor which I wrote using C/80.

Here's a program which will include the functions which I described above:

```
#include "printf.c"

cls ()
{
putchar(27);putchar(69);/* escape, "E" sequence for H89's */
}

locate (row,col)
int row, col;
{
row+=31; col+=31;
putchar(27);putchar(89);   /* escape, "Y" sequence for direct */
putchar(row);putchar(col); /* cursor control */
}

fetchc ()
{
#asm
DIR:    MVI E,0FFH      /* "direct" call to CP/M Dos */
        MVI C,006H       /* for I/O (input) */
        CALL 5
        CPI 0
        JZ  DIR
#endasm
}

gofor (c,s)
int s;
char c[];
{
int i;
for ( i=0 ; i <= s && c[i-1] != 13 ; ++i )
{
c[i]=fetchc();/* c[i]=toupper(c[i]) */ putchar(c[i]);
if ( c[i]==8 )                        /* this subroutine "calls" */
   {                                  /* fetchc () for character I/O */
   i-=2;putchar(32);putchar(8);
   }
}
if ( i-1 == s )            /* a return or s+1 characters will */
c[i-1]='\0';              /* cause this routine to return */
if ( c[i-1]==13 )         /* and will replace the return with a null */
c[i-1]='\0';              /* ('\0'). This is a "C" requirement */
}

main ()
{
char string[80];
cls ();locate (4,6);printf("We have tested the clear screen and");
locate(6,6);printf("the direct cursor addressing functions incorporated");
locate(8,6);printf("in this program.");
locate(10,8);printf("Now we'll test the string functions, press");
locate(12,8);printf("Return to continue ");gofor(string,2);
cls();locate(6,4);printf("Enter your string: __________________________");
locate(6,23);gofor(string,25);
locate(10,4);printf("Your string = %s\n",string);
locate(12,4);printf("Press any character to return to CP/M ");
gofor(string,0);exit(0);
}
```

Here we have four functions which can be included in future programs and some hints to be used in designing additional functions to suit your own needs. If you wish to use these functions in other programs that you may write in the future, simply copy the program as it is listed and test it. Be sure that everything works as it should before taking the next step. Murphy (of Murphy's Law fame) seems to ride around in.my pocket when I'm writing programs so I do a lot of fixing!

After thorough testing, use your word processor to delete the main function. Then give the remainder of the program an appropriate filename, filetype. The filename may be anything of your choosing (within the boundaries specified by CP/M) but the filetype must be C. Then when writing your next "C" program and wanting to use one or more of these functions, all you have to do is make sure this program segment is on the disk with your creation and at the beginning of the program where you will be listing the files to include be sure you enter: #include "filename.c". Your program may then call these functions at will, all without your having to write this code again.

Let's go back and examine the program. The first function encountered is cls (). This is the old and familiar clear screen function, but now we have simplified it and we can utilize it from the body of our program with a cls ();. No argument was needed so none was passed. Please note the semicolon after the parenthesis. This tells the "C" compiler that it has reached the end of a statement. This function was executed using the call to Putchar (). We passed the decimal value of the characters to be sent to the terminal as the argument. Putchar(27) sends "escape" to the terminal.

I borrowed a bit of terminology from ZBASIC for the next function. Locate (row,col) will place the cursor at the coordinates specified in the argument. We do this with the escape "Y" sequence. We increment the arguments with an arg+=31 which will satisfy the terminal with the proper information to place the cursor in row 6, column 10 with the function call, locate (6,10);. Notice that row and col were declared to be of type int outside of the open brace which precedes the body of our function. This satisfies the "C" compiler with a data type for the variables passed.

We have to resort to assembly language for the next function in order to create another method of character I/O. This function emulates the INPUT$ function in BASIC, but we do not stop there. Stick around, things are just beginning to get interesting! Notice that we begin this function with #asm and end it with #endasm. This tells the "C" compiler that we are inserting assembly language code here. By the way, when you are entering this code, be sure you do so in upper case as shown in the listing. The AS assembler included with C/80 does not like instructions in lower case! We load the "E" register with 0FFH, the "C" register with 6 and call CP/M, (5 is a CP/M entry point in memory). We then loop until a character is found in the "A" register (not zero). This code constitutes a "direct" call to CP/M for character I/O. The character is NOT echoed under this protocol, we will determine how the character is handled with the next function, gofor (c,s).

Here again we are passing two variables, the character string and the number of characters in the string. These are declared as to type before the open brace preceding the function. We also declare i as an int within the function. This will be used as a subscript in dealing with the character array which we will be creating. A "for" loop is used to create the string using successive calls to fetchc () and incrementing the subscript [i]. After each call to fetchc () we check to see if we have exceeded the number of characters to be included in this string or have input a return (13). This is accomplished using the logical "and" (&&) within the "for" loop. When either condition described within the loop becomes false, we fall out of the loop and execute the code below it. Meanwhile, we check each character for a backspace (8) and take appropriate action if we encounter this condition. This is done by printing a space, backing up the cursor, decrementing the subscript by two (it will then point to the character preceding the backspace) and waiting for another character. The code below the "if backspace" provides a means of terminating this string with a null character. "C" demands this and if you do not provide it, you will encounter all kinds of strange behavior from your console. This function ensures that the user will not input any more characters than specified. That is a bit of insurance that you will find very handy since "C" is very unforgiving when the confines of an array are exceeded. In other words, if an array is declared for 25 characters and 26 are input, the last character will overrun the program memory with unpredictable results. This new function can help make your programs "bullet proof".

Now we come to the "main" body of the program, pun intended. From here we call our new functions for testing. You should find that each of them works as described and now you can print a message such as "Input string", jump over it and place the cursor for character input. Try inputting over 25 characters when asked to "Enter your string". You should find that the input ends with the 25th character. The next character will cause the program to "fall out of the loop" and print your 25 character string. The next command in the body of this program might cause a bit of consternation since it says "gofor(string,0)". Here we are looking for just one character that will not be stored, hence any input will exceed the counter reference and cause a "fall out" and return to CP/M.

I have painted a rather rosy picture of "C" so far, how about the other side of the coin? As the saying goes, "there is no such thing as a free lunch". That applies to computer languages too. The price extracted for all this power is in the process of writing and testing your new program. The first problem that we encounter is that of storage. You will need, at a minimum, the compiler, the assembler, the editor of your choice, the "C" library files, as well as the source code that you are creating. In addition to the above files I keep "PIP" and "STAT" on standby. The above files, not including the source code, exceed the capacity of a hard-sectored disk. This is no problem if the programmer has two or more disk drives or if he is using an H/Z-100 series computer with 360k capacity per drive, his problems of disk storage are, "no problem". The reader with an H-89 and one hard-sectored disk is going to be very busy using "PIP", "ERA" and swapping disks. The solution to this dilemma was described in the January '86 issue of REMark (see "A Tight C/80 Environment" by Don Keller). This article describes the writers solution to this unenviable situation, but he is using HDOS, so some of his solutions will differ from those that would work under CP/M.

My answer to this problem is an H-89 with 3 drives (hard-sectored), as well as an H-120 with a single 360k soft-sectored double-sided drive. Your solution will probably be between these two extremes.

When writing a program in BASIC, the reader has only to exit the input mode and type "Run" on the console, to test his creation. Things are not quite so simple when writing code for a compiler, whether it be Fortran, "C", Cobol, or something similar. Testing involves compiling the source code, assembly of the resulting asm file, and then a final testing. All of this, providing the compiler does not bombard the programmer with error messages (I get lots of those). You might wonder, at this point, if the product of your endeavors is worth the price in time. My answer is a resounding, yes. The benefits are many, speed of execution is one of the most notable, but a language that I can customize for my needs is the most valuable to me. I'm no longer confined to the functions provided by BASIC or Fortran. C/80 has been my language of choice for some time now. Only when I need to do some "quick and dirty" programming, do I resort to BASIC.

I hope that this article will help other "Huggies" get started in exploring "C". The pages of REMark have provided me with a shove in the right direction for many of my programming projects.

Equipment required to implement C/80:

H/Z-89 with 64k and one drive (two recommended).

Other articles of interest:

"An Introduction to 'C'", Parts 1 through 7, by Brian Polk, REMark. Aug. 83, Sep. 83, Dec. 83, Feb. 84, Mar. 84, Apr. 84, Aug. 84.

"Two Useful 'C' Functions", REMark, April 1985 by Charles R. Winchester.

C/80 is a product of:

The Software Toolworks
15233 Ventura Boulevard, Suite 1118
Sherman Oaks, CA 91403

*

Continued from Page 14

If you are planning to put a hard disk in your H/Z-100, the CDR317 deserves a close look. A starter system including a 10 megabyte drive will cost you less than a comparable Z-217 system, and offers more expansion possibilities.

Product Nutshell

Product:	CDR317
Manufacturer:	Controlled Data Recording Systems Inc. 7210 Clairemont Mesa Boulevard San Diego, CA 92111 (619) 560-1272
Prices:	Various packages available. Call for prices.
System Requirements:	Any H/Z-100 series computer with any H/Z-100 operating system that supports a Winchester system.

*

Are you reading a borrowed copy of REMark? Subscribe now!

ANALYTICAL PRODUCTS **805/688-0826**

213 Teri Sue Lane Buellton, CA 93427

EMULATE - H89 breaks format barrier!

Purchase CP/M software from vendors who do not support Heath disk type. Handles these disk formats:

Actrix	Eagle II	NCR DecMate 5	Televideo
Altos	Epson QX-10	NEC PC-8001A	TRS80-1 CP/M
AMPRO	Fujitsu CP/M86	Osborne 1	TRS80-3 CP/M
Beehive Tpr	IBM CP/M86	Otrona	TRS80-4 CP/M
CDR Systems	IMS 5000	PMC MicroMate	Xerox 820
Cromemco	Kaypro II	Royal/Triumph	Zorba
DEC VT180	Magnolia	Sanyo 1100	
DEC Rainbow	Morrow MD	Superbrain	

H37 version includes formatting capability.

H89 or H8 with H37 .... $59
For CDR BIOS 2.91 .... $49

CPC CP/M ⇐⇒ PCDOS

Transfer files between PCDOS and CP/M disks. Includes access to subdirectories. For H89 or H8 with soft sector controller.

CPC — specify controller................ $35

THE SMART CHECKBOOK

A personal finance program by Softquest. Has power to meet sophisticated financial needs. Easy to use, even for the novice.

Smart Checkbook $89
Specify CP/M or MSDOS and disk format

SUPERSORT

Powerful sorting program originally sold by MicroPro.

Supersort $119
Specify CP/M or MSDOS and disk format

TURBO MODULA 2

Borland/Echelon Turbo Modula 2 software for CP/M.

Turbo Modula 2 — specify disk format $65

TURBO PASCAL

Popular Pascal from Borland International.

Turbo Pascal 3.0 .... $59
Turbo Toolbox $45
Specify CP/M or MSDOS and disk format

C/80 Compiler

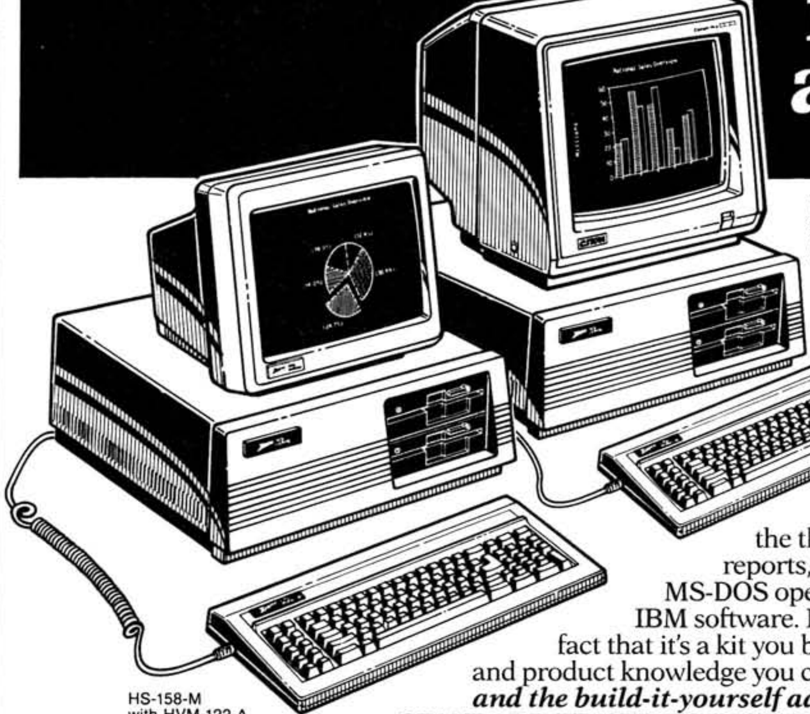
The C Compiler from *The Software Toolworks.*

C/80 Compiler $45
C/80 Mathpak $27
Specify CP/M or MSDOS and disk format

CALL OR WRITE FOR CATALOG **PRICES SUBJECT TO CHANGE**

Terms: Check or Money Order — Visa, M/C — C.O.D.
Add $3 per order for shipping and handling
California residents add 6% tax

A whole lotta meg for a whole lot less



HS-158-W
shown with C.ITOH
RGB Color Monitor
(sold separately)

HS-158-M
with HVM-122-A
Monitor

The IBM PC compatible kit computer with memory to spare

Having enough memory and storage for your business, education, or home use is important. And the Heath/Zenith HS-158 was designed with that in mind. Its 20 Meg Winchester and 640K memory are just the things for storing volumes of important financial reports, term papers, and budgets. The HS-158 uses the MS-DOS operating system and runs most industry-standard IBM software. But what makes this PC even more special is the fact that it's a kit you build yourself. For that pride of accomplishment and product knowledge you can't get with other PCs. ***More memory/storage and the build-it-yourself advantage – a combination that can't be beat.***

STANDARD FEATURES INCLUDE: • color and mono support • serial/parallel port • 6 expansion slots • 5 or 8 Meg speed switch • PC compatibility

HS-158-M with FREE HVM-122-A monochrome monitor, one 5¼" drive, 640K RAM, diagnostic software and DOS

ONLY $1379*

HS-158-W with one 5¼" drive, 640K RAM, 20 Meg Winchester with controller, diagnostics and DOS. (Monitor not included.) C.ITOH RGB Color Monitor **ONLY $369**

ONLY $1579*

*HUG members get 10% off

Available NOW at Heath/Zenith Computers & Electronics Centers in the U.S.

PHOENIX, AZ
2727 W. Indian School Rd.
602-279-6247

TUCSON, AZ
7109 E. Broadway
602-885-6773

ANAHEIM, CA
330 E. Ball Rd.
714-776-9420

CAMPBELL, CA
2350 S. Bascom Ave.
408-377-8920

EL CERRITO, CA
6000 Potrero Ave.
415-236-8870

LA MESA, CA
8363 Center Dr.
619-461-0110

LOS ANGELES, CA
2309 S. Flower St.
213-749-0261

POMONA, CA
1555 N. Orange Grove Ave.
714-623-3543

REDWOOD CITY, CA
2001 Middlefield Rd.
415-365-8155

SACRAMENTO, CA
1860 Fulton Ave.
916-486-1575

WOODLAND HILLS, CA
22504 Ventura Blvd.
818-883-0531

WESTMINSTER, CO
8725 Sheridan Blvd.
303-429-2292

JACKSONVILLE, FL
8262 Arlington Expressway
904-725-4554

HIALEAH, FL
4705 W. 16th Ave.
305-823-2280

PLANTATION, FL
7173 W. Broward Blvd.
305-791-7300

TAMPA, FL
4019 W. Hillsborough Ave.
813-886-2541

ATLANTA, GA
5285 Roswell Rd.
404-252-4341

HONOLULU, HI
98-1254 Kaahumanu St.
808-487-0029

CHICAGO, IL
3466 W. Devon Ave.
312-583-3920

DOWNERS GROVE, IL
224 Ogden Ave.
312-852-1304

INDIANAPOLIS, IN
2112 E. 62nd St.
317-257-4321

MISSION, KS
5960 Lamar Ave.
913-362-4486

LOUISVILLE, KY
12401 Shelbyville Rd.
502-245-7811

KENNER, LA
1900 Veterans Memorial Hwy.
504-467-6321

BALTIMORE, MD
1713 E. Joppa Rd.
301-661-4446

ROCKVILLE, MD
5542 Nicholson Lane
301-881-5420

PEABODY, MA
242 Andover St. (Rt. 114)
617-531-9330

WELLESLEY, MA
165 Worcester St. (Rt. 9)
617-237-1510

FARMINGTON HILLS, MI
29433 Orchard Lake Road
313-553-4171

EAST DETROIT, MI
18149 E. Eight Mile Rd.
313-772-0416

ST. JOSEPH, MI
2987 Lake Shore Drive
616-982-3215

HOPKINS, MN
101 Shady Oak Rd.
612-938-6371

ST. PAUL, MN
1645 White Bear Ave.
612-778-1211

BRIDGETON, MO
3794 McKelvey Rd.
314-291-1850

OMAHA, NE
9207 Maple St.
402-391-2071

OCEAN, NJ
1013 State Hwy. 35
201-775-1231

FAIR LAWN, NJ
35-07 Broadway (Rt. 4)
201-791-6935

AMHERST, NY
3476 Sheridan Dr.
716-835-3090

JERICHO, LI
15 Jericho Turnpike
516-334-8181

ROCHESTER, NY
937 Jefferson Rd.
716-424-2560

N. WHITE PLAINS, NY
7 Reservoir Rd.
914-761-7690

GREENSBORO, NC
4620C W. Market St.
919-299-5390

CINCINNATI, OH
131 West Kemper Rd.
513-671-1115

CLEVELAND, OH
28100 Chagrin Blvd.
216-292-7553

COLUMBUS, OH
2500 Morse Rd.
614-475-7200

TOLEDO, OH
48 S. Byrne Rd.
419-537-1887

OKLAHOMA CITY, OK
7409 South Western
405-632-6418

FRAZER, PA
630 Lancaster Pike (Rt. 30)
215-647-5555

PHILADELPHIA, PA
6318 Roosevelt Blvd.
215-288-0180

PITTSBURGH, PA
3482 Wm. Penn Hwy.
412-824-3564

WARWICK, RI
558 Greenwich Ave.
401-738-5150

DALLAS, TX
12022C Garland Rd.
214-327-4835

FORT WORTH, TX
6825-A Green Oaks Rd.
817-737-8822

HOUSTON, TX
1704 W. Loop N.
713-869-5263

SAN ANTONIO, TX
7111 Blanco Rd.
512-341-8876

MIDVALE, UT
58 East 7200 South
801-566-4626

ALEXANDRIA, VA
6201 Richmond Hwy.
703-765-5515

VIRGINIA BEACH, VA
1055 Independence Blvd.
804-460-0997

SEATTLE, WA
505 8th Ave. N.
206-682-2172

VANCOUVER, WA
516 S.E., Chkalov Dr.
206-254-4441 – Washington
503-241-3776 – Oregon

MILWAUKEE, WI
5215 W. Fond du Lac Ave.
414-873-8250

Phone orders accepted
OFFER GOOD THROUGH APRIL 15, 1987

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

Heath® ZENITH®
Computers & Electronics

HZC-289

Units of Veritechnology Electronics Corporation

The Fourth Operating System (UCSD Pascal)

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

Most readers are familiar with the Big Three operating systems for the Heath/Zenith 100 series computers (the H/Z-110 and 120.) A recent poll suggested that approximately 70 per cent of the H/Z-110 and -120 owners use MS-DOS (or ZDOS), 20 percent use CP/M-85 and nearly 10 percent use CP/M-86 in one form or another. There is, however, a fourth operating system, conspicuous by its absence, that's used by perhaps one or two out of a hundred owners, and that's the one I want to talk about. It's UCSD Pascal.

When IBM released the first versions of their personal computer, they could choose PC-DOS, which was a version of MS-DOS, for their operating system, or they could choose UCSD Pascal. At that time, in 1981, IBM clearly felt that there was sufficient interest to warrant making UCSD Pascal available, and they expected a sizeable percentage of buyers to choose it. Rather to their surprise, almost no one did. One recent interview with a high-ranking IBM marketing manager revealed that only one or two buyers out of a hundred bought UCSD Pascal in the early years of the IBM PC, and that percentage has dropped even lower today.

When Zenith released the 100 series computers, they followed IBM's lead and made UCSD Pascal available. It wasn't cheap, going for $395.00, and it didn't sell any better with Zenith than it had sold with IBM. Why this happened, and the story of this unique operating system, is the thrust of this article.

Background

First, a bit of history. Pascal, the creation of Professor Niklaus Wirth in the late '60's, was originally initially intended as a teaching language, a classroom construct that would teach students how to create programs in structured, easily understood and maintained ways. Not until 1971 was Pascal implemented on a "real" computer, a CDC 6000. Like most other computer languages of the day, Pascal was batch-oriented (like FORTRAN or COBOL) which meant that it used punch cards for input and massive line printers for output. BASIC had been created by Kemeny and Kurtz only a few years prior, and was not widely used despite its interactive nature.

Wirth had based Pascal on an earlier language called ALGOL, and when Pascal reached the Southern California campus of the University of California in 1973, Professor Kenneth Bowles thought it was a natural follow-on and an excellent choice for the beginning programming course. Initially implemented on UCSD's Burroughs B6700 mainframe, it soon became apparent that many smaller interactive computers would be preferable to one hulking mainframe, and work began on a version of Pascal to run on the PDP-11 minicomputers that abounded at UCSD and elsewhere.

Moving from the mainframe to the mini environment forced the development of a comprehensive operating system that included the Pascal compiler and a set of programming tools including a text editor for creating programs and a file manager for keeping track of them. The entire package became known as the UCSD Pascal System, later shortened to p-System. Eventually other language capabilities were added to the p-System, including FORTRAN and BASIC. However, most people mean UCSD Pascal when they refer to UCSD or the p-System, and so shall we.

Bowles and his co-workers soon realized that having versions of the system for other small computers would be a Good Thing, and came up with the approach of creating a pseudo-machine that emulated the UCSD Pascal system environment. This software would allow the p-System to run in its own language (pseudo-code, or p-code, as opposed to natural machine language, or n-code) with only the low-level (assembly language) interface between the p-System and the computer's hardware having to be written for each particular computer. This confers great portability, and for the first few years it was felt by many that UCSD Pascal would be the standard microcomputer language. (In the late '70's, Carl Helmers at "Byte" magazine became enamoured with UCSD Pascal and made a big push for this standardization, but nothing came of it.)

With Professor Bowles' decision, the UCSD Pascal system was launched. Since it could be adapted to run on a great many mini and microcomputers, it rapidly became a favorite with the new users of these small systems, and by 1979 the UCSD Pascal project had become so large that university lawyers were worried that sales of the system (at $200 per license) would impact on the university's non-profit tax status. A new sponsor had to be found, and after much searching and questioning, SofTech Microsystems of San Diego was selected. Exclusive rights to manufacture and sell UCSD Pascal were sold to SofTech for an undisclosed sum, with the primary concern being that they maintain and improve the p-System, and market it.

In some ways, this was an unfortunate choice. Many industry observers felt that SofTech spent years resting on Bowles' laurels and expecting the money to roll in. While the original UCSD $200 license allowed clubs and other groups to duplicate software and documentation and distribute it free of charge for non-profit use, SofTech wanted $400 for an end-user license, no copying, no distribution, and as it seemed for quite some time, no support. Version II was the standard release for far too long, and Version IV was available on far too few computers.

The IBM deal changed all that. Anything that IBM chose to market through their own distribution channels had to have better support than UCSD Pascal had enjoyed previously, and things got better fast. When Version IV (the latest version) became available on the IBM PC, that meant that most other MS-DOS capable computers could run it, too. UCSD Pascal seemed to have a new lease on life.

The System Described

UCSD Pascal is a complete system. While it's available for many MS-DOS compatible computers, it does not use MS-DOS or any other operating system. When you buy Turbo Pascal, it's assumed that you already have MS-DOS or CP/M, but with UCSD Pascal's p-System everything is there. You simply insert the disk and boot.

The p-System has two immediately obvious characteristics. It's big and it's slow. Big, because it's written largely in p-code, and slow because it's written largely in p-code. Use of p-code requires the use of a p-code interpreter. When your source program is compiled, it's compiled into p-code which is interpreted at run-time. This can be faster than a straight interpreted MBASIC, but not as fast as the same BASIC program compiled, say, with Microsoft's BASCOM. By the same token, the p-code file can be smaller than the ASCII text of the equivalent BASIC program, but when you add in the run-time support required by the p-code interpreter and the overhead of the p-System itself, that advantage goes away. In other words, the great portability gained by the p-code approach is paid for in size and speed. It would not be the language of choice for small software tools that are loaded once, used quickly and put away again. Neither would it be a good choice for interactive video games or other programs where speed of execution are essential.

What is UCSD Pascal good for? In the late '70's a great deal of business-related work was done, partly because it was easier in Pascal than in BASIC, due to Pascal's structure, and partly because of the extended numerical precision UCSD Pascal allows. Primarily, though, the main appeal of Pascal is in learning to program. Pascal, in general, forces the student to adopt structured ways of designing and coding a program, while many other languages allow sloppy habits to form. Both Wirth and Bowles were far more interested in this aspect of Pascal than in the actual production of programs. There are large numbers of programs available for UCSD Pascal, but you should be prepared to look hard, as they're difficult to find. Pascal users' groups are a big help.

On the H/Z-100, UCSD Pascal can directly use only the first 128K of memory, so it will run on a minimal system. If you have more memory, there is a built-in RAM disk that can be made to look like another disk drive for program storage. This speeds up the disk read and write functions wonderfully, but has the obvious drawback that you must remember to save your work to a real disk before you quit. You also need a good deal of RAM; 192K in a standard H-120 won't be too much help.

Booting UCSD Pascal is done just like any other operating system. If you have a winchester hard disk, you can install the system to boot from that. UCSD Pascal takes a bit longer to boot than MS-DOS, but when the process is finished, you enter into the menu-driven environment that is UCSD's hallmark.

Everything you can do under UCSD Pascal is menu-driven with one-letter prompts. I won't go into all the available functions; that would take a book, several of which I've listed at the end of this article. But the initial menu selection looks like this:

```
Command: E(dit, R(un, F(iler, C(omp, L(ink, X(ecute, A(ssem,
         D(ebug, ? []
```

These are the top-level commands (or low-level commands, depending on your point of view.) Everything starts here. If you want to create or revise a program or document, press E for Edit. If you want to compile and run an existing code file, press R. C is for compiling a program that may not already be in your work file. L is for the linker, that links two or more code files into a single file, used especially when an assembly language file must be incorporated in a Pascal file. X is for executing precompiled programs, especially the utilities that come with the system, such as the disk formatting program or the configuration program. A is for assembling 8088 assembly language programs, and D invokes the debugger.

The ? at the end of the line means that there are further options not shown, and pressing it flips to the hidden line. Pressing the ? again returns you to the original line. For the above top-level command line, pressing the ? changes the menu to the following:

```
Command: H(alt, I(nitialize, U(ser restart, M(onitor, ? [IV.13]
```

Halt returns you to the Z-100 monitor (just like a reset), Initialize reinitializes the UCSD p-System, User restart reruns the last file you executed (useful for repetitive runs), and Monitor allows you to create batch files and redirect outputs. The number in brackets [IV.13] refers to the version of the UCSD p-System.

Each level of the p-System has the same menu structure. There are far too many commands to go through here, but you should get the general idea.

Creating a program can be a tedious process until you get used to the procedure. As with any other compiled language, you have to alternate between the editor and the compiler until all the bugs are out and the program works. Since the UCSD p-System is a completely integrated operating system, however, these transitions are easier and faster than on, say, an MS-DOS computer with separate editor and compiler programs. Since everything is menu-driven, it's relatively easy to learn the commands to move from one main function to another.

This integration of functions is a two-edged sword. While UCSD Pascal is protective of the system and generally won't let the user do something that will damage the software and crash the system, neither will it give easy access to the disk operating system for special applications or non-standard operations. This is part of the secret of the portability that Professor Bowles wanted, but it's not a hacker's delight. It's possible to write your own assembly language programs for incorporation into a Pascal program, and it might be possible to patch system software to make it do something beyond the norm, but that's about it.

As you might expect, there's no provision for using the text editor of your choice. Your favorite PIE, WordStar or PTP will not run. In fact, you can't read an MS-DOS disk under the p-System unless you can come up with your own disk read utility; the p-System doesn't include one. This means that other widely used utilities like ZDUMP or EDisk, useful for making bit and byte-level changes directly on the disk, are unavailable. It's possible that the UCSD Pascal Users' Group, USUS, might be able to provide such software, but I have no experience with them.

What do you get when you buy Heath's version of the p-System? For starters, there's a very good general introduction in the form of a 448 page paperback book called "Personal Computing with the UCSD P-System," written by Mark Overgaard and Stan Stringfellow, both of SofTech Microsystems in San Diego. The manual set is admirably complete, and its four IBM PC-format manuals tell you more than you may ever want to know about the system. Printing quality is variable; the Z-100 supplement is very good, but the SofTech manuals are somewhat poor and give the impression of worn master copies. Since the page size is small, this makes the manuals rather difficult to read sometimes. It's not a fatal flaw, however.

There's a manual for the operating system, which includes a Heath/Zenith-specific section to tell you how to get started to the point where the Overgaard-Stringfellow book takes over. There's a two-part manual on program and application development, and one on assembly language programming and interfacing. And there's a manual on the p-System's internal architecture. This is without the best manual set I've seen, far superior to what Microsoft provides for MS-DOS, and even more superior to the Digital Research CP/M manuals. No one who takes the rather considerable time required to read through the manuals could have many questions left. Examples abound, and there are numerous short programs to type in and execute. The utility software is particularly well illustrated. While a beginner might feel overwhelmed by the documentation, a person with a reasonable amount of discipline and interest can learn the entire system without excessive trouble. UCSD Pascal was, after all, intended as a teaching tool.

The software comes on three distribution disks. One is a set of utilities, including such programs as a disk formatter and a configuration program. One disk is pre-configured for two-word (32-bit) real numbers and one is configured for four-word (64-bit) real numbers. Both disks are bootable, so you can start immediately with the disk copy and configuration, described in excellent detail in the Z-100 supplement to the operating system manual.

So far, I've found no bugs in the p-System, and with the exception of turtle graphics, had no difficulty finding what I needed in the manuals. Turtle graphics presented a problem because of the general nature of the documentation. Intended to cover UCSD Pascal for all computers, the specifics of specific machines are missing. As a result, it took quite some time to realize that the graphics capability existed. It's slow to execute, but it's easy to learn how to program. There are examples, but a few demo programs on the distribution disks would have been helpful.

In working with the turtle graphics package, I soon discovered that the ability to run the graphics demo did not mean that the system library contained all the routines needed to program the graphics. When I discovered this shortfall, I wrote both to Heath and to SofTech Microsystems. I have yet to hear from Heath, who as we all know are generally not able to provide technical assistance on "out-of-house" software. However, I heard quickly from Pecan Software Systems, and I learned that they now distribute UCSD Pascal. Apparently, what we all feared when SofTech took over from the University of California happened: they nearly killed UCSD Pascal with limited support, slow development and extremely high prices. Pecan has reversed all these trends; let's hope it's not too late.

Pecan solved my immediate problem by explaining that the version of UCSD Pascal sold by Heath/Zenith does not include the interface section for SCREENOPS (why, we can only guess.) Only the implementation section is present. They also generously sent a copy of the missing section, ready for inclusion in the SYSTEM.LIBRARY. Once installed, turtle graphics programs work just as they should.

I'm very impressed with the support Pecan provides. For one thing, I bought from Heath and registered with Heath, so Pecan didn't know me from Ken Bowles. But they sent the segment anyway. For another, the price of the UCSD system is now $79.95, a highly competitive price. Working with UCSD is more difficult (for me, at least) than with Turbo Pascal, but there's no comparison as far as UCSD's ability to provide a full-service Pascal.

When Heath first introduced Version IV for the Z-100 series, the price was $395. In the past few months, the catalog price has dropped to $99. I suspect this was due to lagging sales, and while Heath hasn't issued their "final call" on UCSD Pascal, it probably won't be too much longer. At the original price, I felt the system didn't have enough to offer, especially with products like Borland's Turbo Pascal available on MS-DOS, an operating system, I already had and knew. At Heath's new price of $99 or Pecan's price of $79.95, however, it was worth taking a chance to see what the p-System was all about.

I have no regrets about buying the package, and found my money's worth in learning a new operating system. Not everyone would agree with this assessment, and I must say in all candor that while the p-System is an excellent learning environment, I can't recommend UCSD Pascal for work on serious applications today. For one thing, it simply is not as portable now as MS-DOS programs have become. If there is a standard Pascal in the micro world, it's more likely to be Turbo, not UCSD.

Reference Books

In addition to the manual set and the Overgaard-Stringfellow book, I've found the following books to be extremely valuable.

The UCSD Pascal Handbook, by Randy Clark and Stephen Koehler. $15.95 from Prentice-Hall, New York, 1982. The UCSD p-System manual set refers to this book often. Highly recommended as a reference guide for programmers. If you buy only one book to help you learn UCSD Pascal, this is the one.

UCSD Pascal, A Beginner's Guide to Programming Microcomputers, by J.N.P. Hume and R.C. Holt, $12.95 from Reston Publishing Company, Reston, Virginia, 1982. An easy introduction to UCSD Pascal and the p-System.

Beginner's Guide for the UCSD Pascal System, by Kenneth L. Bowles, $11.95 from Byte Books (McGraw-Hill), Peterborough, NH, 1980. The original guide to UCSD Pascal, written by the originator. Since the publication of the Overgaard-Stringfellow book, Bowles' guide is optional, but gives a better description in some cases. Note that this book was written to illustrate an earlier version of UCSD Pascal, so there are some minor disconnects.

Pascal Programming Structures, by George W. Cherry. $14.95 from Reston. Not as widely known as Peter Grogono's book, this is the best tutorial I've found for learning the true strengths of Pascal, the structure of the language. Not UCSD-specific, some minor syntax differences must be taken into account; worth the effort.

NEW!! FOR Z-171 USERS
A SPECIAL INTRODUCTORY OFFER ON THE...
DISKIT PORTABLE 20 MEGABYTE HARD DRIVE

Only $895, including interface adapter, controller, cables, chassis, power supply, and high-speed 20 Mb. Winchester disk. The **Diskit Portable** weighs only 4 pounds, is **Z158 Compatible**, and can be carried easily in a briefcase or other carry-on luggage.

- fast access 65-80 ms.
- battery power option
- video board compatible
- 2nd drive option (backup)
- REMEDY Removable Option
- head parking feature
- quick connect/disconnect
- one year warranty
- no tool installation
- 30 ms. high speed option

For information or a demonstration see your nearest Zenith dealer or call Systems Peripherals Consultants at (619) 693-8611.

SPC 9747 Businesspark Avenue, San Diego, CA 92131

NEW!! FOR THE Z-100
THE REMEDY SERIES
REMOVABLE HARD DRIVES

Now you can back up and restore files quickly and easily using standard DOS instructions, maintaining the ability to randomly access and copy individual files. The removable cartridge provides a simple means to transport files between computers or to secure safely when the computer is unattended. The system consists of the drive, one cartridge, controller, and adapter.

- Zenith software compatible
- Z217 direct replacement
- Easy internal installation
- One year warranty
- Multi-drive controller
- 22 or 33 Mb. option
- External cabinet option
- Interchangeable media

Price. . . only **$795,** add $125 for external system. For information, call or write Systems Peripherals Consultants.
SPC 9747 Businesspark Avenue, San Diego, CA 92131 • (619) 693-8611

NEWWORD - the better word processor by NewStar Software

This fine word processor can be considered a clone of the popular WordStar program and it even uses the same commands. However, NewWord is faster, easy to customize, has the mail merge feature built-in, supports more printers, has an UNDELETE function and excellent documentation. It also includes the **WORD Plus** spelling checker by **Oasis** — an excellent product in itself! version 3 includes Macros and other enhancements. We offer this package at a discount. Just check the prices below.

Available in most computer formats including Heath hard sector and BIOS-80.

30 Day Money-Back Guarantee!

CP/M 80 ....$145
MSDOS or CP/M 86 ....CALL
MSDOS version 3 ....CALL

Add $3 per order for shipping and handling.
Terms: Check or Money Order — VISA/MC — COD.
California residents add 6% tax.

ANALYTICAL PRODUCTS **805/688-0826**

213 Teri Sue Lane **Buellton, CA 93427**

Want New & Interesting Software?
Check Out HUG Software

Budget Desktop Publishing

A Review of the FONTASY Page Composition Program

Pat Swayne
HUG Software Engineer

When I was a kid, you could buy little printing press kits, and with some difficulty you could actually compose and print out pages of text with them. It was the dream of some boys (and a few girls too, I guess) to publish their own neighborhood newspapers, and the makers of those presses were happy to help the dreams along to the extent that technology in those days allowed, despite the groans of mothers when they discovered play clothes ruined with printer's ink.

Times and technology have changed. There don't seem to be as many kids interrested in neighborhood publishing these days, but some of the "kids" from the old days who had the dream have not lost it, and so today's neighborhood "newsletter" is likely to be a medium for sharing recipes or distributing the minutes of organizational meetings.

FONTASY comes with a number of sample clip art pictures, including the program logo.

The tool used to produce these newsletters is the home computer. At first, the software used was an ordinary word processing program such as WordStar, and if you had a good (daisy wheel) printer, you could produce a pretty good looking newsletter. The word processing software could handle such things as underlining or bold print to improve appearance, but if you wanted headlines in a larger font, you had to resort to using stick-on letters or something like that. And if you wanted to include pictures on a page and fit the text around them, it took some careful work and possibly some cutting and pasting.

Page Composition Programs

When Apple Computer introduced their graphics oriented Macintosh computer and their laser printer, programs began to appear that allowed you to compose a page on the screen containing different fonts and even pictures, and then print a near typeset quality image of the page. These kind of programs, called "page composition" programs, have been responsible for a new type of home industry called "desktop publishing". Some of the page composition programs are quite sophisticated, and provide a true WYSIWYG (What You See Is What You Get) environment to help you develop a page on the screen before you print it. Page composition programs have made their way to the PC-type computers that many of us HUGgers use, and they have inspired the creation of simpler and less expensive page composition programs for use with dot matrix printers instead of laser printers. Among the first of these budget page composition programs were Print Shop, Print Master, and Newsroom.

Print Shop and Print Master are not designed for doing newsletters, but they can be used to make greeting cards, posters, banners, and calendars that contain different fonts, borders, and pictures. Newsroom is designed for doing newsletters, but the output from it looks rather amateurish, and it does not handle the combination of pictures and text on the page very well.

Enter FONTASY

Now a page composition program called FONTASY (by PROSOFT) is available that is inexpensive ($69.95), works well on an inexpensive dot matrix printer, but that has much of the power and sophistication of expensive page composition programs. Here are some of the features of FONTASY.

Proportional Spacing and Kerning

To make the text on a page resemble text from a printing press, FONTASY uses proportional spacing. That means that small letters will take up less page space than large letters. The following example shows some M's and i's printed with proportional spacing on and off

MMMM iiii   M M M M i i i i

Notice that the four i's in the first group take up much less space than the four M's, but in the second group, the i's take up as much space as the M's. With proportional spacing on, you can get more text on the page with a given font size. To make even better use of page space, FONTASY uses kerning. Kerning causes small letters to be tucked into recesses of large letters, if they will fit. For example, if I print "Te", the e will be tucked under the arm of the T.

Templates

Most of the expensive page composition programs allow you to either compose text directly within the program, or compose it using another editor and feed it into the program. Some of them even accept the codes from some of the popular word processing programs. FONTASY accepts only standard ASCII text from an external editor, but dot commands can be imbedded into the text to control how the text is formatted. In addition, you can design templates that will automatically control how text is fitted onto the page. With templates, you can divide the page into columns and reserve space for headlines and pictures. You can fill in the headline and picture areas of a template so that the headlines and pictures are part of the template, or you can have a general purpose template that just reserves space for headlines and/or pictures. When you load text from an external file into FONTASY, it will fit the text into the areas you have designated, and flow it around any areas reserved for pictures. FONTASY can be set up to automatically load external text, combine it with templates and print the result, even if the document is several pages long, with a different template for each page.

In case you haven't guessed it, the first two pages of this article were done using FONTASY. Everything except the actual text of the article on the first page is part of a template. The text was prepared using the non-document mode of WordStar and then loaded by FONTASY into the templates for the two pages.

Justification And Hyphenation

When you are reading in text from an external file with FONTASY, you can have it justify the lines so that margins are even, or you can have "ragged" margins. You can also justify a line that you have typed directly into FONTASY by typing Alt-J when the cursor is anywhere on that line. When it justifies, FONTASY uses the printing press method of putting more space between words (look closely at the narrow columns in a newspaper) rather than spreading everything out as WordStar does on a daisy wheel printer. If you have a lot of long words, you might get some lines with a lot of "white space" in them. Hyphenating long words can help this situation, so FONTASY provides a way to put conditional hyphens into words. A conditional hyphen will only be printed if a word has to be broken at the end of a line. FONTASY uses a user-definable character (normally `) for the conditional hyphen that you must insert manually into your text. The manual mentions utilities, including some that are public domain, that will hyphenate words for you using your special character, but I have not seen such a program. It would have been nice of them to include one of the public domain versions with FONTASY.

FONTASY Fonts

As the name suggests, the main purpose of FONTASY is to let you print using several different character fonts. Each font is a separate disk file that contains a description of how each character is formed. A "font" in FONTASY is not just a particular style of characters, but it defines the size of the characters as well. Two different sizes of a particular style requires two font files. You can, however, change the size of a font in three different ways.

The first way is a "magnify" command, that allows you to magnify a font by whole numbers, and you can magnify the horizontal and vertical dimensions separately. Once the magnify command has been issued, any text typed or read in from an ASCII file will be magnified to the desired size. The subtitle of this article is in one of the fonts supplied with FONTASY, and the title is that same font magnified by two in each direction. As you can see, resolution suffers when you magnify a font, so it would be better to have a separate font in the larger size.

The second way to change the size of a font is a "size" command that allows you to mark off a block of a page once letters and/or pictures have been placed on it, and alter the size of the block. You can increase or decrease each dimension of the block separately, and by fractions rather than by whole numbers. If you cannot figure out what numbers to use with the size command, you can alter the size of a block manually, watching it change on the screen as you manipulate it. You can also rotate, make a mirror image, or reverse white and black in a marked block. The block commands in FONTASY are very versatile.

The third way to change the size of a font is to magnify it at print time. You can magnify by whole numbers in each direction, and you can also tell FONTASY to attempt to smooth out the "bumps", so that the poor resolution resulting from extreme magnification is reduced. The main purpose of the print magnify capability is to allow you to print banners, and also to correct the aspect ratio of characters on 24-pin printers. The fonts supplied with FONTASY were designed for use with 9-pin Epson compatible printers, and will appear flattened on 24-pin printers unless magnified somehow. To print a banner with FONTASY, you can type out a message in your chosen font, rotate it 90 degrees, and then print it out with a lot of magnification. Personally, I would not recommend using FONTASY for large banners, though, because it prints using a dense graphics mode, and can eat up a printer ribbon before you realize it.

FONTASY comes with 24 assorted font files, which may be enough to get you started if all you want to do is a simple newsletter or some posters, etc. The PROSOFT company also sells a number of additional fonts, which are supplied on disks containing about 8 to 12 fonts per disk, at a cost of $24.95 per disk. They also sell a program called the Letterset Design System, also for $24.95, with which you can design your own fonts. Font disks are discounted if purchased in quantity, including the Letterset Design System disk.

FONTASY Clip Art

FONTASY comes with four files of clip art pictures. Each file contains a number of pictures which can be extracted and placed anywhere on the page you are developing. The pictures can be altered with the size command as they are being placed, so that you can easily make them fit into assigned spaces. One of the picture files supplied with FONTASY contains samples from several other files that are available on separate disks from PROSOFT.

If you want to draw your own pictures, there is a free hand drawing mode available, and also commands to draw lines, boxes, and circles. You can save any pictures you draw (you can save any portion of a page) and combine them into your own clip art files.

The FONTASY Screen Display

FONTASY supports either a standard CGA (Color Graphics Adapter) or a Hercules Graphics card for the screen display. The video display built into most Heath/Zenith PC-compatible computers is a CGA-type display, so it works fine on them. It changes video modes using only BIOS calls, not direct port accesses, so it should also work on any EGA video card capable of multiple video modes. It also works under ZPC on an H/Z-100 dual processor computer without any patches or hardware support.

When FONTASY displays a character or picture on the screen, it does it using the exact number of pixels that will be used to form the character or picture on the printer. Since the dots on the screen are not as dense as those made by the printer, the characters appear much larger on the screen than on the printer. In addition, the aspect ratio on a CGA display is different from that on the printer, so the characters appear taller in relation to their width than they do on the printer. As you may have guessed, only a small portion of a page can be displayed on the screen at a time. On a standard CGA display, you can see 14 lines of text written with the font used in this article, and you can see about two thirds of the width of the page. A status command lets you see a shrunken view of the entire page along with some statistics about the page. The status view is so small that you cannot read anything but the largest fonts on the page, but it does let you get an idea of what the page will look like when it is printed. If you have a Hercules video card, you will be able to see more of the page on the normal screen view, and the aspect ratio will more closely resemble that of an Epson 9-pin printer.

Because the aspect ratio of a standard screen display is different from the printer, circles you draw will not be circular if they are drawn with the correct ratio for the printer. FONTASY allows you to configure the program so that circles, when drawn, will appear correct for either the printer or screen, and allows you to change the aspect ratio of a circle while it is being drawn. You can, therefore, draw ovals rather than circles if you wish, but you may have to test-print them to see what you will actually get.

Supported Printers And Print Density

FONTASY supports all printers that are compatible with the Epson Graftrax standard (including IBM and the TI Omni series), and it also supports the Epson LQ series and printers by C.Itoh, Radio Shack, Microline, Toshiba, and others, and the Hewlett-Packard Thinkjet and Laserjet. The print resolution used with Epson 9-pin and compatible printers is 120 (horizontal) by 72 (vertical) dots per inch, and the supplied fonts are designed to look "right" at that density. Some of the 24-pin printers supported print with a density of 180 by 180 dots per inch, and you may have to magnify fonts vertically to make them look right. The H-P Laserjet (the only laser printer supported) can print at 300 by 300 dots per inch, so even the large fonts come out pretty small on one (print samples are included in the documentation). However, you can make a Laserjet print at 150 by 150, or at 300 by 150. And if you can afford a Laserjet, you can probably afford a page composition program better suited for use with it than Fontasy.

The printer I used to print the first two pages of this article is an Epson MX-80 with Graftrax Plus. With that printer it took about 4 or 5 minutes to print a completely filled page at the normal density. A newer Epson printer, such as the FX-85, would be a bit faster. You can select a "dark" mode, in which the print head makes two passes over each area on the page, and the page is moved up a fraction of a dot between passes. In this mode, it takes twice as long to print a page, but the result is dark solid characters without the "scan line" effect that you get with normal printing. I used the dark mode to print the pages from this article. Another trick that you can do with some 9-pin Epson printers (the FX series) is to print in a "quad density" mode. This density is about equivalent to a 24-pin printer, but it would take very long to print a page, because the print head must make four passes over each area.

Memory Usage

FONTASY composes its pages entirely in memory. You can calculate the amount of memory required for a page by multiplying the horizontal resolution of your printer by the width of your page (in inches), dividing the result by 8 (because of 8 dots per byte), multiplying that result by the vertical resolution, and multiplying that result by the length of your page. Then you have to double that because an extra page is required for an Undo feature. You can see that a lot of memory is required. In fact, there is not enough memory in the 640k DOS limit to build a full page with a 24-pin or laser printer, so FONTASY automatically shortens the length (but not the width) of pages for those printers. You can build up a full page by printing several short pages, or by magnifying vertically at print time. You can defeat part of the Undo feature to make your partial page larger, if you want to.

If you use FONTASY with a 9-pin Epson-type printer and you have 640k of memory, you can build more than one page in memory (each additional page should take about 100k, according to my calculations), and you can transfer data (pictures or blocks of text) from one page to another. FONTASY reserves as many pages as will fit, and the status command will show you how many you have.

A chapter in the FONTASY manual on custom installation tells how you can change the page height and width to anything you want. If you make the page as small as the screen (which works out to about 3 by 5 inches on an Epson printer), you can have up to 25 pages in memory.

FONTASY pages may take up a lot of space in memory, but when they are saved to disk, they take up less space because some kind of compression is used. A page filled with headlines, text, and pictures will usually take up less than 40k of disk space. You can load any page that has been saved as either an ordinary page or as a template. The only difference is that when you load the page as a template, the template settings that were in effect when the page was saved will govern the placement of text added to the page, either from an external text file or from the keyboard.

Documentation And Ease Of Use

It is probably evident by now that FONTASY has many more features than programs like Print Shop and Newsroom (and I haven't mentioned all of them). It is also quite a bit more complex to use than those programs. There are a lot of commands, usually entered using two-character abbreviations, and several Alt- and Ctrl- key combinations. However, there are three levels of operation for novice, intermediate, and expert users that make it easy to get started with FONTASY and learn its features rapidly. The novice level of operation is the menu driven mode. In this mode, you can select any of the two-character commands from a menu, and you can get help on any command. Once you have memorized most of the commands, you can advance to the intermediate method of operation, in which you press the Escape key followed by the two-character name of the command. At the expert level, you can define "Soft Keys" to execute commands that you use frequently. "Soft Keys" are most of the Control-alpha (Control-A, Control-B, etc.) keys or Control-function keys. A single "Soft Key" can be programmed to execute several commands.

FONTASY comes with a 240 page manual in the form of a soft bound book. Not surprisingly, the book was prepared on a laser printer, but not with FONTASY (with the exception of font and clip art illustrations). There are also two reference cards supplied: A keyboard reference card that you can lay across the top of your keyboard, and a command reference card. Two additional booklets are provided which are catalogs of font disks and clip art disks that you can purchase.

A FONTASY Bug

While I was working on this article, I found a bug in FONTASY. On the second page of this article, you will find lines at the top and bottom of the page. These lines were originally to be part of the template of the second page. Notice that the text is fairly close to the lines, especially at the top line. The feature in FONTASY that causes text to flow around drawings (called "variable margins") will not allow text to get that close to a line, so I had to turn that feature off. (FONTASY will print text right over drawings that are in their way if the variable margins are turned off.) The template for the second page worked OK by itself, but when I put everything together so that the template for the first page was used followed by the template for the second page as text was read in, the program did not insert text correctly into the second page. It seems that you cannot have a template with variable margins on followed by one with variable margins off. So I had to design the second page template with variable margins turned on, but without the lines at the top and bottom of the page. After the text was read in, I drew the lines. The FONTASY manual encourages the user to report any problems, so I intend to report this one.

Another problem in FONTASY is that the variable margins feature only works with text read in from an external file, and not with text typed in directly. So if you are planning to have pictures in the text area of your page (as on the first page of this article), you will have to put the text in an external file and then read it in.

Conclusion

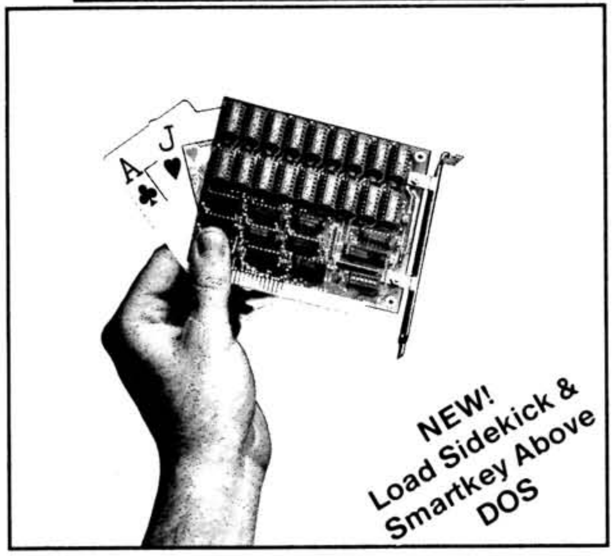
With only a few drawbacks and numerous positive features, I recommend FONTASY as an expensive way to get started in desktop publishing. Not only could you use it to do newsletters, but it would also be good for software documentation or other printed materials required for "cottage industry" operations.

Product Nutshell

Product Name:	FONTASY version 2.07
Manufacturer:	PROSOFT 7248 Bellaire Avenue Box 560 No. Hollywood, CA 91603-0560 (818) 765-4444
Price:	$69.95 (plus $3.00 s/h)
System Requirements:	Any PC-compatible computer with at least 250k of memory (448k or more recommended) and MS-DOS version 2 or higher. Requires CGA, multi-mode EGA, or Hercules compatible display. Also runs on H/Z-100 (not PC) computers under HUG's ZPC emulator.
Printer Requirements:	Runs with Epson Graftrax compatible printers and several others. Contact PROSOFT if you are not sure that your printer will work.

*

The HiCard Wins!



INSTALLS UP TO 896KB TOTAL MEMORY

ADD 64K ABOVE 640KB TO ALL DOS PROGRAMS

INCREASED DOS SPACE FOR NETWORKS, SPREADSHEETS and DATABASES

It's not how much you have, it's how you use it! Take a closer look and see what the HiCard Memory Board can do for you.

THE HICARD 256KB/512KB MEMORY BOARD FOR ZENITH PCs and COMPATIBLES...

...My FRAMEWORK II files run faster than ever, and the HiCard is the perfect pair with my 1 1/2 slot FLASHCARD. *Steve Hutchinson, Engineer Berthoud, Colorado*

- Expands DOS to 704KB
- Supports all application software - no upgrades required
- Works with EGA, networks, accelerators.
- Compatible with Rampage, AboveBoard, and other memory expansion.
- Fills conventional and extended memory
- Top off a PC-AT to 704KB of DOS...plus

Advanced Features

- Install and execute RAM resident programs, network drivers, and other control programs above 640K.
- Multiple print spoolers support up to 3 parallel printers simultaneously while you work
- Multiple RAMdisks can be installed & sized as required
- **Programmers**: Install and execute Code in HiPage™ above DOS

2 Year Warranty on Parts & Labor **Made in U.S.A.**

DESIGNED FOR IBM PC/XT/AT AND ZENITH COMPATIBLES RUNNING DOS 2.0 OR LATER

*HiPage Utility Software included

Call your local computer dealer or

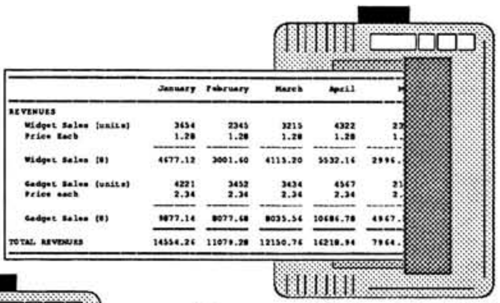
RYBS ELECTRONICS, INC.

303-444-6073
303-444-R Y B S
P.O. BOX 4521
BOULDER CO 80306

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.

T E C H N O L O G I E S, I N C.

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%)

Weaving Spells With Spellbinder

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

One thing is certain. As new computer systems pile up in your office or workroom, so do word processors. I've been collecting them over the years the way my wife collects books on Faulkner, and I probably have more software in this one category than all other programs put together.

I suppose it's like conducting a search for the ultimate word processor. The one that will solve all your needs. The easiest to use. Flexibility. Features. Good documentation. I haven't mentioned price because it should only be a consideration in relation to the features and versatility of the program.

Take PIE and PFS: WRITE. PIE is a simple text editor that allows your screen to act as a typewriter. Editing commands are limited, but for $30 I have gotten more use out of PIE in grinding out letters and short articles than you could ever imagine. PFS: WRITE, which was created for the IBM-PC and which was adapted in a modified form to become IBM's Writing Assistant, is $140. Like PIE, it forms a typewriter out of your screen — somewhat more sophisticated because it uses color, help menus and a couple of other doodads. But if I had to buy it, I would think twice about it, because it doesn't offer a whole lot of additional features for an additional $110.

A few months ago, I began using Spellbinder. Some of my occasional writing projects required a full-blown word processor, and I had grown weary of Magic Wand and WordStar. I looked forward to working with it, though not without some trepidation caused by the usual anxieties of learning hundreds of strange commands in a short period of time.

Spellbinder is marketed by Lexisoft, and is one of the throwbacks to the early days of computing. It originated in 1977 when Perry Gee joined a company named Testan Scientific, makers of scientific instruments, which was owned by John Bintz. Personal computers were beginning to arrive on the marketplace, and Bintz and Gee saw an opportunity to gain a share of the market if they could create a word processor that the average user would be comfortable working with.

They launched their fledgling in 1978 under the name of AUTOTYPE. It was also bundled with the old Exidy Sorcerer as the Word Processor Pac. Spellbinder has come a long way in the last 8 years. Not only has it undergone numerous revisions and adaptations so that there is a compatible version available for almost every computer and operating system available on the market, but it has grown beyond being just a mere word processor. More than 21 OEMs have had license agreements to market Spellbinder with their computer systems, among them Hewlett-Packard (as WORD 125), Eagle Computer (as EAGLEWRITER) and Xerox Corporation (as the XEROX OFFICE MANAGEMENT SYSTEM).

More than any other type of program on the market, the merits of word processors are debated in the magazines and at computer club meetings mostly for subjective reasons. One user's excitement is another's irritation, and the debate goes on. Some publications, like PC MAGAZINE, seem to have a fetish about word processors. They like to group them together every few months, and hire a lot of high-priced talent to tear them apart, make fancy charts of their features that are as legible as the hiking paths on the Appalachian Trail. In one issue they compare them one way, in another issue they have seemingly discovered a better way.

While reading these comparison reviews is occasionally interesting, I find them all too often much too confusing. It's almost like walking into a computer store and trying to decide which of a dozen programs to buy without really understanding how they work.

I had no preconceived notions about Spellbinder when I started working with it, and I kept a modest journal in which I recorded observations as I went along. My primary objective was to work with it from start to finish in order to determine if I could live with it as my primary writing tool.

What Is Spellbinder?

Spellbinder is marketed as a word processor and office management system, claiming to be able to perform most of the tasks a business would need to conduct its correspondence, mailings, modest database tasks and some other features.

It is nicely packaged in one of those half-sized padded vinyl binders that fold into an easel. The manual is in excess of 400 pages, including a host of Appendices that we will discuss as we go along. I worked with the CP/M-80 version for the H/Z-89 computer system. It is also available under CP/M-85, CP/M-86 and MSDOS for the Z-100 series; the IBM version is available for the Z-150 PC series. The number of disks you receive depends on your disk format. I received 6 hard-sector disks: 3 for the word processing program and 3 for Electric Webster, which is a spelling checker program to proof your text. It is also capable of reviewing your grammatical errors. And, it incorporates a series of macro programs that allow you to create templates for special forms.

Getting Started With Spellbinder

A nice feature of Spellbinder that gives it an automatic plus in my book is that it is not copy-protected, so you can make back-up copies for yourself. I refuse to buy copy-protected programs, unless I can obtain a copy program that will allow me to create a working back-up program without corrupting the program files.

The first step after making a back-up copy, is to create a customized working disk with Spellbinder's installation program. The CONFIG program establishes a number of important parameters:

1. You can enable the special function keys of the '89s Terminal board so that you can use them for various editing functions. This allows you to use 23 keys for cursor control, scanning, indenting, mode enhancing, tabs, delete, repeat, etc. All of which avoids the need to use a CTRL key sequence. Lexisoft offers a set of replacement keycaps for the '89 which are worth obtaining for $35, because they give you a quicker confidence while working in the EDIT mode.

2. Spellbinder has the ability to work with nearly every dot matrix and letter quality printer on the market. You select your specific printer by entering the corresponding number from the screen display. If you have both types of printers, you should configure a version for each in order to take advantage of the proportional-spacing capabilities of the letter quality printer.

3. If you have a photographic memory, you can elect to disable the on-screen User Guides that remind you how to perform various functions. User Guides usually clutter the screen and cut down on the number of text lines displayed, but with Spellbinder they're not as densely populated as WordStar. They're kept at a minimum because Spellbinder creates a Command driven environment, rather than a Menu driven format. If you're finished working on a document and have forgotten the Save-to-Disk command, you can type CTRL/Q to toggle the screen into the Command mode and obtain the data you need from a series of menu and sub-menu screens.

3. You can elect to have Line and Row/Column numbering screen.

4. You then enter the number of drives on your system, including floppies and hard disks.

5. Unlike some CP/M software for the '89, Spellbinder can be configured to operate under 2 or 4 MHz.

6. Lastly, if you have specific requirements for your printed text, you can set up the "Y" and "YT" tables for printer formatting by establishing more than 60 parameters including:

   Printer Type
   Print Length
   Form Length
   Page Eject
   Left Indent
   Spacing
   Justification
   Line Width
   Line Feed Size
   Character Size
   Special Characters
   Proportional
   Top Title
   Top Spacing
   Bottom Title
   Bottom Spacing
   Page Formatting
   Page Numbering

   This is not the complete list, nor does it show all the options within each category that can be manipulated. Since Spellbinder has obviously been designed to handle almost any printing requirement, additional disks can be created for Memos, Forms, Reports, etc., each having all printer parameters specified and saved to the disk. Of course, you can elect not to create a permanent disk setup, and issue these commands when you are ready to print your document. This, of course, steals time from your productivity. The "Y" and "YT" tables do not contain all the options you may need. Additional formatting is established by imbedded commands for tabbing, enhanced, underlined or emboldened printing of words and phrases. So it does resemble a mini printshop in terms of allowing you to use your printer's available features to the limit.

Once you have a system working disk configured, label it to identify the type of printer options you saved to the disk and then make another copy for your actual daily use. Because Spellbinder contains a countless number of files, the disk you created will always remain in drive A:, and your text files will be saved on drive B:.

Like most professional word processors, a carriage return is not issued unless you wish to terminate a line (such as in an address or a column listing) or to end a paragraph. Words do not break at the extreme right. If you type a word that is larger than the available space on the line, it is automatically transferred to the next line.

The manual is laid out as a continuous working tutorial. That is, you work your way through nearly 160 pages that take you step-by-step through all the editing commands, imbedded printer commands, reading and writing of text, moving text on screen, working with your drives, covering a seemingly endless array of commands. But after you have worked with Spellbinder for a few days, you'll discover about 18–20 basic commands that you will use most of the time. I wrote these on a small index card and taped them on the front of the computer so that I was to dispense with the manual.

This is not to slight the Spellbinder tutorial section. The text is well written — in fact, more amiable than most others. But it is obvious that Lexisoft took into consideration that a large segment of users would be neophytes and decided to hold their hands during the learning process. I think what made me impatient was that the tutorial text was too long, and that an editor's pencil could have cut it in half without any serious detriment to learning.

The manual is printed in a light gothic face, with all key commands to be learned in the tutorial printed in a bold gothic. Since a great many commands are either the CTRL key plus another key, or a series of two letter commands, these are printed in lower case bold that somehow gets lost on the line. However, the mini-chapters (some only a page or two in length) end with a summary of all the commands just covered with a short description of their function. This aided me in the skimming process of learning the operating commands. I might have been more kindly disposed if these command letters were capitalized. But as this was the only criticism I could muster after several months of working with Spellbinder, it may sound that I'm being a little picky.

Working With Macros

What makes Spellbinder unique is that it is more than just a word processor. Its Macro programs allow you to perform a variety of useful tasks that not only enhance the word processing abilities, but allow users to create databases, mailing lists, design boilerplate forms, do sorting according to specified fields, print texts in two columns, create customer lists according to product categories, prints mailing labels, shift sections from one text file to another, as well as performing calculations (addition, subtraction, multiplication and division) on numbers that are already in your text or that you enter from the keyboard.

The Macro files supplied with Spellbinder are general purpose that can serve a variety of needs as I mentioned in the previous paragraph. If you have highly specialized needs in terms of creating databases or the need to combine several functions, Lexisoft has the Spellbinder Technical/Macro Manual available for $60 that teaches you how to create special macro programs that are prepared as ASCII text files.

Macro programs that define specific fields are created out of need and imagination once you have established just what it is you want your program to do. For example:

```
t/w/wd/d
```

means "go to the top of the document, write everything in workspace to disk, close the write file and clear the screen." While this is a simplistic example of a simple macro program, all macro programs are prepared in this format in Spellbinder's Edit mode and then saved as a .WPM file.

Computer Resources of Waimea (P.O. Box 1569, Kamuela, Hawaii 96743), better known as CROW, has a series of disks available that contain a host of ready-to-use macro programs. Each disk contains several programs and costs $29.95, and purchasers are encouraged to copy these disks for their friends. If any recipient of such a disk finds the programs useful, he is asked to send $10 to CROW to become a registered member and will automatically receive information on new macro program disks.

In addition, Becky Winter of CROW, has written an 81-page summary of Spellbinder commands called "All You Wanted To Know About Spellbinder But Couldn't Find Out". I don't know what the price is. I gleaned the information from one of the bimonthly "Dealer Notes" newsletters distributed by Lexisoft.

Electric Webster

Until about a year ago, this 50,000 word spelling checker was sold separately. It has since become an integral part of the Spellbinder Word Processing & Office Management package. Like Spellbinder, Electric Webster requires that you configure the working disk you prepare to recognize your system — the operating system and version you are using and the customization of the hyphenation feature.

When you specify the file to be proofed, Electric Webster will display the total number of words in your document and the total number of different words. This was my first exposure to a spelling checker and I enjoyed using it. It managed to proof a 15-page document in a matter of seconds and isolated my typing goofs. After spending hours in front of my green screen, banging out letters and articles, my eyes have a tendency to skip over an obviously mistyped word, no matter how embarrassing the typo looks.

The grammar checking ability of Electric Webster groups potential errors into categories like homonyms, slang, and passive verb construction. Each time it presents a phrase for your consideration, the grammar checker displays the category of that particular phrase. It will also suggest replacement words which you can instruct it to insert into your text.

It will also present you with a summary of statistics:

Average word length
Average phrase length (in words)
Average sentence length (in phrases)
Average paragraph length (in sentences)
Average sentence length (in words)
Number of long words (percentage of document)
Number of long phrases (percentage of document)
Number of long sentences (percentage of document)
Number of long paragraphs (percentage of document)
Number of marks (^) placed in text

So, if you write short, simple sentences like Hemingway, or wordy phrases like Faulkner, I'm sure Electric Webster will display a list of suggestions for you.

Summing Up

Spellbinder has a series of useful appendices. There is a Quick Reference Guide to all Edit Mode Functions, Command Mode Commands, Table Settings, Dot Commands and In-line Commands in a compact 13 pages. I would have liked to have this printed on a sheet of folded card stock to keep next to my computer rather than the bulky manual. All major programs distributed and customized by Heath/Zenith include this and I appreciate the convenience of it.

There is a Glossary of words and phrases used in Spellbinder and corresponding explanations. Most seasoned users will ignore this.

Another appendix lists the Terminal Function Keys for the computers and terminals supported by Spellbinder. Replacement keycaps are also available for the Z-100. Interestingly enough, the '89 has 6 more usable function keys available than the Z-100.

There is also an interesting section on Special Printer Applications that assist you in getting the most out of your dot matrix or letter quality printer, in precision and nonprecision modes, using advanced printer features and ASCII charts.

Finally, there is a list of Spellbinder error messages and their meaning, and a comprehensive Index.

To keep users current on new uses for Spellbinder, Lexisoft issues periodic "Spellbinder Application Notes" that offer tips on Writing Macro Programs, notes for Professional Writers, Technical Writing, Mass Mailing & Individual Correspondence, Academic Writing & Office Management, Automatic Footnote & Pagination, Time Management, Construction Office Management, Advertising/Public Relations Offices, Forms Handling, Organizations, Sorting & Selecting, Medical/Dental/Veterinarian Offices, Insurance Offices, Real Estate Offices, Retail Stores, and so forth.

Viewing Spellbinder from an overall perspective, it is very impressive, powerful and productive. It has its little quirks, but I have found these to be more subjective than universal. I have now used it on and off for more than five months and my enthusiasm is still high, as there are always new uses to be discovered.

Lexisoft offers versions of Spellbinder for Law Offices and Scientific use. They have a decent upgrade policy that allows registered owners to obtain updates at nominal charges. In addition, if you started out using Spellbinder on the '89 and then acquired a Z-100 or Z-150 PC, you can obtain specific versions for these systems for only $75-100. There are several decent media conversion programs that will allow you to transfer files between these and other computers, so that Spellbinder can easily become a universal program for most word processing and database needs. I should make mention of the fact that Spellbinder also interfaces easily with most major database programs, including dBASE II.

Spellbinder In A New Disguise

A short while after I started working with Spellbinder, I learned that Lexisoft has again performed some interesting legerdemain with the release of Desktop Publisher. This $650 program has been designed to work only with the new Hewlett-Packard LASERJET+ printer.

The printed samples I saw were nothing short of amazing. The LASERJET+ has the ability to print typeset quality documents, and Spellbinder has enhanced its basic word processing program with a series of 16 different typefaces and symbols so that the two can function on the same level as phototypesetting equipment.

Different type styles can be mixed on the same line while maintaining full justification. Moreover, the computer screen displays the text you are preparing on a what-you-see-is-what-you-get basis. That is, enhanced type, boldface, super or subscript characters, ornate initials positioned as chapter lead-ins, italics, underlining, serif type, textured screens, rules, borders, headers, footnotes, etc. are shown exactly as they will be printed. In addition, bar graphs of nearly any complexity can be designed and printed. Outlines as small as 1/100th of an inch can be controlled. There is also a zoom mode that lets you focus on any finely-detailed area up to 128x the original size.

One can prepare quality brochures, flyers, even magazines with Desktop Publisher and LASERJET+ that will look as though they had been produced at a professional printshop.

Obviously, this combination of hardware and software is geared for the corporate or professional marketplace. The Spellbinder Desktop Publisher costs $650. The Hewlett-Packard LASERJET+ costs in excess of $2000. You will also need an IBM-PC or compatible.

* * *

The Spellbinder Word Processing & Office Management System has a list price of $495. For additional information, please write to:

Lexisoft, Inc.
P.O. Box 1378
Davis, CA 95617
(916) 758-3630

Except for updates and other versions, Spellbinder is sold only through an international network of dealers. One Heath/Zenith dealer who got me started on Spellbinder and who discounts it for a pleasing $295 is:

Henry Fale
Quikdata Computer Services, Inc.
2618 Penn Circle
Sheboygan WI 53081

Function Key Assignments & Notes On Spellbinder's Command Structure

Heath/Zenith 19/89/90

Key	**Function**	**Replacement Keycap**
Up Arrow	Cursor Up	Up Arrow
Down Arrow	Cursor Down	Down Arrow
Left Arrow	Cursor Left	Left Arrow
Right Arrow	Cursor Right	Right Arrow
Home	Scan	Cursor Scan
F1	Indent	Indent
F2	Soft Hyphen	Soft Hyphen
F3	Mode Enhance	Mode Enhance
F4	Enter Enhance	Enter Enhance
F5	Previous Page	Prev Page
Erase	Next Page	Next Page
Blue Square	Continue	Continue
Red Square	Mark	Mark
White Square	Line Top	Rewrite
0 (keypad)	Edit/Command	Edit/Comm
. (keypad)	Decimal Tab	Dec Tab
Enter	Insert	Insert
IC	Cursor Mode	Cursor Mode
DC	Mode Delete	Mode Delete
IL	Mode Back	Mode Bkward
DL	Mode Forward	Mode Fward
Delete	Delete	Delete Txt
Repeat	Repeat	

Heath/Zenith 110/120

F0	Edit/Command	Edit/Comm
F1	Indent	Indent
F2	Enter Enhance	Enter Enhance
F3	Previous Page	Prev Page
F4	Next Page	Next Page
F5	Soft Hyphen	Soft Hyphen
F6	Line Top	Rewrite Top
F7	Reline Screen	Reline
F8	Decimal Tab	Decimal Tab
F9	Mode Back	Mode Back
F10	Mode Forward	Mode Forward
F11	Mode Enhance	Mode Enhance
F12	Cursor Mode	Cursor Mode
I CHR	Mode Delete	Mode Delete
Ins Line	Insert	Insert
Home	Scan	Scan
Help	Mark	Mark

Notes On The Command Structure

The Spellbinder command structure can be intimidating. There are approximately 150 commands covering Cursor Movement, Editing, the Command Mode, Text Movement and Deletion Commands, Read/Write Commands, Search Commands, Printing Commands, and some miscellaneous provisions. Also included in this grouping are in-line DOT DOT commands that relate to text printing and changing character modes.

In addition, there are more than 60 formatting commands (called Table Settings) to set up your printed text. The Macro abilities of Spellbinder have their own specific Commands, as well as incorporating some from the word processing command structure.

This bulk has been compiled into separate groupings in an 11-page Quick Reference Guide, called Appendix A.

Since the Spellbinder manual lacks a fold-out card that can be placed next to the computer, I simply Xeroxed this section and used a yellow highlight marking pen to isolate the more commonly-used Editing and Read/Write commands. All of this information, of course, can be called up directly through the CTRL-Q toggle switch as I mentioned within the body of my article. This reduces the "What do I do next?" dilemma.

One interesting fact struck me as being somewhat significant. When I first become involved in learning a sophisticated program and accidentally issue more than one command in sequence, some programs have a tendency to simply lock up and die. I have noticed this peculiarity with Microsoft WORD, LOTUS 1-2-3 and SYMPHONY. The lockup is so complete that a CTRL-ALT-DEL doesn't even work and my PC has to be reset or turned off and on again. Spellbinder seems relatively immune to this problem and is more kindly disposed toward issuing an error message or just switching from the Command Mode back to the Edit Mode allowing me to re-enter the desired keystrokes without risking the possibility of data loss.

✱

IMAGE ACE II
Video Capture System

$295.00

— complete —

For PC Compatible Computer Systems

- single slot card
- software included
- complete manual
- video cable

Digitize video images from cameras, television tuners, and video recorders directly to your PC compatible display

- No additional video monitor needed
- 320 x 200 x 4 levels B/W or pseudocolor
- 1.3 second capture time
- Works with IBM-PC/XT/AT or PC compatible with standard color graphics adapter.
- Software included to capture and save images in DOODLER-V .PIC format
- Includes stand-alone software to capture, edit, save, and print digitized images.*
- Transfer captured images to Z-100 using DOODLER-V

Full 90 day manufacturers warranty

IMAGE ACE II Video Capture System manufactured by Lodge Electronics

Available from...

PAUL F. HERMAN Inc.

Software Graphics Tools
3620 Amazon Drive
New Port Richey, FL 33553

C.O.D. Orders accepted... Call 813-376-5457

*Software provided by Lodge Electronics is written in BASIC, and includes source code. User modifications for specific applications are easily done with CW-BASIC.

TaxAide

Income tax preparation worksheets for use with your Lotus 1-2-3.

● PRODUCES IRS APPROVED PRINTOUTS - NO SPECIAL FORMS TO BUY ●

● Easy to Use — Menu Driven — Integrated ●

● Screen Displays Resemble Actual IRS forms ●

● Automatically Calculates Income Tax ●

● Full Technical Support — Satisfaction Guaranteed ●

TaxAide Plus — Provides worksheets for Forms 1040, 1040A, 2106, 2119, 2210, 2441, 3468, 3800, 3903, 4562, and 6251; Schedules 1, A, B, C, D, E, F, G, R, SE, and W; plus IRS worksheets. Multiple Forms 2106 and 4562 and Schedules C and SE. TAXAIDE PLUS PRODUCES IRS APPROVED PRINTOUTS FOR ALL FORMS AND SCHEDULES, INCLUDING 1040 AND 1040A. Ideal for income tax preparers. **$59.95**

TaxAide Personal — Provides worksheets for Forms 1040, 1040A, 2106, 2441, and Schedules 1, A, B, D, G, R, and W, plus IRS worksheets. Two Forms 2106 are provided. Produces IRS approved printouts (except Forms 1040 and 1040A). **$29.95**

REQUIRES ONLY 192K of RAM — If you have more memory, you may still run your background program such as SideKick or Perks.

Requires Lotus 1-2-3 Release 1A or later, and DOS 2.0 or later.

To order, send check to:
Kansas residents add
5% sales tax

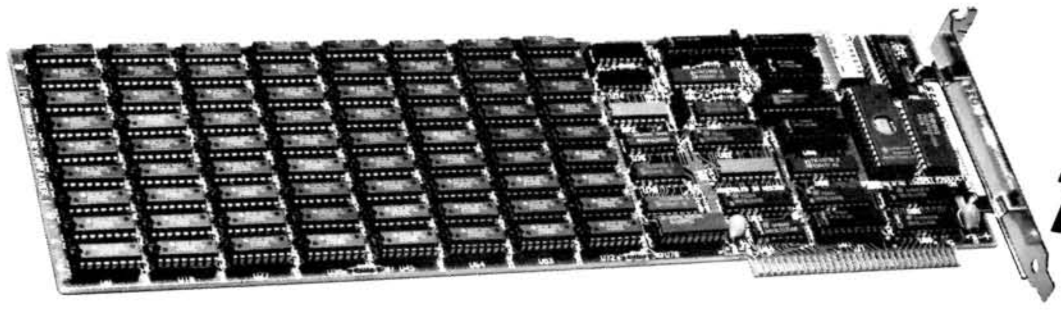
Software Applications of Wichita
2204 Winstead Circle
Wichita, KS 67226

For VISA or MasterCard orders, call (316) 684-0304.

TaxAid is a trademark of Software Applications of Wichita.
Lotus and 1-2-3 are registered trademarks of Lotus Development Corp.
SideKick is a trademark of Borland International, Inc.
Perks is a trademark of Barry A. Watzman.

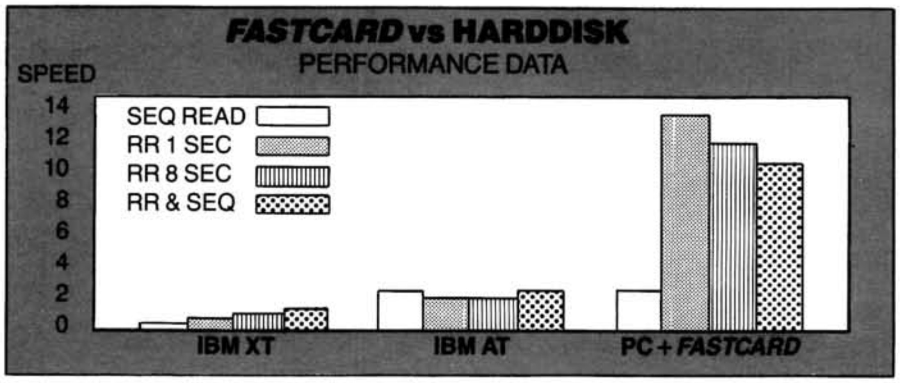
Expand your ZENITH PC Memory with FASTCARD®

fully loaded with 2 *MBYTES* for just $295*



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

One Little, Two Little, Three Little Pixels

Jim Buszkiewicz
HUG Managing Editor

The lightpen has become one of the most popular and easy to use input devices ever to be interfaced to a home or business computer. Instead of typing in a function from the keyboard, or positioning a little arrow to that same function, you simply point at the function on the screen with the lightpen. For most applications, selecting a small area on the CRT is sufficient, but what if we want to select a single pixel (a pixel is the smallest addressable point on the screen)? Our pen (remember, we all bought one from the Lite-Pen Company a few months back) is surely capable of single pixel resolution, but is the electronics part of the lightpen port? Maybe, but the Lite-Pen Company's Pixel-Plus (tm) card eliminates all doubt.

The Pixel-Plus is designed to provide computer graphics with superior resolution and accuracy. This half wide adaptor card enables lightpen functions at a single pixel level in text, CGA, and EGA modes for resolutions up to 1024 × 1024! In addition, this card provides a lightpen port should your system not have one. Included with the card is corresponding driver, installation, and demo software. The card is easily installed in one of the backplane slots of your PC compatible computer (NOT H/Z-100). A short 9-pin video cable (supplied with the card) is then connected between that card and your color or hi-resolution monochrome card. The 9-pin connector from your monitor is then plugged into the Pixel-Plus card. As you can now guess, by the connection scheme, the Pixel-Plus card now has access to all the sync and video signals going to the monitor. With these signals, its onboard logic can now (very accurately) pick out individual pixels with the lightpen connected to it. Also on the back of the card is a standard telephone connector to which the Lite-Pen Company's standard lightpen can be connected. So as not to conflict with the address space of any other peripheral board, the Pixel-Plus card is configurable (via dip switch) to any one of 256 different locations. The card comes properly configured and will not conflict with existing IBM lightpen port assignments.

The manual included with the Pixel-Plus card was quite thorough in its explanation of how the card was to be installed, the basic theory of operation, and how programs could be written to work with the card. In addition, the demonstration software supplied was written in BASICA, which the user can study, so as to learn how to write his own application software.

This magic card normally retails for $159 from the Lite-Pen Company. However, HUG has managed another super deal for its members. If you decide to order the Pixel-Plus card, simply mention that you're a HUG member, and you'll get 50% off! That's right, HALF OFF! The Pixel-Plus will only cost you $79.50. For further information, or to order, contact the Lite-Pen Company, P.O. Box 45255, Los Angeles, CA 90045-0255, (213) 305-7616.

-- 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

Sines And Cosines And 2–D Graphics On The Z–100

Part II

Thomas J. Vaden
5765 Grand Avenue
Riverside, CA 93504

Introduction

In Part I, we introduced the relationship between graphs of mathematical functions and the transformations necessary to display these graphs on the screen. These transformations included the translation and scaling of display areas. In addition to plotting simple math functions, we plotted and rotated a square. We also covered the use of the aspect ratio to control distortion of graphical displays, and the balance between the resolution of graphs and computational speed.

This month, we will discuss the relationships between character and graphical display spaces. We will then extend the graphics to include rotations of various points, lines and shapes around arbitrary centers in the two-dimensional plane. The combination of translations and rotations will provide us a measure of control over the movement of shapes, and a tool to create a variety of visual displays.

Character Screen Vs Pixels

The character screen is divided into 25 rows and 80 columns with each character occupying a space eight pixels in width and nine pixels in height. Thus a full screen is 640 pixels across (80 characters × 8 pixels per character) and 225 pixels down (25 characters × 9 pixels). A rectangular box 72 pixels by 72 pixels holds nine characters across and eight characters down. A space 10 characters across by three characters down occupies a rectangle 80 pixels across by 27 pixels down.

The location of the upper left corner of a graph (in pixels) can be tied directly to the character display by the following formulas:

```
LX = 8 * (COL-1)
LY = 9 * (ROW-1)
```

For example, to locate the upper left corner of a display space at ROW 8, COL 5 on the character screen, we set

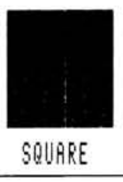
```
LX = 8 * (5-1) = 32
LY = 9 * (8-1) = 63
```

Program 2-1 demonstrates an integrated use of characters and graphs by plotting and labelling several regular polygons on the same screen. Since program 2-1 is designed for the color monitor, the Aspect Ratio is set equal to 0.4844. Ten polygons are arranged in two rows and five columns. Each polygon occupies a square display space 104 pixels (13 characters) across by 51 pixels down (almost six characters). The 51 pixels down was obtained by multiplying 104 pixels by the Aspect Ratio.

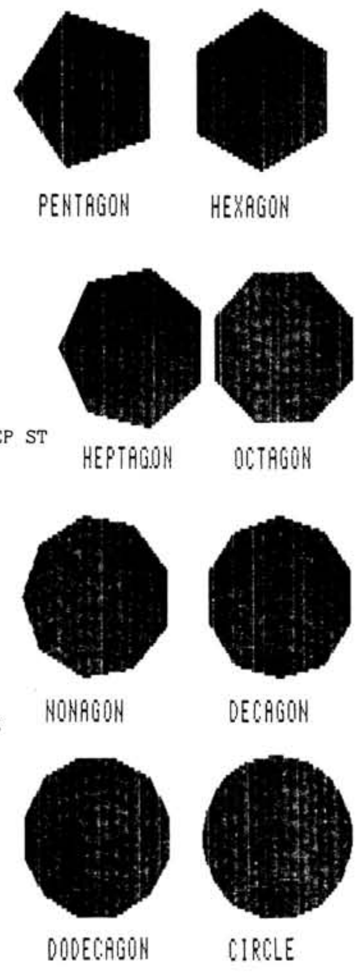
Formulas in program 2-1 tie the upper left corner of each polygon display space to a character display area 15 characters/columns across and 10 rows down. The polygon display occupies the first 13 columns and 6 rows of the character display area. The title for each polygon is located in the eighth row of the character display area. In the data statements, each title is centered within a 12 character title space.

Program 2-1

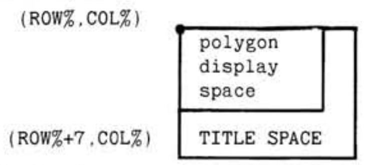
```
10 ' PROGRAM 2-1, LOCATES AND DRAWS POLYGONS
20 '*****************************'
30 '*****    THOMAS J.VADEN   *****'
40 '*****    COPYRIGHT 1985   *****'
50 '*****************************'
110 CLS
120 DX=104
130 DY=.4844*DX
310 PI=3.14159
340 RX=2
350 RY=2
```



```
360 SX=DX/RX
370 SY=DY/RY
500 FOR R%=1 TO 2
510 FOR C%=1 TO 5
550 READ NS,A$,CF
560 ST=2*PI/NS
570 COLOR CF
610 ROW% = R%*10-8
620 COL% = C%*15-12
630 LX% = 8*(COL%-1)
640 LY% = 9*(ROW%-1)
650 CX = DX/2+LX%
660 CY = DY/2+LY%
670 S1=  PI/NS
680 S2=2*PI+S1
700 X1=CX + SX*COS(S1)
710 Y1=CY - SY*SIN(S1)
1000 PRESET (X1,Y1)
1010 FOR I = S1 TO S2+ST/2 STEP ST
1030 X= CX + SX*COS(I)
1040 Y= CY - SY*SIN(I)
1050 LINE-(X,Y)
1099 NEXT I
1200 PAINT (CX,CY)
1250 LOCATE ROW%+7,COL%
1260 PRINT A$
1500 NEXT C%
1510 NEXT R%
1600 COLOR 7
1999 END
2000 REM - SIDES  SHAPE  COLOR
2010 DATA  3,"  TRIANGLE   ",1
2020 DATA  4,"   SQUARE    ",2
2030 DATA  5,"  PENTAGON   ",4
2040 DATA  6,"  HEXAGON    ",5
2050 DATA  7,"  HEPTAGON   ",6
2060 DATA  8,"  OCTAGON    ",1
2070 DATA  9,"  NONAGON    ",2
2080 DATA 10,"  DECAGON    ",4
2090 DATA 12,"  DODECAGON ",5
3000 DATA 30,"  CIRCLE     ",6
```



Display Space:



Each polygon space is located on the character screen using the following formulas:

```
COL% = C%*15 - 12
ROW% = R%*10 - 8
```

For example, to locate the upper left corner of the first polygon on the character screen we set

```
COL% = 1*15 - 12 = 3  (Column 3)
ROW% = 1*10 - 8  = 2  (Row 2)
```

The next polygon to the right is located in column 18 of the character screen

```
COL% = 2*15-12 = 18
ROW% = 2
```

The location of the center of each graph is defined in pixels (lines 630-660), and the horizontal orientation of the graphs is determined (lines 680-690). The graphs are then scaled and placed on the screen. The titles for each graph are located in the eighth character row (lines 1250-1260) using the LOCATE command.

```
LOCATE ROW%+7,COL%
PRINT "    TITLE     "
```

Rotation Of Shapes (Polar Coordinates)

In practice, we would like to rotate various assortments of points, lines, and shapes on the screen, without distortion. For example, we might want to draw an airplane and fly it around in a circle, or graph some letters and rotate the letters on the screen. Most shapes can be defined by connecting the corners or vertices with straight lines. If a shape has curved lines, additional points are needed to fit the curvature.

In this section, we will draw a simple figure on graph paper and define the vertices of this figure in terms of polar coordinates (see Figure 1). The locations of the vertices relative to a center point are defined in terms of the angle (in radians) and the distance from the center. If you are accustomed to defining the angles in degrees, you will need the formula A = A*PI/180 to convert the degrees to radians. For example, 90 degrees = 90*PI/180 radians or PI/2 radians. The polar coordinates for the vertices in Figure 1 are:

VERTEX	RADIUS	ANGLE
A	50	0
B	20	PI/2
C	50	PI
D	10	5*PI/4
E	0	0
F	10	7*PI/4

The x and y coordinates of each vertex are then located using the sine and cosine formulas. For example, vertex 3 is

```
X3 = COS(PI)*50
Y3 = SIN(PI)*50
```

These coordinates are then scaled (SX,SY) and assigned a screen location relative to a center point (CX,CY) on the screen. Finally, an angle of rotation (A) is added to the vertex angle.

```
X3 = CX + SX*COS(PI+A)*50
Y3 = CY - SY*SIN(PI+A)*50
```

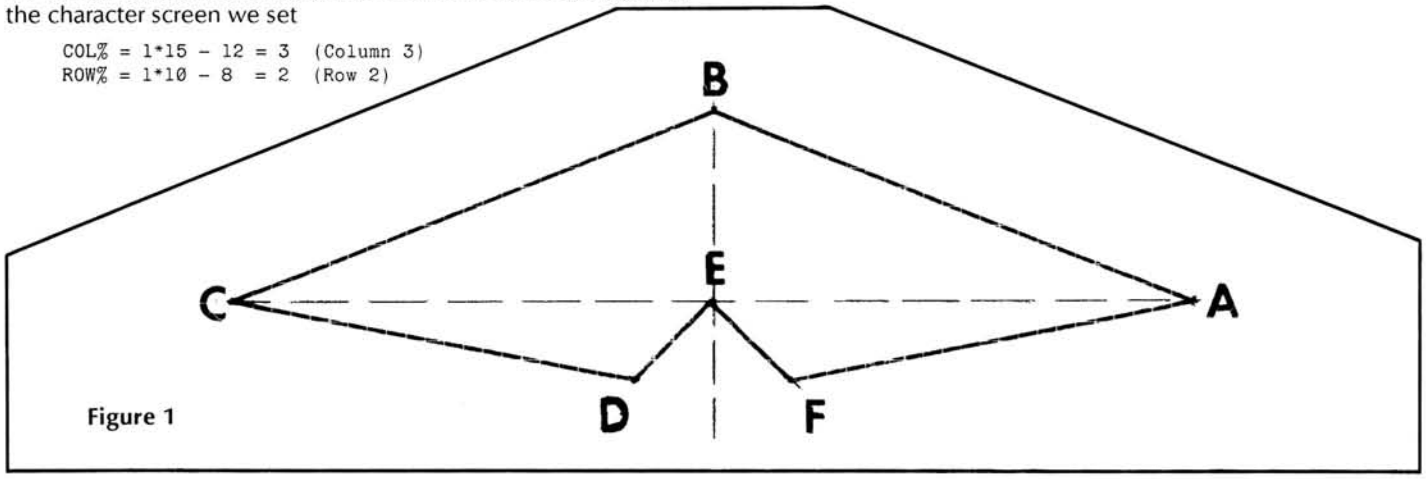


Figure 1

Program 2-2 plots Figure 1, then rotates the figure 12 times (NR = 12) through a cycle of 360 degrees (2*PI radians), returning the figure to its original position. The program also specifies the number of cycles (NC). If NC is set equal to 1.5, then the figure will be rotated through 1.5 revolutions.

Program 2-2 can be used to create some impressive graphics on the color monitor. If you have a color monitor, try the following program modifications. Change the size of the display area in line 120 to 400 pixels (DX = 400). Then change the aspect ratio from 0.4375 to 0.4844 in line 130. Add some color with the following lines

```
1020 CLR = CLR+2
1022 IF CLR = 6 THEN CLR = 2
1024 COLOR CLR
1300 COLOR 1
```

Delete line 1150 (CLS). Try running several different values for the number of rotations (NR) in line 420 such as 2, 4, 8, 16, 32, 60, 80. Test several different color schemes.

Program 2-2 has several drawbacks. When graphing a figure, it is much easier to define vertices in terms of x and y coordinates than in terms of angle of rotations and radii. Also, if you have a lot of vertices and do a lot of rotations, it takes the computer a while to calculate all those sines and cosines, and this slows the graphics down noticeably.

Rotation Of Shapes (Rectangular Coordinates)

An alternate method to rotate points, which is somewhat faster computationally, is derived from trigonometric identities. If we let the angle S and radius R locate a point on a graph, then

```
X = R*COS(S)
Y = R*SIN(S)
```

The formulas

```
XN = R*COS(S+AN)
YN = R*SIN(S+AN)
```

represent the new point (XN,YN), after rotation through angle AN. These formulas may be expanded and simplified to

```
XN = X*COS(AN) - Y*SIN(AN)
YN = X*SIN(AN) + Y*COS(AN)
```

In this form, we need to calculate the angle of rotation only once to draw the figure in its new orientation. If we let

```
A = SIN(AN)
B = COS(AN)
```

then we can calculate the new points directly with simplified formulas

```
XN = X*B - Y*A
YN = X*A + Y*B
```

where A and B are constant. These formulas have two advantages. The first advantage is that the new vertices can be calculated from the x and y coordinates of the old vertices, and it is easier to draw the figure on graph paper using rectangular coordinates. Second, multiplication by constants is much faster on the computer than calculation of trigonometric functions.

Program 2-2

```
10 ' PROGRAM 2-2
20 ' ROTATES A SHAPE, POLAR COORDINATES
30 '*****************************'
40 '*****   THOMAS J.VADEN   *****'
50 '*****   COPYRIGHT 1985   *****'
60 '*****************************'
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
310 PI=3.14159
350 R=61
360 SX=DX/R                        'SCALE X-VALUE
370 SY=DY/R                        'SCALE Y-VALUE
400 REM - INITIALIZE ROTATIONS
410 N1=0                           'START POINT
420 NR=20                          'ROTATIONS PER CYCLE
430 AR=2*PI/NR                     'ANGLE OF ROTATION
440 NC=1.5                         'NO. OF CYCLES
1000 FOR K% = 0 TO NR*NC
1010 A = N1+K%*AR
1030 X1 = CX + SX*COS(A)*50
1040 Y1 = CY - SY*SIN(A)*50
1050 X2 = CX + SX*COS(A+PI/2)*20
1060 Y2 = CY - SY*SIN(A+PI/2)*20
1070 X3 = CX + SX*COS(A+PI)*50
1080 Y3 = CY - SY*SIN(A+PI)*50
1090 X4 = CX + SX*COS(A+5*PI/4)*10
1100 Y4 = CY - SY*SIN(A+5*PI/4)*10
1110 X5 = CX
1120 Y5 = CY
1130 X6 = CX + SX*COS(A+7*PI/4)*10
1140 Y6 = CY - SY*SIN(A+7*PI/4)*10
1150 CLS
1200 PSET (X1,Y1)
1210 LINE -(X2,Y2)
1220 LINE -(X3,Y3)
1230 LINE -(X4,Y4)
1240 LINE -(X5,Y5)
1250 LINE -(X6,Y6)
1260 LINE -(X1,Y1)
1299 NEXT K%
```

Rotating An Airplane

In this section, we draw a figure of an airplane on graph paper, and locate the corners and the point of origin (see Figure 2).

	Coordinates	
Vertex	X	Y
Origin	0	0
A	0	25
B	-4	8
C	-30	-4
D	-3	-4
E	-3	-10
F	-8	-14
G	8	-14
H	3	-10
I	3	-4
J	30	-4
K	4	8

In Figure 2, the distance between the origin and point 1 is 25 units. The distance from the origin to any vertex is found using a geometric formula for calculating the distance between two points:

```
R*R = (X-Xo)*(X-Xo) + (Y-Yo)*(Y-Yo)
```

If the Origin (Xo,Yo) = (0,0), this reduces to

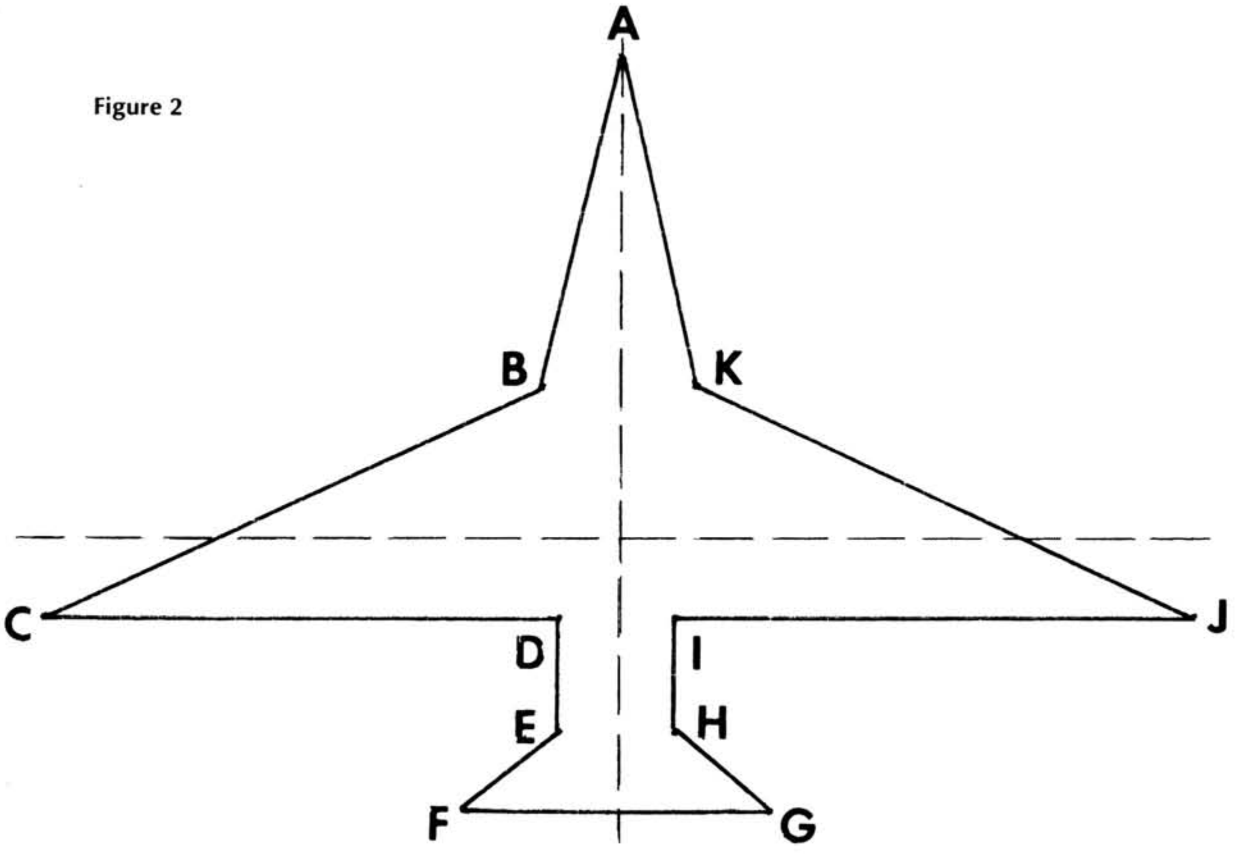
```
R*R = X*X+Y*Y
R = SQR(X*X+Y*Y)
```

For example, to find the distance from the origin to either wing tip, the radius

```
R = SQR(4*4+30*30) = SQR(916) = 30.27
```

Since the maximum distance from the Center of rotation (0,0) to any vertex is the distance to a wing tip, the maximum possible range is twice the radius (R) or 61 pixels.

Figure 2



Changing The Center Of Rotation

Program 2-3 includes an easy method to change the center of rotation. The center of rotation is read in as the point (XP,YP) in the data statements. We can easily rotate the airplane on its left wing by setting the center point (XP,YP) = (-30,-4) in the data statements. The program automatically finds new vertices relative to the center point using the equations

```
X1 = X - XP
Y1 = Y - YP
```

Several centers of rotation (XP,YP) are suggested:

(XP,YP)	Location of Center of Rotation
(0,0)	Center of airplane
(-30,4)	Left wing tip
(-50,4)	20 units off port
(30,-4)	Right wing tip
(40,-4)	10 units off starboard
(0,25)	Nose
(-3,-4)	Tail
(-13,4)	10 units to the rear

There are several words of caution. If the center of rotation is placed off the right wing tip, the airplane will fly backwards (counterclockwise rotation). To fly the airplane forwards, change the sign of the sine of the angle of rotation from A = SIN(AN) to A = -SIN(AN). Also, changing the center of rotation changes the maximum range of the function. This affects the scaling factor and size of the display area (which was originally 61 pixels). To maintain the correct display size on the screen, we need to find an easy way to calculate the required scaling factor. Program 2-4 includes a routine (lines 500-640) which automatically calculates the scaling factors from the coordinate data. The program calculates the distance from the center of rotation to each vertex, finds the maximum distance, determines the range, and converts this range to the correct scaling factors (SX,SY) so that the rotated graphics display will fit within the predefined screen display area (DX,DY).

Program 2-3

```
10 ' PROGRAM 2-3
20 ' ROTATES A SHAPE, RECTANGULAR COORDINATES
30 '*****************************'
40 '*****    THOMAS J.VADEN   *****'
50 '*****    COPYRIGHT 1985   *****'
60 '*****************************'
100 REM - SET DISPLAY BOUNDARIES
110 CLS
120 DX= 300                     'SIZE OF DISPLAY AREA
130 DY=.4375*DX                 'CORRECTS FOR DISTORTION
210 CX=320                      'CENTER COORDINATES
220 CY=112
300 REM - INITIALIZE GRAPH
310 PI=3.14159
320 DIM X(50),Y(50)
330 RX=61                       'MAXIMUM RANGE
360 SX=DX/RX                    'SCALE X-VALUE
370 SY=DY/RX                    'SCALE Y-VALUE
400 REM - INITIALIZE ROTATION
410 N1=0                        'START POINT
420 NR=12                       'ROTATIONS PER CYCLE
430 AR = 2*PI/NR                'ANGLE OF ROTATION
440 NC=1                        'NO. OF CYCLES
1000 REM - ROTATES AIRCRAFT
1010 FOR K% = 0 TO NR*NC
1020 AN = N1+K%*AR
1030 READ NP,XP,YP               'NO. OF POINTS, ORIGIN (Xo,Yo)
1040 A = SIN(AN)
1050 B = COS(AN)
1060 FOR I% = 1 TO NP
1070 READ X,Y
```

```
1080 X1 = X-XP
1090 Y1 = Y-YP
1100 X(I%) = CX+SX*(X1*B-Y1*A)
1110 Y(I%) = CY-SY*(X1*A+Y1*B)
1120 NEXT I%
1199 CLS
1200 PRESET (X(NP),Y(NP))
1210 FOR I%=1 TO NP
1220 LINE -(X(I%),Y(I%))
1230 NEXT I%
1290 RESTORE
1299 NEXT K%
1300 DATA 11,0,0
1310 DATA 0,25,-4,8,-30,-4,-3,-4
1320 DATA -3,-10,-8,-14,8,-14,3,-10
1330 DATA 3,-4,30,-4,4,8
```

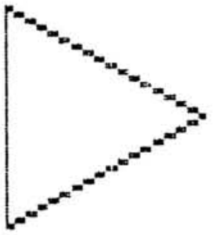
Creating Artistic Symmetrical Patterns

Program 2-4 creates symmetrical patterns by rotating arbitrary shapes in a two dimensional plane. Specific shapes are determined by the vertices and center of rotation read in data statements. A variety of displays are formed by changing the center of rotation (XP,YP), the number of rotations (NR), the start/end points (N1,N2), or the size of the display area. Other interesting experiments include multiplying the SIN(AN) or the COS(AN) by some factor, or changing the data set to form a new shape for creating a set of patterns.

Program 2-4 can be quickly modified to rotate other shapes by changing the data set. A data set is formed by reading in the number of vertices, the origin or center of rotation, and the coordinates of the vertices.

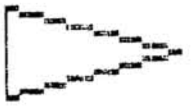
Equilateral Triangle:

Point	Coordinates X	Y	Data set:
Origin	0	0	DATA 3,0,0
A	2	0	DATA 2,0
B	-1	1.732	DATA -1, 1.732
C	-1	-1.732	DATA -1,-1.732



Isosceles Triangle:

Point	Coordinates X	Y	Data set:
Origin	0	0	DATA 3,0,0
A	3	0	DATA 3,0
B	-1	1	DATA -1, 1
C	-1	-1	DATA -1,-1



The location of the vertices and point of origin for a triangle requires a knowledge of trigonometry. For example, the center of a triangle is located 2/3 of the distance from any vertex to the midpoint of the opposite side.

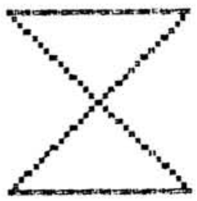
When drawing a closed figure, starting with vertex A, each successive point is normally located in the counterclockwise direction. If the order in which the vertices are located is mixed, a new shape emerges. For example, a square is changed into an hourglass by reversing two adjacent vertices:

Square:

Point	Coordinates X	Y
Origin	0	0
A	1	1
B	-1	1
C	-1	-1
D	1	-1

Hourglass:

Point	Coordinates X	Y
Origin	0	0
A	1	1
B	-1	1
C	1	-1
D	-1	-1



Suggested centers of rotation/origins:

Origin	Location of origin
(1,0)	Midpoint of left side
(-1,-1)	Lower left vertex
(3,3)	Point to upper right
(0,3)	Point to right of center
(-.5,.5)	Point inside the square

A square is stretched into a rectangle by multiplying either the x values or y values by a constant larger than one:

Rectangle:

Point	Coordinates X	Y
Origin	0	0
A	1	1.4
B	-1	1.4
C	-1	-1.4
D	1	-1.4

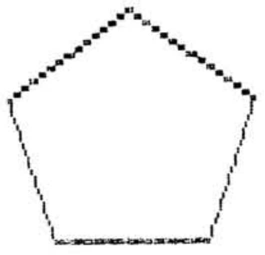
Rectangular Coordinates For Regular Polygons

The data sets for these figures are not unique. The set of vertices can be multiplied by a scalar constant and translated or rotated to a new orientation in the plane without distorting the shape of the figures.

Program 2-5 is designed to calculate rectangular (Cartesian) coordinate system data sets for any regular polygon. The start angle (AA) controls the orientation or location of the first vertex A, and the incremental angle (AN) locates the next vertex. Successive vertices are calculated recursively using the trigonometric identities developed for rotations. These vertices can then be interchanged to form other patterns. For example, we can find the coordinates of a pentagon and mix these coordinates to form a five point star or other weird shape:

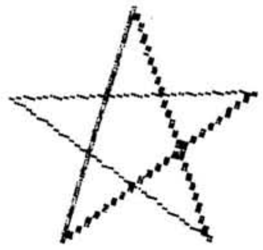
Pentagon:

Point	Coordinates X	Y
Origin	0	0
A	0	1
B	-.951	.309
C	-.588	-.809
D	.588	-.809
E	.951	.309



Five Point Star:

Point	Coordinates X	Y
Origin	0	0
A	0	1
B	-.588	-.809
C	.951	.309
D	-.951	.309
E	.588	-.809

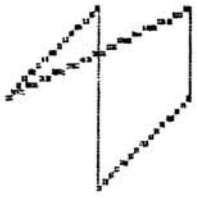


Irregular Figures

The last example is an arbitrary mixture of five data points:

Five Point Doily:

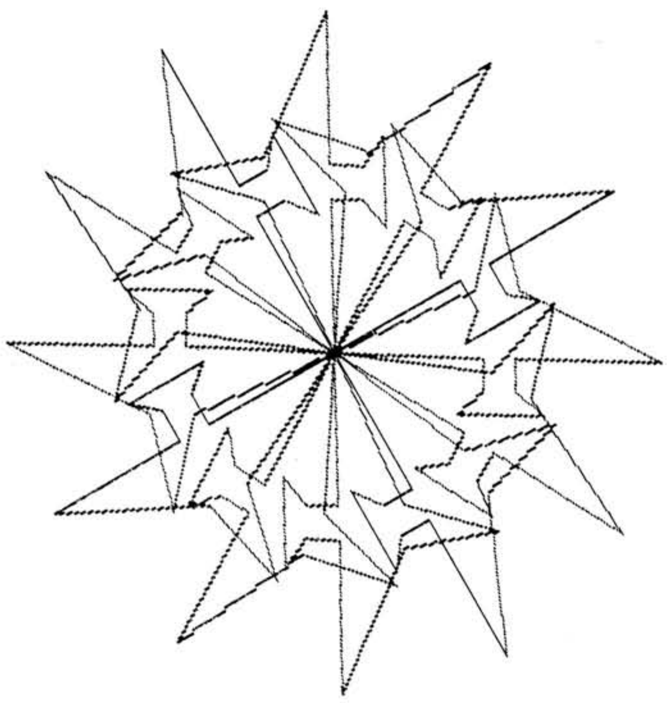
```
         Coordinates
Point     X        Y
------  -----    -----
Origin    0        0
  A       0        1
  B       1        0
  C      -1        0
  D       0       -1
  E       1        1
```



The reader is encouraged to get out his graph paper and try his own data sets, centers of rotations and colors. Program 2-4 has limitless possibilities for creation of unique, aesthetically pleasing visual displays.

Program 2-4

```
10 ' PROGRAM 2-4
20 ' CREATES ARTISTIC DISPLAYS
30 ' USING AIRPLANE SHAPE
40 '*****************************'
50 '*****   THOMAS J.VADEN   *****'
60 '*****   COPYRIGHT 1985   *****'
70 '*****************************'
100 REM - SET DISPLAY BOUNDARIES
110 CLS
120 DX= 450                     'SIZE OF DISPLAY AREA
130 DY=.4844*DX                 'CORRECTS FOR DISTORTION
210 CX=320                      'CENTER COORDINATES
220 CY=112
300 REM - INITIALIZE GRAPH
310 PI = 3.14159
320 NR=12                       'ROTATIONS PER CYCLE
330 N1=0                        'STARTING POINT FOR ROTATIONS
350 AR = 2*PI/NR                'INCREMENTAL ANGLE OF ROTATION
500 REM - CALCULATE MAX RANGE
510 READ NP,XP,YP               'NO. OF POINTS,ORIGIN (Xo,Yo)
520 DIM X(NP),Y(NP)
530 MAX = 0
540 FOR I% = 1 TO NP
550 READ X,Y
560 X1 = X-XP
570 Y1 = Y-YP
580 SS = X1*X1+Y1*Y1
590 IF SS > MAX THEN MAX = SS
600 NEXT I%
610 RX = 2*SQR(MAX)
620 SX = DX/RX
630 SY = DY/RX
640 RESTORE
1000 REM - DISPLAY PATTERNS
1010 FOR K% = 0 TO NR
1020 AN = N1+K%*AR
1030 READ NP,XP,YP
1040 A = SIN(AN)
1050 B = COS(AN)
1060 FOR I% = 1 TO NP
1070 READ X,Y
1080 X1 = X-XP
1090 Y1 = Y-YP
1100 X(I%) = CX+SX*(X1*B-Y1*A)
1110 Y(I%) = CY-SY*(X1*A+Y1*B)
1120 NEXT I%
1205 PRESET (X(NP),Y(NP))
1210 FOR I%=1 TO NP
1215 CLR = CLR+2
1216 IF CLR = 6 THEN CLR = 2
1217 COLOR CLR
1220 LINE -(X(I%),Y(I%))
1230 NEXT I%
1290 RESTORE
1299 NEXT K%
1300 COLOR 7
1500 DATA 11,-30,-4
1510 DATA 0,25,-4,8,-30,-4,-3,-4
1520 DATA -3,-10,-8,-14,8,-14,3,-10
1530 DATA 3,-4,30,-4,4,8
```



Program 2-5

```
10 REM PROGRAM 2-5
20 REM - CALCULATES RECTANGULAR COORDINATES
30 REM - FOR REGULAR POLYGONS WITH RADIUS 1
40 '*****************************'
50 '*****   THOMAS J.VADEN   *****'
60 '*****   COPYRIGHT 1985   *****'
70 '*****************************'
110 PI = 3.14159
120 READ NS,A$
125 IF NS = 0 THEN 999
129 AA = PI/2
130 IF NS MOD 2 = 0 THEN AA = PI/NS
140  X = COS(AA) :  Y = SIN(AA)
150 AN = 2*PI/NS     'INCREMENTAL ANGLE IN RADIANS
160  A = SIN(AN) :  B = COS(AN)
165 PRINT:PRINT
170 PRINT A$+":"
180 PRINT ,"Coordinates"
190 PRINT "POINT","X","Y"
200 FOR K% = 1 TO NS
210 PRINT CHR$(64+K%),
211 PRINT USING "##.###     ##.###";X,Y
215 XH = X
220 X = X*B-Y*A
230 Y = XH*A+Y*B
299 NEXT K%
300 GOTO 120
999 END
1000 REM - SIDES, SHAPE
1010 DATA 3,"TRIANGLE"
1020 DATA 4,"SQUARE"
1030 DATA 5,"PENTAGON"
1040 DATA 6,"HEXAGON"
1050 DATA 7,"HEPTAGON"
1099 DATA 0,"QUIT"
```

Continued on Page 51

Computer Data Security (Old Wine In New Bottles)

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

If you work with microcomputers in almost any part of a large (or maybe not-so-large) business, or have anything to do with the Federal Government, then you have security problems. Everyone does. It's the nature of organizations to try to guard their working information, whether it's to prevent tipping off the competition about a new sales campaign or tipping off the Russians about the Defense Plan.

This kind of security has an almost endless number of facets. Break-ins to on-line databases have become so common that they rate barely a mention in passing. Industrial espionage by computer is on the rise. Software theft and piracy is a raging controversy that has developed a life of its own, and will probably be debated by our grandchildren (I know my children are already discussing it; so much for that generation.) Theft of the systems themselves is an increasingly serious problem, as witness UPS's loss rates. And the list goes on.

What we're concerned about in this article, however, is the security of the data that you and I so laboriously type onto our disks, counting on that data to be available the next time we need it, and safe from unauthorized use. First, how can we protect our data so it's there the next time we insert disk 22 into Drive B:? Second, how can we be sure that the sneaky so-and-so down the hall at work hasn't "borrowed" data we thought we'd eliminated, and done something nasty (and unauthorized) with it?

How To Keep What You've Got

Making sure the data we have on our disks stays on our disks is largely a matter of careful handling and accident prevention. Careless treatment of the disk will certainly cost us in lost data, sooner or later. Accidental damage to the disk will cost us immediately, if not sooner. What are the risks and what are the safeguards? Let me share some of my experiences with you. While mainly oriented toward the office user, most of the following will apply to the home user, as well.

First, careless handling. Pick up one of your disks. Is it in it's envelope? It should be. That envelope is your first line of defense from dust, moisture, fingerprints, hairs, doughnut crumbs, spilled coffee and all the other foreign material that might come in contact with the disk surface and obliterate something. That something that gets obliterated need only be one bit (one-eighth of a byte) of data in order for your data to be unreadable. This might seem like an extreme case, and admittedly there are often ways to retrieve data lost that way, but sometimes your luck runs out, and all is lost.

So we always store our disks in their envelopes when they're not in use. What do you do when you have to switch disks, perhaps just for a few minutes while you copy a bit of information? Put the disk in the envelope. Don't lay it down, even for a moment, without its envelope. Disks spinning in their protective jacket tend to generate static electricity, and that statically charged disk acts just like any other object would. Remember rubbing the hard rubber comb on a piece of fur in high school physics? Same principle, same results. The disk will pick up any small objects nearby, such as dust, hair, crumbs, etc. Another excellent reason to always use the envelope.

While you have your disk envelope in hand, look at the back. Unless you use Brand X disks, there will be some form of legend on the back that tells you what to avoid. The only one that might not be obvious is the caution about magnets. Take a moment to think about all the magnets around your area. The computer you use generates a magnetic field, but the strength is low and the major offenders (transformers and motors) are usually buried deep inside the case. There are others lurking about, though. The worst offender is the telephone. Leave a disk by the phone, let the phone ring, and there's a good chance some damage has been done to the magnetic characters on the disk. You might get away with it for a while, but eventually you'll get caught. Watch out for your printer. It has some fairly powerful motors that can generate enough of a magnetic field to scramble data.

Once you've put your disks in the filing cabinet or safe, they're generally all right. In the case of fire, flood or other disaster, they're no more vulnerable than paper, and possibly less. Simply getting wet won't hurt a disk, so long as you let it dry gently. If all else fails, you can always cut open the damaged envelope, remove the disk in such a way that you don't touch the magnetic surface (the three-quarters of an inch closest to the outer edge) and put in into another similarly emptied, but undamaged envelope. You stand a very good chance of being able to read the disk, and your cost is minimal. I've had to do this numerous times when atmospheric humidity has caused the disk to become so tight in its envelope that the drive couldn't spin it.

Naturally, any sort of disk box is better for storage than nothing. Metal, wood or plastic boxes are better than the cardboard box the disks came in, but mostly to protect from water damage in case of a spill. In general, any sort of protective housing will suffice.

Need to transfer your data to another location? The three obvious ways, in order of decreasing security, are to carry it with you as you travel, mail it or send it via MODEM. We'll concentrate on the first two.

Plan A. If you plan to carry it, you have only two worries. First, don't lose it. That's up to you. Second, protect it as it goes through airport security. Many folks fear the x-ray machines now encountered at almost all airports. Not to worry. For the x-rays to harm a disk, they'd have to be so powerful that they would warm your fillings when you walk by the machine. X-rays won't hurt your disks because x-rays are primarily a radiation phenomenon, not a magnetic phenomenon.

The conveyor belt on which you place your briefcase is another matter. It has one of the best disk-killers imaginable, a large motor with a correspondingly large magnetic field. The conveyor belt motor will definitely harm the disk if the disk gets too close. It would be better to hand-carry your disks through the security check rather than run them through on the belt. Treat your disks as if they were high-speed (sensitive) camera film. Naturally, if an attractive enemy agent tries to seduce you into surrendering your sensitive information, none of this will help in the slightest.

Plan B is to mail the disks. Insured, preferably. Not that you'll be able to recover the value of the data on the disks; you probably won't, unless it was a commercial program with demonstrable (catalog price) value. But the post office seems to take better care of insured packages than others, so it's probably worth the little added cost. Naturally, you have backups, just in case the worst happens . . .

Use a strong mailer (there are many on the market that are excellent) or sandwich your disks between two pieces of corrugated cardboard. You might write "DO NOT BEND" on the envelope to discourage the post person from trying to roll it up to fit the receiving mail box. Don't worry about postal x-ray equipment; the same thing applies here as at the airport. The real risk is the same, too. Lots of mail rolls along on motorized conveyor belts, and if the disk gets too close, there could be problems. How often is this likely to ruin a disk? I don't know, as it's never happened to me. Here in Germany at the end of the pipeline, I send and receive lots of disks, and none have ever been damaged in the mail. Seems pretty safe to me.

How To Dispose Of Those Unwanted Secrets

As microcomputers proliferate throughout all levels of business and government, the problem of data security grows more important to the individual user. Time was, not too long ago, when data security was the job of that sour-faced person down the hall, who was in charge of system passwords and who made you log on and off with a string of characters that no one could remember without writing them down in their telephone list finder. The Zenith contract with the U.S. Air Force and Navy, by itself, has been responsible for over 50,000 new microcomputers in the Federal Government alone, not to mention countless thousands being used to maintain compatibility. All these new users represent some level of potential security risk, and anything you and I can do to reduce that risk will be of real benefit.

Microcomputers have both helped and hurt the security problems of businesses and governments. On one hand, the ability to remove a couple of floppy disks from the computer and lock them up at the end of the day has been a plus. Once secured in the safe or filing cabinet, the data disks are as secure as their paper counterparts ever were. If the computer has a hard disk drive with a removable disk, the same is largely true for them, too. Computers with fixed hard drives, from which removal of the disk is a warrantee-voiding screwdriver operation pose a more serious problem that we'll return to.

The key thing to remember is that MS-DOS wasn't designed as a secure operating system. Created by Seattle Computer several years ago, and enhanced by J. G. Letwin, of Wintek (supplier of Heath's original tape software) and HDOS fame, MS-DOS sprays data all over the disk, in locations you might never think to look. The problem is the disk write buffers, which can be left containing data at the end of a disk operation. This data can then be written into some other disk location totally unrelated to what you had in mind. That data is just waiting to be read by a disk-oriented editor such as EDisk, Zdump, DAE, or any one of a dozen other disk edit utilities that read a disk a sector at a - time.

Once you get into the habit of thinking that any disk you've had in the computer, while you were working with sensitive data, might be contaminated, you're on the path to security. Treat all your working disks as if they contained something you don't want anyone else to read. If you copy files from one of those disks to another disk, use the COPY command, not the DISKCOPY utility. The former will pick off just the file you want, with no extraneous material. The latter will faithfully move the stuff you don't want along with the stuff you do.

Once you've gathered all your sensitive information onto the desired disks, what do you do with the old, contaminated disks? If you simply use them over without purging the sensitive information, there's a good possibility that should someone borrow your disk, he or she could read the "old" information.

One way to purge the disk is to cut them up into small pieces and throw them out, just as you'd do with an unwanted credit card. It's extremely difficult to reconstruct a disk so badly damaged, but it's barely possible. Besides, there's a better way.

You might not want to try this method with the latest Top Secret NATO war plans, but for almost anything else it's one hundred percent secure. Take a brand new (virgin is the usual term) disk and format it using whatever format utility you generally use. Don't put a system on it, just prepare it as you would a data storage disk. When you've done that, you have a disk with nothing but blank characters in every memory location. Now, whenever you want to completely erase the contents of a data disk, use DISKCOPY to copy the contents of your blank disk onto your data disk. Hardly a revolutionary idea; many of us, myself included, have inadvertently copied a blank disk onto a data disk, usually at 3:00 AM after an all-night session at the keyboard. The result is the same no matter what. The old disk, the one that was once full of sensitive data, now contains nothing but blanks.

Why not just reformat the old disk? After all, the instructions in the manual and in the program itself warn you that the formatting process will destroy all data on the disk. Well, not exactly. Depending on the format utility you got with your operating system, the data may still be there, with just the disk directory reset, showing all the disk data space as available for use. The information itself may still be there, ready to be read by one of our disk editing utilities.

In Summary

This has been a brief look at the threats to the security of the information on your computer disks. Some threats, both physical and security, can be overcome by careful handling. Other threats, more in the nature of unexpected, unsuspected of unauthorized access to your information, require special precautions and techniques. We have focused on one simple method of insuring that when you delete sensitive information from a disk, it's truly obliterated, lost beyond recall by disk editors and similar techniques. ✱

EVERYTHING YOU NEED . . . $279^00

Now it's easy to program the Heath-Zenith HERO-1* Robot with an Apple* II. HERO* Macros for the S-C Software 6800 Cross Assembler program in Heath's Robot Interpreter Language with easily remembered mnemonics.

For example, the line: 1130 > MVWRIM GRIP, OPEN, 60, FAST instructs the HERO* to open his gripper 60 units at fast speed. Motor position is expressed in base-10.

The HERO* Macros come with 30 pages of documentation.

Transfer to HERO* with ROBI . . . an affordable interface for the robotics experimenter . . . is simple.

- ROBI is a complete package. No additional hardware required for Apple* or HERO*.
- ROBI installs quickly in an Apple* II, II+, or IIe. Once installed, no hardware changes are needed. Within minutes, you will be programming HERO*.
- With ROBI and the Cross Assembler, the programmer uses Apple*'s memory to write the program, and HERO*'s memory to run the program.
- Not "copy protected," archival copies may be made as needed.
- ROBI offers expansion potential.

VISA and MasterCard accepted.

BERSEARCH
Information Services
26160 Edelweiss Circle
Evergreen, Colorado 80439

The Cross Assembler with HERO* Macros sells for $100.00; the ROBI Interface sells for $199.00. Both as a package — $279.00. To order, or for more information, call (303) 670-6137.

APPLE* is a trademark of Apple Computer HERO* is a trademark of Heath Electronics

ZCHESS

For Z100 (not PC)
REQUIRES 5¼ FLOPPY
192Kb OF MEMORY
DOS vers 2.1 or above

An innovative chess challenge! Infinite levels of play! All moves are checked for legality. Setup, Save, Restore Positions, Change Sides, Keep Time, Tournament Mode and *more features!*

$24.95 (Ohio residents add sales tax)
COMETSOFT TECHNOLOGY
634 Tealwood Drive, Dayton, Ohio 45430
[513] 426-7486

Continued from Page 48

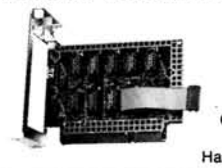
Conclusion

This article has covered a large number of topics associated with graphics. The specification of screen display location and size is equivalent to selecting a VIEWPORT and setting a range is the same as defining a WINDOW. SCALING factors were used to convert functions/graphs to display areas. To prevent distortion, the screen ASPECT RATIO was introduced. TRANSLATIONS (locations) and ROTATIONS were discussed, and a foundation was established from which other two-dimensional graphical concepts can be explored.

Computer graphics is not an easy subject to learn, but can be very rewarding in the sense that it stirs a certain creative spirit that so often lies dormant. Hopefully, this article will stimulate the reader to further explore this exciting field. I am looking forward to an increase in the number of articles in the area of computer graphics. ✱

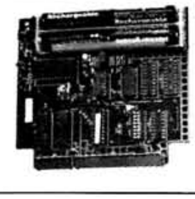
HEATH/ZENITH 88, 89, 90 PERIPHERALS

16K RAM EXPANSION CARD



Only $65.00
Shipping &
Handling $5.00

REAL TIME CLOCK



Price $130.00 with Batteries
Shipping &
Handling $5.00
$114.00 w/o Batteries

2 PORT SERIAL/3 PORT PARALLEL I/O CARD

Price $199.00
2nd Oper. System Driver $25.00
Ship. &
Hdlg $10

PRICES ARE LESS SHIPPING & TAX IF RES. OF CALIFORNIA.

MAIL ORDER: 12011 ACLARE ST. CERRITOS, CA 90701
(213) 924-6741

TECHNICAL INFO / HELP:
8575 KNOTT AVENUE, SUITE D
BUENA PARK, CA 90620
(714) 952-3930

TERMS & SPECIFICATIONS SUBJECT TO CHANGE WITHOUT NOTICE
VISA & MASTER CARD GLADLY ACCEPTED

Are you reading a borrowed copy of REMark? Subscribe now!

*** 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 $632.00, 31 MEG $727.00 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, 150 ns .... $2.94 each
256K Dynamic RAM, 120 ns .... $3.40 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 $448.00. 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 $128.00

*** 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 M501	5.25"	48 TPI DS/DD	320K/360K	$105.00
MITSUBISHI M504	5.25"	96 TPI DS/DD	360K/1.2 MEG	$177.00
MITSUBISHI M4853	5.25"	96 TPI DS/DD	640K	$109.00

*** LAPINE HARD DISK DRIVES *** 2 YEAR WARRANTY CALL FOR PRICES

*** SEAGATE HARD DISK DRIVES ***

ST-225 20 MEG Winchester Hard Disk $355.00
With Western Digital Controller & Cables $448.00
ST-238 30 MEG, Requires RLL type controller $405.00
With Adaptec RLL Controller & Cables $499.00
ST-4026 20 MEG High Speed for Z-200 .... $508.00
ST-4038 30 MEG High Speed for Z-200 .... $619.00
ST-4051 40 MEG High Speed for Z-200 .... $739.00
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.

Binary To BCD Conversion

Friedrich W. Bruegmann
1212 Regent Street
Niles, MI 49120
Copyright 1986. All Rights Reserved.

When writing assembly language programs, it is often necessary to output a binary number in decimal form. While this can be done quite easily when the binary number is only one or two bytes in size, it becomes far more interesting when dealing with larger binary numbers. This article will describe a method that can be used to output an unsigned binary number, of practically any size, in decimal form by first converting it into a binary coded decimal (BCD). Once a number is in BCD form, it becomes very easy to output it in the desired decimal format.

The most common method of outputting a binary number in decimal form is to divide the number (the dividend) by the largest power of ten (the divisor) that will yield an integer (the quotient) between one and nine. This quotient is the most significant digit of the desired decimal number and is output to the CRT. The remainder from the above division is then divided by the next lower power of ten, giving the next digit of the desired decimal number, which is then output to the CRT. This process is repeated until the divisor is one (ten to the zero power). Figure 1 demonstrates this process and Listing 1 is the code that implements this procedure.

```
65021/10000 = 6 with a remainder of 5021.  Output the 6.
 5021/1000  = 5 with a remainder of  021.  Output the 5.
  021/100   = 0 with a remainder of   21.  Output the 0.
   21/10    = 2 with a remainder of    1.  Output the 2.
    1/1     = 1 with a remainder of    0.  Output the 1.
```

Figure 1

Those of us using a computer with an 8088 CPU are fortunate, since the 8088 instruction set includes a division instruction. Using an 8080 CPU, it is necessary to do the division by means of a loop using repeated subtractions and/or shifts. However, even with the added power of the 8088, there are still some serious limitations. The 8088 DIV instruction supports a maximum dividend size of four bytes (split between the DX and AX registers). This gives us a dividend that can be as large as 4,294,967,295! The problem here comes from the fact that the maximum size of the divisor is only two bytes. This means that the largest power of ten that we can divide by is only 10,000, not the 1,000,000,000 that we need to divide by if our dividend is between 1,000,000,000 and 4,294,967,295. It should be possible to divide the dividend by the largest power of ten allowed (10,000) and handle the quotient separately through repeated divisions. Once finished with the quotient, we can go on and handle the remainder as we did in the example above.

Listing 1

```
PAGE ,132
;
TITLE - BINTODEC - PROGRAM TO DISPLAY A 2 BYTE BINARY
                   NUMBER IN DECIMAL FORM
;
COMMENT *
WRITTEN BY :        FRED BRUEGMANN
                    1212 REGENT ST.
                    NILES, MI 49120

                    MARCH 1, 1986

TO ASSEMBLE :
                    MASM BINTODEC BINTODEC;
                    LINK BINTODEC;
                    EXE2BIN BINTODEC BINTODEC.COM
                    ERASE BINTODEC.OBJ
                    ERASE BINTODEC.EXE
*
;
;  ****  SYSTEM FUNCTIONS  ****
;
DOSF_CONOUT     EQU     2               ;CONSOLE INPUT
;
;  ****  SYSTEM INTERRUPTS  ****
;
DOSI_TERM       EQU     20H             ;PROGRAM TERMINATE
DOSI_FUNC       EQU     21H             ;PERFORM A FUNCTION
;
CODES   SEGMENT
        ASSUME  CS:CODES,DS:CODES,ES:CODES,SS:CODES
;
        ORG     100H        ;START FOR ALL .COM PROGRAMS
;
START:
        JMP     BEGIN
;
;****************
;**  DATA AREA **
;****************
;
BINARY  DW      0FDFDH
;
;*******************
;**  PROGRAM AREA **
;*******************
;
BEGIN:
        LEA     SI,BINARY       ;POINT TO BINARY NUMBER
        LODSW                   ;GET LOW WORD
        XOR     DX,DX           ;SET UP DX FOR WORD DIVISION
        MOV     BX,10000        ;DIVIDE BY 10,000 (DECIMAL)
        DIV     BX              ;  (WORD DIVISION)
        PUSH    DX              ;SAVE REMAINDER
        MOV     DL,AL           ;PUT QUOTIENT IN DL
        CALL    DECOUT          ;OUTPUT THE QUOTIENT
        POP     AX              ;GET REMAINDER FROM STACK
        XOR     DX,DX           ;SET UP DX FOR WORD DIVISION
        MOV     BX,1000         ;DIVIDE BY 1000 (DECIMAL)
        DIV     BX              ;  (WORD DIVISION)
        PUSH    DX              ;SAVE REMAINDER
        MOV     DL,AL           ;PUT THE QUOTIENT IN DL
        CALL    DECOUT          ;OUTPUT THE QUOTIENT
        POP     AX              ;GET REMAINDER FROM STACK
        MOV     BL,100          ;DIVIDE BY 100 (DECIMAL)
        DIV     BL              ;  (BYTE DIVISION)
        PUSH    AX              ;SAVE REMAINDER
        MOV     DL,AL           ;MOVE QUOTIENT INTO DL
        CALL    DECOUT          ;OUTPUT THE QUOTIENT
        POP     AX              ;GET REMAINDER
        MOV     AL,AH           ;PUT REMAINDER IN AL
        CBW                     ;SET UP AH FOR DIVISION
        MOV     BL,10           ;DIVIDE BY 10 (DECIMAL)
        DIV     BL              ;  (BYTE DIVISION)
        PUSH    AX              ;SAVE REMAINDER
        MOV     DL,AL           ;PUT THE QUOTIENT IN DL
        CALL    DECOUT          ;OUTPUT THE QUOTIENT
        POP     AX              ;GET REMAINDER FROM STACK
        MOV     DL,AH           ;PUT THE REMAINDER IN DL
        CALL    DECOUT          ;OUTPUT THE REMAINDER
EXIT:
        INT     DOSI_TERM
;
DECOUT:                         ;OUTPUT A DECIMAL DIGIT
        ADD     DL,30H          ;CONVERT TO A ASCII DIGIT
        MOV     AH,DOSF_CONOUT
        INT     DOSI_FUNC
        RET
;
CODES   ENDS
        END     START
```

Unfortunately, this does not work well either. While the DIV instruction supports a maximum of four bytes for the dividend, it only allows the quotient and remainder to be two bytes each. Using the maximum dividend of 4,294,967,295 and dividing by 10,000 we get a quotient of 429,496 with a remainder of 7,295. However, if we try to do this division our program will stop with an Interrupt 0 (divide overflow), since 429,496 is larger than 65,535 (the quotient is larger than two bytes). Another problem is the fact that the maximum dividend size is four bytes. It would be difficult to use the DIV instruction to output a six byte binary number as a decimal. While none of these limitations are impossible to overcome, they were enough to make me look for an alternative algorithm.

The solution to this problem came in the form of Motorola Application Note AN-506. This "AP-NOTE" describes a method of using a Motorola chip to do a hardware binary to BCD (binary coded decimal) conversion. BCD is a method used to represent a number in a computer where each digit of the base ten number is represented using four bits. In normal unsigned binary representation, one byte can be used to represent the numbers between 0 and 255, inclusive. Using BCD representation, one byte (two groups of four bits) can be used to represent the numbers between 0 and 99 inclusive. BCD is essentially the same as hexadecimal number representation except that A – F are illegal digits. Once the number is in BCD representation it is easy to output. Below are two examples of the different ways numbers can be represented.

The decimal number 117 can be represented by the binary number 0111 0101, the hexadecimal number 75, or the BCD representation 0001 0001 0111. Note that the BCD representation can also be represented as 117 in hexadecimal.

The decimal number 75 can be represented by the binary number 0100 1011, the hexadecimal number 4B, or the BCD number 0111 0101. Again, note that the BCD representation can also be represented as 75 in hexadecimal.

The method described by Motorola is simple, efficient, easy to implement in software, and has the advantage of being able to handle very large numbers. The process uses two "register" groups, one group of BCD "registers" followed by one group of binary "registers" (read "registers" as bytes of RAM). The BCD registers should all be set to zero, and the binary registers should contain the binary number that is to be converted to BCD. See Figure 2 for an explanation of the terms used in the following paragraphs.

```
                        BINARY REGISTER
==============================================================
byte #              1                              0

bit # 15 14 13 12    11 10  9  8    7  6  5  4    3  2  1  0
      --------------------------------------------------------
117 =  0  0  0  0     0  0  0  0    0  1  1  1    0  1  0  1
```

Bit # 0 is the least significant bit of the least significant byte and bit # 7 is the most significant bit of the least significant byte. Bit # 8 is the least significant bit of the most significant byte and bit # 15 is the most significant bit of the most significant byte.

```
                        BCD REGISTERS
==============================================================
byte #              1                              0

digit #       3              2              1              0

bit #   15 14 13 12    11 10  9  8    7  6  5  4    3  2  1  0
        ------------------------------------------------------
117 =    0  0  0  0     0  0  0  1    0  0  0  1    0  1  1  1
```

Bit # 0 is the least significant bit of the least significant BCD digit and bit # 3 is the most significant bit of the least significant BCD digit (digit # 0). Bit # 12 is the least significant bit of the most significant BCD digit and bit # 15 is the most significant bit of the most significant BCD digit (digit # 4). Byte # 0 is made up of BCD digits 0 (bits 0–3) and 1 (bits 4–7). Byte # 1 is made up of BCD digits 2 (bits 8–11) and 3 (bits 12–15).

Figure 2

The conversion starts by shifting every bit in the binary registers one position to the left. The vacancy created at the least significant bit in the least significant binary register is filled with a zero bit (this occurs automatically by clearing the carry flag and using an RCL instruction). The carry out of the most significant bit of each binary register is moved into the least significant bit of the next binary register (this also occurs automatically using the RCL instruction).

Next, each bit of the BCD registers is shifted one position to the left. The vacancy created at the least significant bit in the least significant BCD register is filled with the carry out of the most significant bit of the most significant binary register. As with the binary registers, the carry out of the most significant bit of each BCD register is moved into the least significant bit of the next BCD register. The carry out of the most significant bit of the most significant BCD register is ignored.

The process up to this point has, in effect, treated the BCD and binary registers as a single entity and shifted each bit one position to the left. Now we must examine the BCD registers and make an adjustment where necessary. Each BCD register (byte) holds two BCD digits. We must look at each BCD digit in every BCD register and if the digit is greater than or equal to five, we add three to the BCD digit. This shift and adjust process occurs once for each bit in the binary registers (16 times if the binary registers are a total of 16 bits wide) with one exception; there is no adjustment after the last shift.

Don't worry if you are feeling a little confused right now; the following example (see Figure 3) should help to clear things up. Let's use the binary number 0111 0101 (117 decimal) from the previous example and convert it to BCD. Notice that our binary number is only one byte long, but we need to allow three BCD digits, or two bytes (actually we need only one and one-half bytes, but it is simpler to code using two bytes), for the result. Since the binary number is one byte long, we need to do a total of eight shifts (8 bits in a byte).

```
BCD       BCD REGISTERS          BINARY
Digit  #3   #2   #1   #0        REGISTER
       ===============================
       0000 0000 0000 0000    0111 0101     Starting point
       0000 0000 0000 0000    1110 1010     Shift 1 (no adjust)
       0000 0000 0000 0001    1101 0100     Shift 2 (no adjust)
       0000 0000 0000 0011    1010 1000     Shift 3 (no adjust)
       0000 0000 0000 0111    0101 0000     Shift 4
                        + 11                        Adjust
       -------------------
       0000 0000 0000 1010
       0000 0000 0001 0100    1010 0000     Shift 5 (no adjust)
       0000 0000 0010 1001    0100 0000     Shift 6
                        + 11                        Adjust
       -------------------
       0000 0000 0010 1100
       0000 0000 0101 1000    1000 0000     Shift 7
                   + 11 + 11                        Adjust
       -------------------
       0000 0000 1000 1011
       0000 0001 0001 0111    0000 0000     Shift 8
                              (never adjust the last shift)

       0000 0001 0001 0111 (BCD) = 0117 base 10
```

Figure 3

Let's work through the example above. At the starting point we have the BCD registers (two bytes that hold BCD digits 0–3) and the binary register of the appropriate size. The binary register has been loaded with the value that is to be converted to BCD and the BCD registers have all been initialized to zero. Note that the size of each register group could be much larger and the only difference would be the number of times we have to do the shift and adjust process.

At Shift 1, we have shifted each bit of the binary register to the left one position and brought in a zero bit at the right most (least significant) bit. There is a zero bit carry out of the left most (most significant) bit of the binary register that is moved into the right most (least significant) bit of the BCD digit 0 as the BCD registers are shifted to the left one position. Since none of the BCD digits are greater than or equal to five after the shift, there is no need for any adjustment.

At Shift 2, we again shifted all the binary register bits one position to the left and brought in a zero bit at the right most (least significant) bit of the binary register. This time there is a one bit carried out of the left most (most significant) bit of the binary register, which is moved into the left most (least significant) bit of the BCD digit 0 as the BCD registers are shifted to the left one position. Again, there is no need for adjustment.

Shift 3 is the same as Shift 2.

At Shift 4, as in Shift 2 and Shift 3, we have shifted each binary register bit one position to the left, brought in a zero bit at the right most (least significant) bit, and carried a one bit into the right most (least significant) bit of BCD digit 0 as the BCD registers are shifted one bit to the left. This time we need to make an adjustment. The right most four bits of the BCD registers (digit 0) now have the value 0111 (7 decimal) and since this is greater than or equal to 0101 (5 decimal), we need to add 0011 (3 decimal) to this BCD digit. You may have noticed that the result is greater than nine; don't worry about this, since it will correct itself by the time the conversion process is complete. All the other BCD digits are zero, so no further adjustments are needed.

At Shift 5, we again shifted all the binary register bits one position to the left and brought in a zero bit at the right most (least significant) bit. Again, there is a one bit carry out of the left most (most significant) bit of the binary register, which is moved into the right most (least significant) bit of BCD digit 0 as the BCD registers are shifted one position to the left. Once again, there is no need for any adjustment.

At Shift 6, all bits of the binary register are shifted one position to the left, a zero bit is brought into the right most (least significant) bit of the binary register, and a zero bit is carried out into the right most (least significant) bit of BCD digit 0 as the BCD registers are shifted left one position. The right most BCD digit (digit 0) is greater than or equal to five, so we adjust by adding three. The BCD digit second from the right (digit 1) has the value two, so no adjustment is needed. No adjustment is needed on the remaining BCD digits, since their value is still zero.

At Shift 7, all binary register bits are shifted one position left, a zero is brought into the right most (least significant) bit of the binary register, and a zero bit is carried into the right most (least significant) bit of BCD digit 0 as the BCD registers are shifted one position left. After this shift, we have two BCD digits (digits 0 & 1) that need adjustment by adding three.

Shift 8 is the last shift. The shift occurs as before, except this time NO adjustment is made. Looking at the individual BCD digits, starting at digit 3, we have 0, 1, 1, and 7. This is the decimal number 117 that we wanted to output, in BCD form.

To output the BCD number is simply a matter of taking each of the BCD digits individually, adding 30H to the digit to convert it to an ASCII digit, and output it to the CRT, starting at the left most BCD digit and working your way to the right most BCD digit (digit 0). As I stated before, this routine works with numbers much larger than our example. There is no difference between the code that converts an eight bit binary number and the code that is needed to convert a 256 bit (32 byte) binary number. The only considerations are to be sure you have reserved enough space for the converted BCD number and to be sure to change the loop counter for the number of times to shift and adjust. Listing 2 shows the code for this procedure.

There are a few things I would like to point out about the code in Listing 2. In the Data Area, BINBYTES is the number of bytes for the binary registers (this would be set to 1 for the example in Figure 3). The value of BINBYTES must be at least as large as the binary number you want to convert to BCD. It may be larger; the only side effect is that more iterations of the shift and adjust loop will be necessary. The maximum value allowed for BINBYTES is 8191 (decimal), since 8191 * 8 (bits/byte) is 65528 bits, which is the largest multiple of eight that can be held in the 16-bit loop counter.

BCDBYTES is the number of bytes for the BCD registers. The value of BCDBYTES must be at least large enough to hold the converted BCD number (remember that each BCD byte holds two BCD digits). As with BINBYTES, this value may be larger than what is really needed to hold the converted number, but may not be any larger than the number of bytes reserved by the label BCD.

The label, BINARY, must reserve at least as many bytes as indicated by BINBYTES. Again, more bytes may be reserved, the only side effect being that it will make your program a few bytes larger. One important point here, if the binary number you load in BINARY does not have as many bits as you have reserved, you must be sure to set all unused leading bits to zero. It is your responsibility to make sure you have reserved enough space for the BCD number at the label BCD. If you have not, the BCD number will overwrite part of your program, with unpredictable results. Extra space may be reserved here also.

If you let MASM load the binary register, as I have done in Listing 2, you will notice that it stores the least significant byte of the binary number first (at memory location BINARY + offset 0) and the most significant byte last. When using this routine as a subroutine it is very important to follow this convention. The BCD registers will also be ordered this way when the conversion is completed. That is why we start at the end of the BCD registers and work toward the start when we output the BCD number.

One last thing, it is possible to convert a BCD number into a binary number using almost the same code. The only difference is that you must shift right one bit starting from the most significant bit in the BCD registers and to adjust after the shift, you must subtract three from each BCD digit that is greater than or equal to eight.

I hope some of you find this routine to be as interesting and helpful as I did. If you do not want to enter the code yourself, send me your name and address, along with $5.00, and I'll send you a disk with Listings 1 and 2, plus the BCD to binary conversion routine.

Listing 2

```
PAGE ,132
;
TITLE - BINTOBCD - CONVERT UNSIGNED INTEGER INTO BCD
                   AND OUTPUT IN DECIMAL FORM
;
COMMENT *

WRITTEN BY:    FRED BRUEGMANN
               1212 REGENT ST.
               NILES, MI 49120
               MARCH 1, 1986

TO ASSEMBLE :
               MASM BINTOBCD BINTOBCD;
               LINK BINTOBCD;
               EXE2BIN BINTOBCD BINTOBCD.COM
               ERASE BINTOBCD.EXE
               ERASE BINTOBCD.OBJ
*
;****************************
;****  SYSTEM FUNCTIONS  ****
;****************************
;
DOSF_CONOUT      EQU      2              ;CONSOLE OUTPUT
;
;*****************************
;****  SYSTEM INTERRUPTS  ****
;*****************************
;
DOSI_TERM        EQU     20H         ;PROGRAM TERMINATE
DOSI_FUNC        EQU     21H         ;PERFORM A FUNCTION
;
;
CODES   SEGMENT
        ASSUME  CS:CODES,DS:CODES,ES:CODES,SS:CODES
;
        ORG     100H                 ;START OF COM FILE
;
START:
        JMP     BEGIN
;
;********************
;****  DATA AREA ****
;********************
;
BINBYTES         DW      3              ;NUMBER OF BINARY BYTES
BCDBYTES         DW      3              ;NUMBER OF BCD BYTES
BINARY           DQ      0A78E8H        ;BINARY INTEGER TO CONVERT
BCD              DB      10H DUP (?)  ;AREA FOR CONVERTED
                                        (BCD) VALUE
;
;****************************
;****  START OF PROGRAM  ****
;****************************
;
BEGIN:
        MOV     DI,OFFSET BCD         ;ZERO BCD BYTES
        MOV     CX,BCDBYTES
        MOV     AL,00
        REP     STOSB
        MOV     DX,BINBYTES           ;LOAD DX WITH NUMBER
                                       OF BINARY BYTES
        MOV     CL,3                  ;CONVERT DX TO NUMBER
                                       OF BINARY BITS
        SHL     DX,CL                 ;BY SHIFTING LEFT 3 BITS
;
;**********************************
;****  SHIFT BINARY REGISTERS  ****
;**********************************
;
SHFBIN:
        MOV     CX,BINBYTES         ;LOAD CX WITH NUMBER OF
                                     BINARY BYTES
        CLD                         ;SET AUTO-INCREMENT
        CLC                         ;CLEAR CARRY FLAG
        MOV     SI,OFFSET BINARY    ;POINTER TO BINARY INTEGER
        MOV     DI,SI
SHFBITS:
        LODSB                       ;GET NEXT BYTE OF
                                     BINARY INTEGER
        RCL     AL,1                ;SHIFT
        STOSB                       ;STORE SHIFTED BYTE
        LOOP    SHFBITS
;
;*******************************
;****  SHIFT BCD REGISTERS  ****
;*******************************
;
        MOV     SI,OFFSET BCD    ;POINTER TO BCD INTEGER
        MOV     DI,SI
        MOV     CX,BCDBYTES      ;LOAD CX WITH NUMBER OF
                                  BCD BYTES
SHFBCD:
        LODSB                    ;GET NEXT BYTE OF BCD
                                  INTEGER
        RCL     AL,1             ;SHIFT
        STOSB                    ;STORE SHIFTED BYTE
        LOOP    SHFBCD
;
;********************************
;****  ADJUST BCD REGISTERS  ****
;********************************
;
        CMP     DX,1             ;LAST SHIFT DONE ?
        JZ      DONE             ;YES, DON'T DO ADJUSTMENT
        MOV     SI,OFFSET BCD   ;POINTER TO BCD REGISTERS
        MOV     DI,SI
        MOV     CX,BCDBYTES     ;LOAD CX WITH NUMBER OF
                                 BCD BYTES
ADJBCD:
        LODSB                   ;GET NEXT BYTE OF BCD
                                 INTEGER
        MOV     AH,AL           ;DUPLICATE IT IN AH
        AND     AL,0FH          ;MASK OFF HIGH NYBBLE
        AND     AH,0F0H         ;MASK OFF LOW NYBBLE
        CMP     AL,5            ;IS LOW NYBBLE < 5 ?
        JB      NOLOADJ         ;YES, DON'T ADJUST IT
        ADD     AL,3            ;NO, ADJUST IT
NOLOADJ:
        CMP     AH,50H          ;IS HIGH NYBBLE < 5 ?
        JB      NOHIADJ         ;YES, DON'T ADJUST IT
        ADD     AH,30H          ;NO, ADJUST IT
NOHIADJ:
        ADD     AL,AH           ;RECOMBINE LOW AND HIGH NYBBLE
        STOSB                   ;STORE ADJUSTED BCD BYTE
        LOOP    ADJBCD
;
        DEC     DX
        JNZ     SHFBIN
;
;******************************
;****  DISPLAY BCD NUMBER  ****
;******************************
;
DONE:
        MOV     SI,OFFSET BCD    ;POINTER TO BCD INTEGER
        ADD     SI,BCDBYTES
        DEC     SI               ;POINT TO END OF BCD INTEGER
        MOV     CX,BCDBYTES
        STD                      ;SET AUTO-DECREMENT
OUTLOOP:
        LODSB                    ;GET NEXT BCD BYTE
        MOV     AH,AL            ;DUPLICATE IT IN AH
        AND     AL,0FH           ;MASK OFF HIGH NYBBLE
        PUSH    CX               ;SAVE COUNTER
        MOV     CL,4             ;SHIFT HIGH NYBBLE INTO
                                  LOW 4 BITS
        SHR     AH,CL
        POP     CX               ;RESTORE COUNTER
        ADD     AX,3030H         ;CONVERT TO ASCII
        MOV     DL,AH            ;MOVE TO DX FOR CHARACTER
                                  OUTPUT
        MOV     DH,AL
        MOV     AH,DOSF_CONOUT
        INT     DOSI_FUNC        ;OUTPUT DECIMAL INTEGER
        MOV     DL,DH
        INT     DOSI_FUNC        ;OUTPUT DECIMAL INTEGER
        LOOP    OUTLOOP
;
        INT     DOSI_TERM
;
CODES   ENDS
        END     START
```

About The Author

Friedrich W. Bruegmann *is a Mechanical Designer for the Electronics Group at Electro-Voice, Inc., where he designs mechanical components for audio electronic equipment. He is a senior in Computer Science at I.U.S.B., and enjoys anything dealing with computers.*

*

HDS
PRESENTS
THE Z100 VIDEO ENHANCEMENT

Display 16 colors on a standard monitor
Simple software interface
ZHS option

DESCRIPTION

The **V1** video enhancement board from **HDS** will enable your **Z100** (not PC) to display 16 colors. This is done by adding an extra bank of video ram. In addition, a palette is provided which maps the 16 colors to any combination of 16. This can be used to create simple animation effects. A disk (MSDOS 2) is provided which contains numerous demonstration programs and subroutines for inclusion in your own programs.

INSTALLATION

The **V1** board sits above the video board and plugs into the main connector in parallel. In addition a web is used to obtain 26 signals which are not available through the main connector. The web contains modules and sockets into which certain video board chips are placed.
Note: **NO** soldering or trace cutting is involved.

REQUIREMENTS

A Z100 computer with full video ram (with 64K chips) is needed. To display all 16 colors, an IBM compatible color monitor is required.

OPTIONS

To use a composite monochrome monitor, you will need the **V1M** option. An equivalent to HUG's ZHS circuit is available as option **V1Z.** These packages contain chips to be plugged into the **V1** board.

PRICES

V1: $139.00   V1M: $29.00   V1Z:$29.00

Texas residents add sales tax. MC/VISA accepted. Phone orders taken Mon-Fri, 8AM to 5PM Central time. 90 day warranty. Call or write for more information.

HUGHES DEVELOPMENT SYSTEMS
10101 S.W. FREEWAY, SUITE 400
HOUSTON TEXAS 77074
(713) 772-2840

VAX & PC Users

ZSTEMpc™-VT220 Emulator for the PC Series
High performance COLOR VT220. Double high/wide, 132 mode, smooth scrolling, downloadable fonts, user defined keys, 8-bit and full national/multinational modes. XMODEM and KERMIT, softkey/MACROS, DOS access. **$150.00**

ZSTEMpc™-VT100 Fastest, most complete emulation $99.00

ZSTEMpc™-4014 Emulator for the PC Series
Use with ZSTEMpc-VT100, VT220, or stand-alone. Interactive zoom and pan. Save/recall images to/from disk. Keypad, mouse, printer, and plotter support. **$ 99.00**

ZSTEM™-VT100 Emulator with XMODEM FOR THE Z-100 $148.95

DECkHAND™ Utilities for the Z-100 and PC Series
MS-DOS Utilities with VAX/PDP-11 switch processing, DIR, COPY, DELETE, RENAME, TYPE with wildcards, full DATE processing, attribute processing, query, backup and more. **$ 49.00**

Z-100 PC Memory Expansion with ZPAL™

Z-151/Z-161 Users
Use 256K bit memory chips on the original memory board. Extend the memory to 640K or 704K bytes with ZPAL-2 Decoder **$29.00**

Z-158 Users
Extend your system to 768K bytes with 3 banks of 256K bit memory chips and ZPAL-158 decoder.
704K contiguous + 64K separate or 640K contiguous + 128K separate
ZPAL-158 Decoder **$ 36.00**

Z-138/Z-148 Users
Extend your system to 768K bytes with 3 banks of 256K bit memory chips and ZPAL-148 decoder.
704K contiguous + 64K separate or 640K contiguous + 128K separate
ZPAL-148 Decoder **$ 36.00**

ZSTEMpc-VT100 Choice of the U.S. Airforce 5,000-10,000 ZSTEMpc-VT100 Packages AFCAC 254

Call today: VISA and MasterCard accepted. Quantity and dealer discounts available
30 day money back guarantee. AIR MAIL POSTAGE included for US & Canada
KEA Systems Ltd.
2150 West Broadway, Suite 412, Vancouver, B. C. CANADA V6K 4L9
Technical Support (604) 732-7411
TELEX 04-352848 VCR **Order desk (800) 663-8702 Toll Free**

Zstem™ by KEA SYSTEMS

Trademarks VT100, VT220, DEC, VAX - Digital Equipment Corp. ZSTEMpc, ZPAL, DECkHAND - KEA Systems Ltd.

ZP-150 RAM EXPANSION
32K as low as $45

The ZP-150 is a great laptop. It lacks only one thing—sufficient on-line memory. Our 32K low-power CMOS modules plug into existing sockets in the computer to provide up to 416K of on-line, non-volitle memory. Upgrading your machine with from one to twelve modules takes a matter of minutes.

We are in our third year of providing quality memory upgrades similar to the 32K to thousands of satisfied portable computer users. Our memory modules are 100% factory tested and carry a one year materials and workmanship warranty. Because your satisfaction is our goal we offer a full refund 30 day return policy. **Priced at: $59 for 1 or 2**
$49 for 3 to 9
$45 for 10 or more

Illustrated step-by-step instructions included.

TO ORDER
CALL 714-540-1174
or WRITE
M/C, VISA, CHECK or MONEY ORDER
CALIF. RESIDENTS ADD 6% Sales Tax
SHIPPING-ADD
UPS Ground $1.50
2nd Day Air $4.00
Next Day Air $12.00

AMERICAN CRYPTRONICS INC.

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

Zenith is a registered trademark of Zenith Data Systems Corporation.

FBE Products

For the H/Z-150, 160 Series

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

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

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

FBE Smartwatch

Calendar/Clock using Dallas Semiconductor's DS1216E SmartWatch module. Works with H/Z-110/120, 138/148, 150/158. Package includes SmartWatch with our software and documentation. Spacer kit ($2) required for Z-100. ***$44.95***

For the H/Z-100 Series

ZMF100a — Modification package allows installation of 256K RAM chips in older Z-100 without soldering. Works only with old-style motherboard. ***$65***

ZRAM-205 — Kit allows 256K RAM chips to be put on Z-205 memory board to make 256K memory plus 768K RAM disk. Requires soldering. PAL ($8) required for new motherboard. ***$49***

For the H/Z-89, 90 Series

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

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

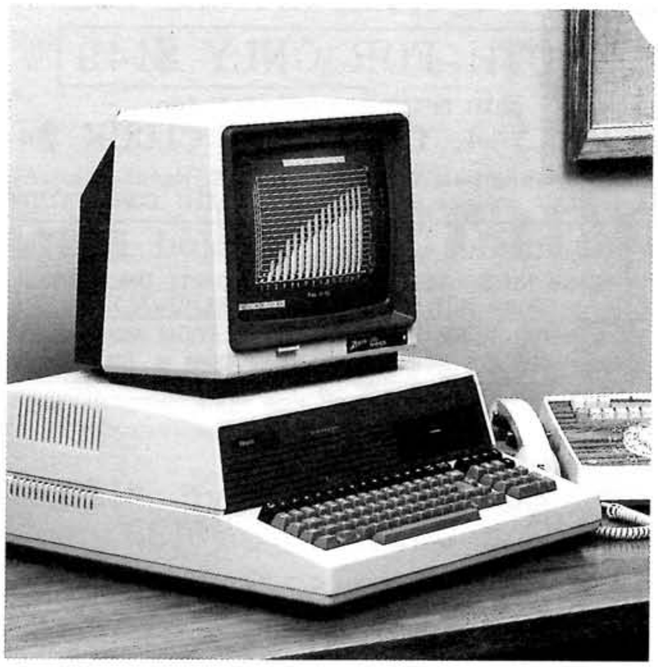
SLOT4 — Extender card adds 4th I/O expansion slot to right side bus. ***$47.50***

FBE

FBE Research Company, Inc.
P.O. Box 68234, Seattle, WA 98168
(206) 246-9815, M-F 9-5

UPS/APO/FPO Shipping Included.
VISA or MasterCard Accepted.

ZPC Update #14



Pat Swayne
Software Engineer

This is the fourteenth 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 ZPC Update, I will present some information for users of INT14.COM from the ZPC upgrade disk, a new patch for Generic CADD, and instructions for using GEM (Digital Research's Graphics Environment Manager) under ZPC.

INT14.COM Information

If you have a Scottie Board with optional serial ports, and you are trying to use INT14 to support them, here is some information I left out of the documentation on the first few hundred copies of the ZPC Upgrade Disk. When you load INT14, your computer should be in the Z-100 mode of operation. If it is not, you should switch to the Z-100 mode and then back to the PC mode before you try to run anything requiring the Scottie Board serial ports. The reason for this is that INT14 patches the serial port addresses within the memory image of ZPC from dummy values to the actual address of PC serial ports. When ZPC switches from the Z-100 mode to the PC mode, these values are put into a RAM area that may be used by programs. If you are already in the PC mode when you load INT14, the RAM area will not be updated until you exit and re-enter the PC mode.

Another Generic CADD Patch

The patch for Generic CADD that was released previously was for a release dated 7-21-86. A HUG member with another release, dated 10-2-86, found that the patch did not work, and developed a new patch for his version. If the CADD.EXE file on your disk is dated 10-2-87, put these lines in your PATCHER.DAT file.

```
GENERIC CADD V. 2.0  10-2-86
Insert the disk containing CADD.EXE.
CADD.EXE
DCCB,B0
DD35,B0
DD72,B0
DDDB,B0
DE2F,0,0
DE94,0,0
DF0F,0,0
z
```

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

Running GEM Under ZPC

Digital Research's GEM will run under ZPC without any patches or hardware support, but the installation requires some modification to a batch file on your GEM Master Disk, especially if you want to install it on 5.25-inch floppy drives. To prepare for installation for any kind of disk, first determine what the name of your FORMAT program is. On some early releases of MS-DOS version 3 for the Z-100, it is called FORMAT.EXE, but on other releases of MS-DOS 3 and all releases of MS-DOS 2, it is called FORMAT.COM. Now, prepare a system (bootable) disk containing FORMAT, PC.COM, and Z100.COM. This disk will be used when the GEMPREP program, used to install GEM, asks for a DOS disk.

Make a copy of the GEM System Master Disk, and edit the file GEMPREP1.BAT on this disk as follows. If the FORMAT on your system disk is called FORMAT.EXE, rename all instances of FOR-

MAT.COM to FORMAT.EXE. Rename all instances of MODE.COM to PC.COM. These are all the changes necessary if you are going to install GEM on a hard disk, but if you are going to install it on 5.25-inch floppies, locate this line in the GEMPREP1.BAT file:

```
FORMAT.COM B:/V
```

Add lines above and below it so that you have this:

```
Z100.COM
FORMAT.COM B:/V
PC.COM
```

Now, locate these two lines in the file:

```
ECHO
FORMAT.COM B:/S/V
```

Change them, and add lines so that you have this:

```
ECHO Press any key to continue.
PAUSE
Z100.COM
FORMAT.COM B:/S/V
PC.COM
```

Now you are ready to install GEM as directed in the manual. Set the PC mode before you begin installation. When you are asked to select a graphics card, select the first one from the menu. When you are asked for a printer port, select LPT1:, whether your printer is serial or parallel. If you have 8-inch drives, or 1.2 megabyte 5.25-inch drives operating as 8-inch drives, follow the procedure for hard disk installation and treat your high capacity floppy drives as hard drives. ✱

There is ALREADY A PC INSIDE Your H/Z-100. You can Bring it to life WITHOUT SPENDING the COST of a CLONE.

HUG ZPC Version 2.1.3 Software
Scottie Systems 'Scottie Board'

BOTH FOR ONLY $149

With OPTIONS AVAILABLE for
COM1 $44, COM2 $39, CLOCK $44.

Many Universities, Colleges, The U.S. Naval Academy, and MANY others use this EXCELLENT COMBINATION.

We make H/Z-100 Mod EASY
EASY NO SOLDER PREWIRED MOD KIT ONLY $5
(Does NOT REQUIRE M. BD. REMOVAL).

FREE WITH ORDER, H/Z-100 MODIFICATION AID DRAWINGS.
FREE PARTS KIT, YOU SOLDER. (REQUIRES M. BD. REMOVAL).

PLEASE CALL OR WRITE FOR MORE INFORMATION.

Scottie Systems
2667 Cropley Ave. #123
San Jose, CA 95132

(408) 259-6226

* ZPC V2 is a product of Heath Users Group

S & K Technology, Inc.
Quality Software for Heath/Zenith Microcomputers

For the Z100...

WatchWord® **$100.00**
The ultimate in word processing with speed and power. See subscripts, superscripts, underlining, and boldface directly on the screen. Create your own fonts and special characters. Other features include centering, formatting, automatic horizontal scrolling with long lines, large file capability, split screen, macros, color, and an extensive configuration facility. See reviews in Remark (July 1985) and Sextant (Jan-Feb 1985, Sep-Oct 1985). Requires 192K RAM.

The Resident Speller™ **$100.00**
Spelling checker for use with WatchWord.
Checks as you type from inside WatchWord or checks a file.
Includes a 50,000 word expandable dictionary.
Requires 192K RAM to check a file.
Requires 300K RAM to check as you type.

Demo disk for both **$ 3.00**

For IBM compatibles including the Z150 and Z200 series...

PC WatchWord® (New) **$ 99.95**
The ultimate in word processing for the sophisticated user. Most of the features of the Z100 version except for screen fonts. Requires 256K RAM.

PC Resident Speller™ **$ 99.95**
Spelling checker for ASCII files such as those created with WatchWord, WordStar, WordPerfect, PeachText, and VolksWriter. Includes Strike. Requires 256K RAM.

Strike™ **$ 49.95**
Adds as-you-type spelling checking to your word processor. Works with the word processors above and also with DisplayWrite, MultiMate and PFS:Write. Requires 100K RAM in addition to that used by your word processor.

Demo disk for Strike and The PC Resident Speller **$ 2.00**

Demo disk for PC WatchWord **$ 2.00**

Texas residents please add state sales tax.

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

MYPSC

A Character Or Graphics Print Screen Utility For The H/Z-100 And C.Itoh Prowriter Printer

Tom Riggs
215 S. Brookwood Drive
Auburn, AL 36830

We Z-100 owners know how tough it is to own excellent hardware that isn't compatible with the quasi-standards established by popularity. The problems that arise from this situation are not restricted to computers. It occurs in all areas of computer hardware/software. Specifically, a quasi-standard has been established by Epson simply because Epson has sold so many printers. There is a considerable amount of nice software that supports Epson printers, but if you don't own an Epson or true compatible, then you must fend for yourself. Thus the plight of C.Itoh Prowriter owners.

Zenith Data Systems (ZDS) is sensitive to the fact that Heath/Zenith owners buy equipment for its value (performance vs. cost) rather than its popular appeal. This is evidenced by that fact that ZDS took the time and trouble to provide six print screen utilities (named PSCxxx) with DOS 2.0, five graphic screendump utilities for specific printers and one generic character dump utility. In contrast, IBM only provides one such utility for none other than the IBM printer (an Epson clone). Unfortunately, the C.Itoh Prowriter is not one of the five that is supported. In fact, the generic utility does not work with the C.Itoh without modification. Don't take me wrong, I'm not complaining. I think it's great that Zenith went as far as it did. However, for those of us that own Prowriters, our systems are still in need of a very useful utility — a print screen utility.

After many wasted hours of trying to modify machine code written for other printers, I finally decided to write my own print screen routine for the C.Itoh. My first successful routine was written in ZBASIC. It worked fine, but was only good in BASIC. In general, not very useful. Then, along came Turbo PASCAL (a product of BORLAND International). Writing the utility in Turbo allowed me to refine and optimize the code, but it still wasn't very useful because the program had to be executed from DOS. I soon realized that I needed to write it in a language that created compact code and would allow me to make the utility resident. The obvious language was assembly. My only problem was I didn't know how to code the necessary algorithms in assembly language. This is where the Heath User's Group came to my aid. I dug out all my old issues of REMark and began reading the many "How to" and "Getting Started" articles on assembly language programming. I applied what I read to analyzing code written by a lot of sharp and talented people that were considerate enough to provide their source code. That's what I like most about HUG software. The source code is usually provided. Well, after many painstaking hours, I learned enough to produce my own resident print screen utility. The final product I affectionately call MYPSC.

Overview Of MYPSC

MYPSC is different from most print screen utilities I've seen in that it is really two utilities in one. It is both a high speed character dump routine (complements of PSC) and a moderate speed graphics dump routine (complements of me). MYPSC is a resident utility like PSC and is installed and activated in the same way that PSC is installed and activated. From here the similarity ends. Once activated, MYPSC waits for about five seconds for input from the keyboard. If no key is pressed in that time period, MYPSC performs an ASCII character dump to the printer. In the ASCII dump mode, all non-ASCII characters are converted to printable characters or replaced by spaces. If MYPSC detects a key input (any key other than F12) in the first five seconds, it performs a high resolution graphics dump to the printer. In this mode, every byte in active video memory is sent to the printer. In order for the printer to handle this data properly, MYPSC sends the proper Escape codes to the printer to tell it the forthcoming data is bit-mapped graphics data and to switch to graphics mode. The output is expanded and printed on its side so the reproduced screen fits nicely on 8" × 10" paper. You may be wondering why have both modes if the graphics mode will handle everything. The answer is speed. A complete graphics dump takes almost two minutes whereas the ASCII dump takes about 15 seconds.

How MYPSC Works

MYPSC can be divided into three functional blocks, an installation routine (INSTALL), an interrupt handler (INT__5), and the actual screen dump routine (MAIN). The purpose of INSTALL is to make INT__5 and MAIN resident and to tell the system where to go in memory when an interrupt number 5 is detected. So what's an interrupt 5? An interrupt 5 is a system interrupt that is generated when the <SHIFT> key is hit simultaneously with the <F12> key. <SHIFT-F12> is the only key combination that generates an interrupt code. Let's see how INSTALL does its job.

Although INSTALL is the first block of code to be executed, it is placed at the end of the code. There is a reason for this madness. INSTALL is only needed once, at installation. There is no useful purpose in making INSTALL resident. Doing so would only consume precious memory. By placing INSTALL at the end and telling the system that the end of the resident code occurs before INSTALL, the memory needed by INSTALL can be retrieved for other purposes after INSTALL has executed. This is actually done by setting DX equal to the memory location that marks the end of the resident code "INT__END" and executing a DOS call "INT 27H" (Exit, but stay resident). DOS remembers the memory location assigned to DX and protects the code that precedes it. But then you may ask, "How is INSTALL executed first?" Near the top of the program is the label "BEGIN:". This label marks the beginning of the executable code. The first command after the label BEGIN is a jump to INSTALL — "JMP NEAR PTR INSTALL". Thus the program jumps over all the code in INT__5 and MAIN and executes INSTALL. So now INSTALL has made INT__5 and MAIN resident, but how does the system know where to go when an interrupt 5 is detected. Well, it doesn't know exactly, but rather figures it by a stepping stone method. What the system does know is that when an interrupt 5 is detected, it should go to memory location 0000:0014 (hex). (This is a decimal 20 which equals 5*4). This memory location holds the address of the beginning of the resident code (INT__5 in this case). So the system goes to 0000:0014, gets the address of the program, goes to that location, executes the program, and returns. The only thing left to figure out how the correct address gets loaded into 0000:0014. Well you guessed it, INSTALL puts it there. Note the code in INSTALL beginning with "XOR AX,AX" and ending with "STI". Now that we have a good grasp on how the routine is installed, let's see what happens when it is activated.

INT__5's purpose in life is to save the current state or conditions of the computer, call the print screen module and restore the computer to its original state. The program must save the current state and then restore it so that once the dump is completed, the system will be returned to its entry state just like nothing ever happened. If this isn't done, the system will likely go off into the wild blue yonder or even worse begin eating disks and/or other weird things. INT__5 first saves the address of the current stack (a stack is a reserved set of memory locations, often called a buffer) and then sets up its own stack. INT__5 then PUSHes key system state information onto its own stack. Once everything is saved, INT__5 calls MAIN for execution. Once MAIN is finished the program returns to INT__5. INT__5 then proceeds to restore the computer to its original condition. Up to this point, the program has been general in nature. That is to say it has been independent of the mission of MAIN. This is nice because INSTALL and INT__5 really form a general routine for installing and executing (by an Interrupt 5) any function. There is one key restriction on the type of code that goes in MAIN. MAIN can not use any DOS calls. All I/O must be performed directly through BIOS making the routine hardware dependent. Sorry, Z-150 user's. OK, let's see what MAIN does.

MAIN is divided into three subroutines; GET__KEY, CHR__DUMP, and GR__DUMP. The first subroutine that MAIN calls is GET__KEY. GET__KEY does just what its name implies, it gets input from the keyboard. GET__KEY is designed to wait for about five seconds for key input. If a key is struct in that time, the value of the key is placed in the AL register and GET__KEY returns to MAIN. (Each key is assigned a unique code or value ranging from 0 to 255. Refer to the technical manual under programming data for more detail on these codes.) If no key has been entered after the 5 second polling period, GET__KEY puts a zero in AL and returns to MAIN. Next, MAIN checks the AL register. If AL=0 then CHR__DUMP is called; otherwise, GR__DUMP is called.

CHR__DUMP is the subroutine responsible for generating the high speed ASCII dump. Although I would like to take credit for it, I can not. I extracted the routines from the file PSC.ASM that came on the Z-DOS distribution disk. I did make a few modifications to the code. One modification was necessary to make it work with the C.Itoh. The original code sent a linefeed followed by a carriage return (LF/CR) to the printer at the end of each line. This had to be changed to a CR/LF. The other modification was optional and was done simply to save memory. CHR__DUMP is a simple procedure. It first calls the routine INT__SIZE to check if the 25th line is activated. If it is, then the variable "LINES" is set to 25. If the 25th line is not activated, then LINES is set to 24. CHR__DUMP then calls INT__SCRN to actually perform the character dump. INT__SCRN operates on each line starting at the top of the screen and working down one line at a time. INT__SCRN processes each line in the same way repeating the operation 24 or 25 times depending on the value of LINES. Let's see how the first line is processed. INT__SCRN first sets the cursor location to the far left position of the first line. It then gets the character at that position through the monitor BIOS routine. INT__SCRN then makes sure the character does not have a value above 7F (hex) by setting the eighth bit to zero. Next, the character is checked to make sure it's a printable ASCII character (20H-7FH). If it isn't valid, then the character is replaced by a space. The character is then placed in a temporary buffer called VPTR. The cursor is then moved one position to the right and the process is repeated until column 80 has been retrieved. Once the column 80 character has been processed, the entire line is in VPTR. INT__SCRN appends a CR and a LF to the line in VPTR and sends the 82 characters in VPTR to the printer. The cursor is then set to the left most position on the next line and the process is repeated. This entire procedure takes about 15 seconds on my printer.

GR__DUMP performs the graphics dump routine. This routine reproduces the screen completely. The method for doing this is really quite simple once one understands how the screen is created and how the printer executes bit mapped graphics. I've written screen dump routines for a number of different printers including the Epson, Gemini, Okidata, NEC, and IDS Prism and, without a doubt, the easiest one to program was for the C.Itoh. The Z-100 video board has its own random access memory (RAM) separate from the system RAM. The video RAM stores all the information that appears on the screen. The picture that one sees on the monitor screen is actually an array of dots or pixels. The pixels are turned on or off to create the pattern on the screen. The condition of each pixel (on or off) is stored in the video RAM as a bit of digital information. The Z-100 screen is made up of 640 pixels horizontally by 225 pixels vertically. The actual mapping of each bit in video RAM to a corresponding pixel is fairly straight forward. Randy Meyers wrote a good article on how this is accomplished for the May 1984 issue of REMark. (There were several minor errors in the article. Corrections appeared in the July '84 issue in the "BUGGIN' HUG" column.) Also, the Video Logic Board section of the Z-100 technical manual gives a very good explanation of the mapping. If you don't own a set of the hardware tech manuals, I recommend that you purchase a set. There is a tremendous amount of information in them that is useful for programming. The bits corresponding to the CRT pixels are formed into bytes of information. These bytes are stored in the video memory and can be accessed just like any other byte in memory. The C.Itoh conveniently performs bit mapped graphics one byte at a time. Thus, all one must do to re-create the screen on the printer is to read each significant byte of video memory and send those bytes in an ordered manner to the printer. Of course, the last statement is easier said than done, but it is the essence of the screen dump routine. Let's look at the details of the operation.

GR_DUMP consists of four sub-modules; DEF_MVEC, INIT_HW, DO_DUMP, and RESET_HW. DEF_MVEC calculates the offset address of the left most column of video data and stores those addresses in the array VPTR. INIT_HW initializes the hardware in preparation for the dump routine. DO_DUMP performs the graphics print screen routine. Finally, RESET_HW resets the hardware to a normal state. These four sub-modules are called sequentially.

DEF_MVEC is used to save execution time. The video screen is arranged in matrix form of 225 rows by 80 columns. On any row, the columns are counted sequentially from left to right. As an example, let's say that the first (left most) column of a particular row has a memory offset address of 1024. The second column on that same row will have an address offset of 1025. The 47th column would have an address of 1024+46=1070. The 80th column would have an address of 1024+79=1103. Thus the address for any row/column location is equal to the address of the first column of the row + column number - 1. The actual determination of the row addresses is rather complicated. To save time, these addresses are determined only once and then stored in the buffer VPTR for future reference. The equivalent algorithm for the calculation of these addresses is given below in BASIC.

```
10 K=1
20 FOR J=0 TO 24
30 FOR I=0 TO 8
40 VPTR(K) = (I + J*16)*128
50 K = K + 1
60 NEXT I
70 NEXT J
```

INIT_HW enables the video board for access to the video memory and sends a command string to the printer preparing it for bit-mapped graphics. The video board is enabled by setting the 7th bit (most significant bit) of the video control port low. The command string that is sent to the printer sets the pitch to 10 cpi; selects elongated (×2) operation; sets the line spacing to 1/9th inch; and selects unidirectional printing. Now we're finally ready to do what this program is all about. Let's dump the screen!

DO_DUMP operates on columns from right to left. To process a column of data, DO_DUMP performs three operations. First, it tells the printer that the next 225 bytes of data should be considered graphics data (versus ASCII data). Next, it reads from video memory each of the 225 bytes of data that make up the column and sends each byte to the printer as it is read. Finally, after reading and sending all 225 bytes of data, it sends a CR/LF to the printer in preparation for the next column of data. Once a column is processed, the procedure is repeated operating on the next column to the left. Once column 1 has been sent to the printer, the dump is complete. All that is left now is to restore the hardware to its original configuration. That's the function of RESET_HW.

RESET_HW disables the video board and resets the printer to normal ASCII character mode. The video board is disabled by setting the 7th bit of the video control port high. The command string that is sent to the printer turns the elongated character size (×2) off; sets the pitch to 10cpi; sets line spacing to 1/6th inch; and selects bi-directional print mode. Once RESET_HW is executed, program control is returned to MAIN which in turn transfers control to INT_5 so INT_5 can restore the system to its original state.

Possible User Modifications

There are a number of possible modifications that an individual may want to make to suit his or her own preferences. Let me suggest a few possible changes and how to make them.

1. WAIT FOR INPUT delay. The program is set to provide an approximate five second wait state after the program is activated. This time window is based on the assumption that the computer is operating at 5 MHz. If your system is running at another frequency, or if the five second wait is annoying, then you may want to change the program. The wait state is determined by two countdown counters fashioned in a nested loop. Refer to the GET_KEY subroutine in the listing. The sixth and seventh lines are

```
         MOV  DX,06       ; BEGIN LOOPING ^
                            WAITING FOR INPUT
LOOP_BG:MOV  CX,0FFFFH
```

   Adjust the "06" to make the delay longer or shorter. On an 8 MHz machine change "06" to "10" for a five second wait period.

2. COLOR OPTION. This option can be very dangerous. In general, dumping all three color planes of video data to a non-color printer will normally result in a surprising picture. However, I put the option in for those that like to live dangerously. If you want to send the logical OR of the three planes to your printer, set COLOR to TRUE near the top of the listing.

3. RESET PRINTER DEFAULTS. I normally use a pitch of 10 cpi and a line spacing of 1/6th inch for text output. If you prefer another pitch or line spacing, then the code in RESET_HW will have to be modified. The string that is sent to the printer is coded on the line just before the command RET. At present this code is

```
DB    CR,X20FF,ESC,'A',ESC,'N',ESC,'<'+80H
```

   Refer to your printer manual for the escape codes that you want to use. Note — ESC=CHR$(27). Don't forget to add 80H to the last character in your command string.

Assembly And Installation

Use your favorite text editor to create your source file, MYPSC .ASM, and enter the code in the listing. If you want to save typing effort, don't include the comments. If you really don't want to fool with it at all, send me $5 and a disk and I'll send you your disk back with the source code, the assembled file and another C.ITOH utility that I wrote.

Once your source file is created, you're ready to assemble the program. Assembly will require the following programs.

MYPSC.ASM	- Source file you created from Listing
DEFMTR.ASM	- Monitor definition file. Comes on Z-DOS Distribution Disk #2
MASM.EXE	- MACRO assembler
LINK.EXE	- Linker program
EXE2BIN.EXE	- Converts .EXE files to binary files

Refer to the comments at the top of the listing for the specific assembly commands.

Install it by typing in the command MYPSC. After installation, you can activate your print screen utility anytime by hitting <SHIFT-F12>. Enjoy!

Listing 1

```
   PAGE,132
   TITLE MYPSC ;CHARACTER OR GRAPHICS SCREENDUMP PROGRAM
               ;     ZENITH Z-100 TO C.ITOH PROWRITER

;***MYPSC is a print screen utility for the H/Z-100 com-
;   puter and the C.Itoh Prowriter printer. It has the
;   capability to dump the screen as a set of ASCII
;   characters or the entire screen bit pattern for
;   graphics screen reproduction. In ASCII mode only
;   recognizable ASCII characters are sent to printer. In
;   graphics mode the entire screen is sent to the printer
;   with output expanded and turned on it's side to fit
;   on an 8"x10" piece of paper. The mode is user
;   selected at activation without disturbing the screen.
;
;   The code used for the ASCII dump was largely derived
;   from the source listing PSC.ASM that comes with the
;   Z-DOS operating system disk from ZENITH DATA SYSTEMS.
;
;   The remaining code was written by Tom L. Riggs, Jr.
;                                     215 S. Brookwood Dr.
;                                     Auburn, AL 36830
;
;   To assemble:
;
;   MASM MYPSC;
;   LINK MYPSC;
;   EXE2BIN MYPSC.EXE MYPSC.COM
;
;   To install:
;
;   MYPSC
;
;   Once installed MYPSC will reside in RAM waiting for
;   activation. MYPSC can then be activated at any time.
;
;   To Activate:
;
;   Press SHIFT and F12 simultaneously
;   If you desire ASCII dump do nothing more
;   If you desire GRAPHICS dump hit any key other than F12
;   MYPSC will wait approximately 5 secs for key input.
;

BIOS_SEG SEGMENT AT 40H
        ORG 3*3
        BIOS_CONOUT LABEL FAR
        ORG 4*3
        BIOS_PRINT LABEL FAR
BIOS_SEG ENDS

INCLUDE DEFMTR.ASM                  ;Make sure this file is
                                     on Default Drive

COLUMNS =       80                  ; Characters/line

ESC     EQU     27
CR      EQU     13
LF      EQU     10
X2ON    EQU     14
X2OFF   EQU     15
V_PORT  EQU     216
BLU_PL  EQU     0C000H
RED_PL  EQU     0D000H
GRN_PL  EQU     0E000H
KEYDAT  EQU     0F4H
KEYCMD  EQU     0F5H
KEYSTAT EQU     0F5H
TRUE    EQU     0FFFFH
FALSE   EQU     NOT TRUE
COLOR   EQU     FALSE                   ;SET COLOR TO TRUE IF
                                         YOU WANT ALL PLANES
M       EQU     BYTE PTR 0[BX]

CODE    SEGMENT BYTE PUBLIC 'CODE'
        ASSUME  CS:CODE,DS:CODE,ES:CODE,SS:CODE

        ORG     100H
BEGIN:
        JMP     NEAR PTR INSTALL

;*      INT 5 handler
;

INT_5:
        PUSH    DS              ; Save his data segment
        PUSH    AX

        MOV     CS:INT_SP,SP
        MOV     CS:INT_SS,SS    ; Save his stack

        MOV     AX,CS
        MOV     SS,AX           ; Set local stack
        MOV     SP,OFFSET INT_STACK

        STI                     ; Now allow interrupts to
                                  come in

        PUSH    ES
        PUSH    BX
        PUSH    CX
        PUSH    DX
        PUSH    SI
        PUSH    DI

        MOV     DS,AX
        MOV     ES,AX            ; Set up my segments

        CALL    MAIN

        POP     DI
        POP     SI
        POP     DX
        POP     CX
        POP     BX
        POP     ES

        CLI                      ; INTs off while messing
                                   around
        MOV     AX,INT_SP
        MOV     SP,AX
        MOV     AX,INT_SS        ; Restore his stack
        MOV     SS,AX

        POP     AX
        POP     DS
        IRET                     ; Back to BIOS

INT_SP  DW      0                ; Saved stack values
INT_SS  DW      0
I       DW      0                ; Internal pointers
J       DW      0
ID      DW      0
VPTR    DW      225 DUP(?)       ; Column 1 Video Addr's
                                   go here
INT_CX  DW      0                ; Current X location
INT_CY  DW      0                ; Current Y location
MYCHR   DW      0                ; Character at position
LINES   DW      0                ; Number of lines valid
                                   on screen
        DW      128 DUP (?)
INT_STACK LABEL NEAR                ; Internal stack

MAIN:
        CALL    GET_KEY             ; Go see if key pressed
        OR      AL,AL               ; AL=0 if no key pressed
        JZ      DO_CHR              ; do ASCII dump
        CALL    GR_DUMP             ; else do Graphics dump
        RET
DO_CHR:
        CALL    CHR_DUMP
        RET

GET_KEY:                            ; Checks for key input
                                      for 5 secs

RST_KB: IN      AL,KEYSTAT          ; Reset and Clear Keyboard
        AND     AL,02               ; encoder Buffers
        JNZ     RST_KB              ; wait till ready for
                                      command
        MOV     AL,00               ; RESET TO POWER-UP
                                      CONDITIONS
        OUT     KEYCMD,AL           ; Do it

;--- Now let's wait for key input

        MOV     DX,06               ; BEGIN LOOPING - WAITING
                                      FOR INPUT
LOOP_BG:MOV     CX,0FFFFH
WAIT_K:
        IN      AL,KEYSTAT          ; POLL THE KEYBOARD
        CMP     AL,1                ; HAS A KEY BEEN STRUCK?
        JNZ     WAIT_K1             ; IF NOT, WAIT SOME MORE
        IN      AL,KEYDAT           ; ELSE GET THE KEY
        CMP     AL,0E2H             ; Shft F12 ?
        JZ      WAIT_K1
        JMP     GOT_KEY             ; EXIT
WAIT_K1:LOOP    SHORT WAIT_K        ; IF CX <> 0 THEN LOOP
        DEC     DX                  ; UPDATE BIG_LOOP COUNTER
        CMP     DX,0                ; CHECK FOR ZERO
        JNZ     LOOP_BG

        MOV     AL,00H              ; NO KEY WAS HIT. SET
                                      KEY TO NULL.
GOT_KEY:
        RET

GR_DUMP:
        CALL    DEF_MVEC            ;Define the video memory
                                     map
        CALL    INIT_HW             ;Initialize the hardware
        CALL    DO_DUMP             ;Let's do the dump
        CALL    RESET_HW            ;Reset the hardware
        RET

DEF_MVEC:                           ;Calculate the column 1
                                     memory addr
                                    ;to be stored in VPTR
        xor     AX,AX               ;Clear AX
        mov     ID,AX               ;Initialize counters
        mov     J,AX
J_LP:
        cmp     J,25                ;Check to see if done
        jge     DUN_PTR
        xor     AX,AX
        mov     I,AX                ;Set I=0
I_LP:
        cmp     I,9                 ;See if inner loop is done
        jge     NXT_J
        mov     BX,128              ;Do the calculations
        mov     CX,16
        mov     AX,J
        imul    CX
        add     AX,I                ;AX contains solution
        imul    BX                  ;Figure out where to
                                     put it
        lea     SI,VPTR
        mov     DX,ID
        shl     DX,1
        add     SI,DX
```

```
        mov     [SI],AX             ;Store the solution
        inc     ID                  ;Get set to do next
                                     calculation
        inc     I
        jmp     SHORT I_LP          ;Do inner loop again
NXT_J:
        inc     J
        jmp     SHORT J_LP          ;Do outer loop again
DUN_PTR:
        RET

INIT_HW:
        IN      AL,V_PORT           ;Get current video port
                                     status
        AND     AL,7FH              ;Set to Enable it
        OUT     V_PORT,AL           ;Enable video port
        CALL    INT_PRT             ;Set Printer for Dump
        DB      CR,ESC,'N',X20N,ESC,'T16',ESC,'>'+80H
        RET

DO_DUMP:
        MOV     CX,80               ;Initialize CX to Loop
                                    ;One time for each column
J2_LP:
        CALL    INT_PRT             ;Tell Pntr that graphic
                                     data is coming
        DB      '         ',ESC,'S022','5'+80H

        XOR     AX,AX
        MOV     I,AX                ;Set row counter to zero
I2_LP:
        CMP     I,225               ;PRINTED ALL 225 ROWS?
        JGE     NXT_J2              ;IF SO, LET'S DO NEXT
                                     COLUMN
        LEA     BX,VPTR             ;Load addr of VPTR in BX
        MOV     DX,I                ;Move row pointer to DX
        SHL     DX,1                ;Multiply by 2 - Dealing
                                     with words
        ADD     BX,DX               ;Now have addr of VPTR
                                     element
        MOV     SI,[BX]             ;Load source ind with addr
                                     in VPTR ele.
        ADD     SI,CX               ;Add column offset
        MOV     BX,GRN_PL           ;First get the Green
        MOV     ES,BX               ;Set Extra Segment to
                                     Green plane Seg
        MOV     AL,ES:[SI-1]        ;Now get the byte in
                                     green memory

        IF      COLOR
        MOV     BX,BLU_PL           ;Do Blue plane
        MOV     ES,BX               ;Set Extra Segment to Blue
        MOV     BL,ES:[SI-1]        ;Get Blue byte
        OR      AL,BL               ;Combine it with Green byte?
        MOV     BX,RED_PL           ;Do Red Plane
        MOV     ES,BX               ;Set Extra Segment to Red
        MOV     BL,ES:[SI-1]        ;Get Red Byte
        OR      AL,BL               ;Combine it with Blue
                                     & green
        ENDIF

P_BYTE: CALL    FAR PTR BIOS_PRINT ;Send it to the printer
        INC     I                   ;Get ready for next row
        JMP     I2_LP               ;Do it again
NXT_J2:
        CALL    INT_PRT             ;Send CR/LF to printer
        DB      CR,LF+80H
        LOOP    J2_LP               ;Do next column
DUN_DMP:
        RET

RESET_HW:
        IN      AL,V_PORT           ;Reset video port
        OR      AL,80H
        OUT     V_PORT,AL

        CALL    INT_PRT             ;Reset Printer
        DB      CR,X20FF,ESC,'A',ESC,'N',ESC,'<'+80H
        RET
```

```
INT_PRT:                        ;Print string : Last
                                 byte has 8th bit set
        MOV     BP,SP
        XCHG    BX,[BP]
INT_PRT1:
        MOV     AL,M            ;Get Character
        CALL    FAR PTR BIOS_PRINT ;Send to Printer
        INC     BX              ;Point to next char
        OR      AL,AL           ;Check to see if 8th
                                 bit set
        JNS     INT_PRT1        ;If not get next
                                 Character
        MOV     BP,SP
        XCHG    BX,[BP]
        RET

CHR_DUMP:
        CALL    INT_SIZE
        CALL    INT_SCRN
        RET

;*      INT_SIZE - Size Screen
;
;       INT_SIZE checks to see if the 25th line is
;       enabled. If it is, LINES is modified to be 25,
;       so all lines are printed
;

INT_SIZE:
        MOV     WORD PTR LINES,24   ; Assume not
        MOV     SI,OFFSET INT_SIZA  ; Print first line
        CLI                         ; Don't interrupt me
                                      now
        CALL    INT_PLINE           ; Output line
        PUSH    DS
        XOR     AX,AX
        MOV     DS,AX               ; Get data segment
        MOV     DS,DS:[MTR_DS]
        MOV     AL,DS:[MTR_VERP]    ; AL = horiz. position
                                      of cursor
        POP     DS
        PUSH    AX
        MOV     SI,OFFSET INT_SIZB  ; Put cursor back
        CALL    INT_PLINE           ; Output line
        STI
        POP     AX
        CMP     AL,24
        JNZ     INT_SIZ1            ; If not on 24th line
;       Cursor moved to 25th line, must be enabled

        INC     WORD PTR LINES
INT_SIZ1:
        RET

INT_SIZA LABEL  NEAR
        DB      01BH,'j'            ; Save cursor
        DB      01BH,'Y',25+31,1+31 ; Go here
        DB      -1                  ; End of message

INT_SIZB LABEL  NEAR
        DB      01BH,'k'            ; Restore cursor
        DB      -1

;*      INT_PLINE - Print line on console
;

INT_PLINE:
        MOV     AL,BYTE PTR [SI]    ; Get character
        INC     SI                  ; Bump for next time
        CMP     AL,-1
        JZ      INT_PLIN1           ; If end of text
        PUSH    SI
        CALL    FAR PTR BIOS_CONOUT ; Output it
        POP     SI
        JMP     SHORT INT_PLINE     ; Do next
INT_PLIN1:
        RET

;*      INT_SCRN - Interrupt time Screen Processor
;
INT_SCRN:
        MOV     WORD PTR INT_CX,0
        MOV     WORD PTR INT_CY,0   ; Show line 0, column 0

        MOV     SI,OFFSET VPTR      ; Save one line here

;       Do a screen line

INT_SCRN1:
        PUSH    DS
        PUSH    SI

        PUSH    DS
        MOV     AX,OFFSET MYCHR
        PUSH    AX                  ; 32 bit address for
                                      character

        PUSH    WORD PTR INT_CY     ; Vertical location
        PUSH    WORD PTR INT_CX     ; Horizontal location

        XOR     AX,AX
        MOV     DS,AX
        MOV     DS,DS:[MTR_DS]      ; Pointer to MTR data area

        CALL    DS:DWORD PTR MTR_RDC ; Read it

        POP     SI
        POP     DS

        MOV     AL,BYTE PTR MYCHR
        ADD     AL,' '              ; AL = character

;       AL = character, H19 graphics are numbered 80H - A1H

        AND     AL,07FH             ; Make it printable
        CMP     AL,' '
        JNC     INT_SCRN2           ; If printable
        MOV     AL,' '              ; Not printable,
                                      make it a space
INT_SCRN2:

;       Save character

        MOV     BYTE PTR [SI],AL
        INC     SI

        INC     WORD PTR INT_CX     ; Go to next character
        CMP     WORD PTR INT_CX,COLUMNS
        JC      INT_SCRN1           ; If can do more

;       End of character line, print the whole thing

        MOV     SI,OFFSET VPTR
        MOV     WORD PTR INT_CX,0   ; Reset it
INT_SCRN3:
        MOV     AL,BYTE PTR [SI]    ; Get character
        INC     SI
        PUSH    SI
        CALL    FAR PTR BIOS_PRINT  ; Print it
        POP     SI
        INC     WORD PTR INT_CX     ; Do next
        CMP     WORD PTR INT_CX,COLUMNS
        JC      INT_SCRN3

;       Line output, follow with CR/LF

        PUSH    SI

        MOV     AL,0DH
        CALL    FAR PTR BIOS_PRINT  ; do CR
        MOV     AL,0AH
        CALL    FAR PTR BIOS_PRINT  ; do LF

        POP     SI

        MOV     WORD PTR INT_CX,0   ; Start a new line
        MOV     SI,OFFSET VPTR      ;  here

        INC     WORD PTR INT_CY     ; next horizontal line
```

Continued on Page 73

Exorcising Demons In An H-89

Frederick M. Galloway
6507 Blue Wing Drive
Alexandria, VA 22307

Here is a program that I find quite useful, that shows some techniques for using a C Compiler popular with H-89 owners, and that gives you some useful functions to use in your future programs. This article is for someone who knows the basic principles of the C language such as you might get from the REMark series "An Introduction to 'C'", by B. Polk. The program exorcizes demons in an H-89.

"What, demons in an H-89?" you ask. Well, I found enough in mine when I tried to read a key ".doc" file sent me by HUG. When I loaded this program into my editor, the screen showed a strange mishmash of graphics and reverse video interspersed among regular ASCII characters. My Epson printer did worse when I sent this file to type. So I started decoding character by character. I took a break, and when I reloaded the ".doc" file into the editor I discovered that some of the offending characters that I had changed into normal ASCII had changed back. Worse, the funny characters seemed to be corrupting their nearest neighbors. On the other hand, HDOS utilities "type" and "copy" had no trouble putting clean copy on the screen. This made me curious about what other than regular characters might be lurking in ASCII files.

I suspected that the ".doc" file had come from a word processor that set the eighth bit of many characters. This action increases their value to over 128 and thus places them outside the range of ASCII characters (0 – 127). I wanted some sort of program to look at all nonprinting characters. I recalled that there was a Unix utility to do this. Next the program should set the eighth bit to zero thus converting non-ASCII characters to their ASCII equivalents. Finally it should strip out any remaining nonprinting characters (generally the control characters and delete or 0 - 32 and 127.) I wanted to see the output on the screen, print it or store it in a file.

This would be a good project for my C/80 compiler from the Software Toolworks(registered trademark). The program would be fast to write, fast to assemble and fast in operation. It takes up relatively little space in memory. It uses command line arguments to choose the input file and the specific functions performed (showing the nonprinting characters, stripping off the eighth bit of characters with it set and deleting nonprinting characters). Finally input/output redirection allows sending the output to wherever desired without additional programming effort or memory requirements. (This feature, with others is built into the programming overhead that causes the null program, main() { }, to take up 1310 bytes in memory.)

I decided to follow the C/Unix practice of using command line arguments. Thus "view dvn:filename.ext" causes the program "view" to show on the screen any nonprinting characters as their octal equivalents in the file, "filename.ext" on the drive, "dvn:". Using a command line option , "–c", causes the eighth bit to be set to zero on all characters. Then any remaining nonprinting characters are displayed as octal numbers. "–s" causes the program to strip any nonprinting characters before displaying the file. If "–c" and "–s" are used together, then characters are first converted to the ASCII range of 0–127; only the nonprinting characters then remaining are stripped. A ">" followed by a device name or file specification directs the output that would normally go to the screen to the device or file named. Thus the command line, "view –cs story.ws >lp:", reads the file "story .ws", converts any characters in it above 128 to the ASCII range, strips any remaining nonprinting characters, and sends the output to the printer.

The main program, called view, is shown in Listing 1. There are several features of this program that were dictated by the characteristics of the Software Toolworks C/80 compiler. (The program was compiled and tested under HDOS, and should compile under CPM without change. Only minor changes, if any, should be needed with other C Compilers.)

The first feature is the absence of an "#include <printf.c>" statement. "printf" uses a lot of space. Unless there is a lot of formatted or justified output, you are better off to avoid "printf". I used "fputs" for string output. "fputs" is shown in listing 5. With "printf" the assembled or ".abs" program took 15 sectors(of 256 bytes each). With "fputs" the program used only 9 sectors.

Listing 1 - Main Program

```
001: /*   view:  make funny characters visible */
002:
003: #define          FILE    int
004: extern int       fin,finout ;
005: #define          stdin   fin
006: #define          stdout  fout
007: #define          stderr  0
008: #define          EOF     -1
009: #define          NULL    0
010: int              convert = 0 ; /* 1 => drop 8th bit */
011: int              strip = 0 ; /* 1 => discard special
                                        characters */
012: #include         "vis.c"
013: #include         "efopen.c"
014: #include         "octnum.c"
015:
016: main(argc,argv)
017: int      argc ;
018: char     *argv[] ;
019: {    /* main */
020:     FILE     *fp ;
021:     char     *s ;
022:
023:     while (--argc > 0 && (*++argv)[0] == '-')
024:         for (s = argv[0]+1; *s != '\0'; s++)
025:             switch (*s) {
026:             case 'S':   /* -s:strip funny chars */
027:                     strip = 1 ;
028:                     break ;
029:             case 'C':   /* -c: convert funny
                                        characters to ASCII */
030:                     convert = 1 ;
031:                     break ;
032:             default:
033:                     fputs("\"",stderr) ;
034:                     putc(*s,stderr);
035:                     fputs("\" is an illegal
                                   option\n ",stderr) ;
036:             }    /* switch */
037:     if (argc != 1)
038:         fputs("Usage: view -c -s filename\n",stderr) ;
039:     else {
040:         fp = efopen(*argv,"r") ;
041:         vis(fp) ;
042:         fclose(fp) ;
043:     }    /* else */
044: }    /* main */
045: #include "stdlib.c"
```

The "#defines" and declaration in the first five lines of the program are a kludge to bring C/80 nomenclature for input and output into line with current C-language practice.

Lines 23 to 36 process any command line options. The first line looks for any strings starting with "-". The second processes each letter behind the "-" until the end of the string. Thus the program treats "-c -s", "-s -c", "-cs" and "-sc" as the same. The "switch-case" construct was used instead of a series of "else-if's" for ease of expansion and readability. Upper case letters are used as the arguments of the "case" function. This is because HDOS changes any lower case letters on the command line to upper before passing them to the program.

Lines 33-35 replace the expression "fprintf("\\%s\\ is an illegal optionn",*s,stderr)". As I indicated above, replacing one line with three really takes a lot less memory and speeds up compilation and execution of short programs when you also consider that you avoid the overhead associated with "printf". Since the user may well specify output redirection, output statements should carefully distinguish between information to go to the standard output (which may be a file or device) and that which the user needs immediately. The latter goes to "stderr" which is the screen regardless of where "stdout" is. Directing output anywhere other than "stdout" requires "fputs" or "putc" rather than "puts" or "putchar".

After processing any options, there should be one command line argument left, the name of the file to be viewed. Line 37 checks for the file name. (Remember that any redirection of input or output is done before "main" takes charge and is not the responsibility of the program.)

In line 40 I have used a function to open a file rather than including the code in "main". This is because I am lazy. I typed and debugged the function once. Now I just "#include" it whenever needed. It is shown in Listing 2.

Listing 2 — "efopen" Function

```
FILE     *efopen(file, mode)       /* fopen file, die if
                                      cannot */
char     *file, *mode ;

{    /* efopen */

         FILE            *fp ;

         if ((fp = fopen(file, mode)) != NULL)
                 return (fp) ;

         fputs("Cannot open file ",stderr) ;
         fputs(file,stderr) ;
         fputs(" mode ",stderr) ;
         fputs(mode,stderr) ;
         putc('\n',stderr) ;
         exit() ;

}   /* efopen */
```

The function "vis" (line 41) with "octnum" which it calls does most of the work. The code is fairly straight forward. "vis" does not translate spaces, tabs, or newlines into their octal equivalents. They perform their normal functions. This makes the output look more like the input. You may want to delete '| | c == '\t'" when checking how a program converts spaces to tabs.

"octnum" merely duplicates a formatting capability that "printf" provides. Note the cast (change of type) in the first line of the body of "octnum". Without a cast, C/80 treats a character whose value is over 127 as a negative number and "octnum" returns garbage. "vis" and "octnum" are shown in Listings 3 and 4.

Listing 3 — "vis.c" Function

```
vis(fp)     /* make chars visible in file *fp */
FILE     *fp ;
{    /* vis */
         int      c ;

         while ((c= getc(fp)) != EOF) {
                 if (( c > 127) && (convert))
                         c -= 128 ;
                 if (isascii(c) &&
                         (isprint(c) || c == '\n' ||
                             c == '\t' || c == ' '))
                         putchar(c) ;
                 else if (!strip)
                         octnum(c) ;
         }    /* while */
}    /* vis */
```

Continued on Page 73

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

10th Anniversary Celebration!

INTERNATIONAL HEATH/ZENITH USERS' GROUP CONFERENCE

O'Hare Hyatt Regency
Rosemont, Illinois
August 21, 22, 23, 1987

Name(s): ______________________________

____

____

Company: ______________________________

Address: ______________________________

City: ______________ **State:** ________ **Zip:** ________

Enclosed is $27.00 for each of the individuals listed above to attend the International HUG Conference being held the weekend of August 21, 22, and 23, 1987. Please send tickets along with information regarding hotel reservations and transportation.

Amt. Enclosed: ________ No. Attending: ________

For Our Information:

Which Heath/Zenith computer do you now operate? ______________

Are you a Non-User-Attendee?	Yes	No
Are you a computer related manufacturer?	Yes	No
If yes, would you like exhibit information?	Yes	No
Are you, or anyone in your party, interested in activities in or around the Chicago area other than the Conference?	Yes	No

If yes, please indicate any suggestions you may have:

____

Special Notice To Exhibitors:

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

For Your Information:

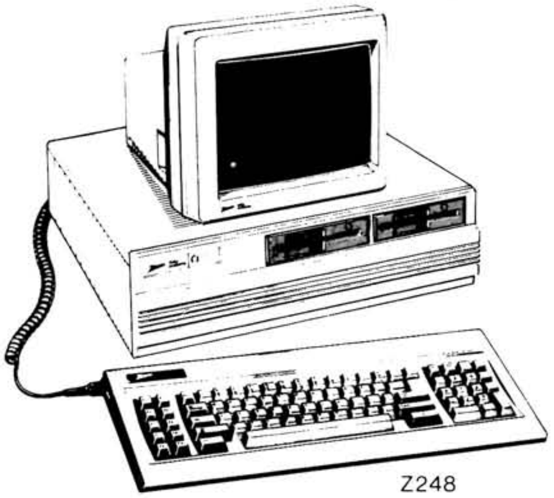
The $27.00 you are paying for your reservation to the International HUG Conference entitles you to all functions of the Conference. Visitor tickets, for those of you simply attending the seminars and exhibits, are available for $12.00. Visitor tickets do not include eligibility for prizes or food while attending the Conference.

Please send your completed registration form or suitable copy to:

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

Registration(s) must be post marked no later than July 31,1987. Cancellation will not be accepted after this date.
Sorry, We cannot accept purchase orders

First Capitol Delivers Zenith. Your Way.



Z248

Z181

Specialists in

CAD Systems
Local Area Networking

Z148

ZENITH | data systems

AUTHORIZED SALES AND SERVICE

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.

FCC *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.

The Best $25 Word Processor In The World

Richard Kelly
136 Sunrise Drive
Knoxville, TN 37919-4122

There is a silent revolution going on in the software industry. Some enterprising people are discovering the fact that powerful software programs can be designed and marketed for a substantial profit without gouging the end user. Word processors such as WordStar, MultiMate, and Microsoft Word typically have sold from anywhere between $200 to $600. The alternative to purchasing high-priced word processors has been 1) to make illegal copies of them, or 2) to acquire public domain or "shareware" programs. The problem with the first alternative is that it is illegal and immoral (despite the rationalization that the companies which produce these programs have already made their money or that they have charged too much in the first place). The problem with the second alternative is that such programs simply are not very good.

Recently, however, the public has been given a third alternative. Companies such as Ann Arbor Software (Textra), Knowledge Engineering (ZenWord), Vantex Data Systems (Vantex Word), and ButtonWare (PC-Type) sell their word processors for $25, $30, $49, and $60 respectively. While these programs do not include all of the desirable features of the more sophisticated (and costly) word processors, one of them, Textra 3.2, is quickly emerging as a serious challenge to the high-priced programs. It is, without question, the best $25 word processor around, and is on its way to becoming one of the fastest and best designed word processors at any price.

Ann Arbor Software is currently revising Textra and preparing a new manual. Previous owners of Textra 3.1 have received several updated disks at nominal cost. By the time you read this version 4 should be available. The version I am working with is called 3.2 Beta. It contains corrections of several bugs caught in previous versions, as well as many new and very valuable features.

Although Textra will run on practically any IBM compatible machine with 128K, it will run more efficiently with 256K, allowing you to create documents of about forty pages in length. Along with the program file comes a superb help file that can be accessed at any time during editing. You can find help through a screen that summarizes editing commands or through a screen that features table of contents. You can find help that is context sensitive. If, for example, you have selected a mode to highlight some text and you forget the correct procedure, when you call for help you will be given information directly related to highlighting text. Besides the help file, the disk contains an excellent tutorial using several "films," that is, graphic-like demonstrations of all the basic commands and their effects. These "films" can also be accessed during the editing process. Since they are fairly long, about five minutes each, as soon as you become experienced with the program you would probably want to remove these aids from your disk, keeping only the files Textra.exe and Help.txt.

Textra offers an amazing number of features. Paragraph formatting, margin controls, line spacing (1-3), page formatting, headers and footers, search and replace (forward/backward), on-screen underlining and bold, moving, copying and merging text, saving text in ASCII or Textra formats, highlighting text, user-defined fonts (for italics, superscripts, etc.) that can be easily inserted into the text with the Scroll Lock Key — are some of the program's notable features.

Word processors, however, do not survive on features alone. The wonderful thing about Textra is its sleek design, which makes the use of its powerful features quick and easy. The opening screen presents the following items: time and date, the logged directory, the name of the last document edited, and a list of all the files in the current drive. Also there is a list of seven function keys that will perform the following tasks: return to DOS, provide information about getting started with Textra (for beginners and forgetters), change the current disk drive, take you to a listing of directory commands, print a document, peek at a document, and create a new document. A new document can be created either by entering its name or number (from the numbered list of files in the directory) at a prompt line.

Textra affords a writer a clean, open screen in which to compose. There is a one-line menu that can be activated through function keys allowing you to reformat text (automatically, if you choose), highlight and delete text, call for help, call up another menu line with search/replace, copy, move, print, layout, merge, save/exit functions, and call up still another menu line with more exotic functions such as page preview, fine-tuning miscellaneous switches and reformatting commands. Textra provides all the necessary prompts you need to follow out any command. Outside of the single-line menu the only other item on the screen is optional but very handy: a notation of what page, line, and column you are on in your text. This can be placed either at the bottom or top of the screen.

When you save your text you are presented with a menu containing the following ten options, all activated by function keys: return to DOS, switches that control ways of saving documents, current mode of saving (ASCII or Textra format), retrieve new document, change current disk drive, directory commands, save document with same name, save with new name, save current document and resume editing, save a portion of the current document. This screen also contains the numbered list of files in the logged drive.

The directory commands screen allows you to turn on or off the display of the document size and how much free space remains on the disk. This screen also enables you to change directory, delete a document, or rename a document. This list of files in the logged drive are displayed thereby providing you with all the data you need to make your changes.

The speed with which you can page through your document is startling. Unlike WordStar with its slow overlay files, Textra allows you to move through your document at RAM speed. Reading in and saving text also take place very rapidly. It can read a 10K file into memory in about two seconds and save it in about four seconds. The slowest element in Textra are the help files but even then there is a solution. You have the option of starting up the program with the help files being read into your computer's RAM. Since the Help.txt file is 61K, it is advisable work with a computer with 256K RAM.

A highly desirable feature of Textra is that you can customize it to suit your needs. This is done through a supplied program file called Txcustom, which is very easy to use. Once you enter the program you can change many of the default values, such as those relating to search/replace, saving, reformatting, printing, and type of monitor. Some of the printer controls, for example, which are accessed through the editing mode, can be permanently changed through Txcustom. Line spacing, number of copies to print, pause between sheets, and insertion of time/date/filename are the print options. Under page formatting you can select the line per page, the top and bottom margins, and whether you want page numbering, headers, or footers. The printer commands include page preview (with word and line counts), print current document, print document from disk, print selected pages, print selected lines, and print letter + envelope. You would want to set up most of the print options and page formatting default values with Txcustom.

The ease and pleasure of using Textra derive from its coherent, efficient design, which I am confident, will assure its survival as the fittest among inexpensive word processors. Buttonware's PC-Type and Vantex Word, for example, are both crammed with features but both programs have serious design problems. PC-Type is best suited for someone who enjoys tinkering with default options and plowing through an array of commands and keystrokes. If you are simply interested in writing letters, articles, or books, PC-Type seems to do its best to get in your way. It lacks the superb clarity and transparency of Textra.

Vantex Word, on the other hand, has some interesting features despite its design problem. It comes with a good spelling check of 90,000 words and can be expanded to 200,000 words. (There will soon be a spelling check available for Textra). Some of its other features include unlimited document size, true split screen, 30 user-settable defaults, blinking and non-blinking cursor, automatic backup and save, macros (700 keystrokes per set), automatic paragraph reform, boldface, underline, superscript, subscript, italics, highlighted block move and copy, backwards and forwards search and replace, preview page breaks before printing, format control features (margins, spacing, etc.), integrated mail merge, print drivers for a few popular printers, redefinable keys, and context-sensitive help.

If you need a feature-packed word processor for a small amount of money ($49), Vantex Word is not a bad choice. The tradeoff, however, comes in its design. First of all, the manual is poorly written. Its chief problem is its failure to move you methodically and comprehensively through the program. Five brief tutorials introduce you to a few simple cursor movements, saving the document, making a block move and copy operation, and a few other basic elements. The manual puts you on your own before you have mastered a sufficient number of commands and options to be able to do any serious writing. The instructions are all clear enough; they simply are not well organized, especially for someone beginning word processing.

Although most operations in Vantex Word take place with great speed, some of the operations require more keystrokes than seem necessary. For example, to save a file and exit you must press four different keys: CTRL F1 and then CTRL Q. There is no template to explain the many keystroke combinations and the manual fails to provide a comprehensive list of these important details. The index does not even have an entry for "Save." Because the programs are so large Vantex Word also requires considerable disk manipulation for users with floppy drives. It is really designed for people with hard disks.

Although you can preview page breaks (and insert hard ones) before printing, they are not visible during the editing session. Similarly, boldface and underlining, double and triple spacing, and many other formatting features are not reproduced on the screen. You must move from your text to a formatting screen from which you select a "format object," and when you return to your text you will see your selection in written form within parentheses inserted at the cursor position. I find it rather cumbersome to have to leave the text screen in order to arrange it. A curious feature is that automatic paragraph reform takes place even when the screen shows a jumble of lines. When you save the text it is reformed on disk even when the screen shows chaos.

Once you get used to bouncing back and forth between the help screens and your text you begin to get the idea how to work your way through a document, but the knowledge does not come easily. Textra, on the other hand, has ingenious sign posts at the bottom of the screen that lead you without a manual or help screen through most details of basic text entry and editing. With Textra you need only consult the help screen when you encounter an obscure or little used feature.

Of the three inexpensive word processors I have worked with, my clear favorite is Textra. You cannot go wrong with it. My second choice would be Vantex Word. It is a powerful program with more features than you would imagine for the price. PC-Type I recommend only for those who enjoy word processing programs as ends in themselves.

Of the high priced word processors, there is no match for WordPerfect 4.1 (which I hope to review in the future). But if you are on a tight budget, get yourself a copy of the latest version of Textra (and its spelling check, when issued), and you can be assured of high quality, ease of operation, and downright fun.

HARDWARE
HEATH/ZENITH

SURPLUS - SALVAGE - ETC.
Most Circuit Boards for the Heath/Zenith Line, Z-100, Z-241, Z-158, Z-151, H-89, Z-171, Z-148. Cabinets, Disk Drives, Power Supplies, C.R.T.'s, Keyboards, Cables, H-DOS Operating System. $5.00 HTX-11 Keyboards $2.50 Ea. (NEW) 20 or More $1.00 Ea. (NEW) L.C.D. 171 $20.00 Ea. (NEW) 171AC Power Supply $25.00 Ea. (NEW) Modem 171 $100.00 Ea. Coaxial Cable (for Networking) 100 ft. Long $35.00 Ea. 25 ft. Long $20.00 Ea. Call or Write for a Price List - Continually Getting More Stock Each Month - Shipped U.P.S.-C.O.D.

SAVE THIS AD!

Self Addressed Stamped Envelope to:
Al Davis
4894 Lake Chapin Road
Berrien Springs, MI 49103
Phone: (616) 471-1792

CHEAP PRICES!

Continued from Page 66

```
        MOV     AX,WORD PTR INT_CY
        CMP     AX,LINES                ; end of screen?
        JNC     EXIT                    ; YEP, Let's get out
                                          of here
        JMP     NEAR PTR INT_SCRN1
EXIT:   RET

INT_END LABEL   NEAR

INSTALL:
        MOV     SP,OFFSET INT_STACK ; Good place for it
        PUSH    DS

        XOR     AX,AX
        MOV     DS,AX                   ; Clear DS

        MOV     SI,5*4              ; SI points to interrupt
        CLI

        MOV     WORD PTR [SI],OFFSET INT_5
        MOV     WORD PTR [SI+2],CS ; Set my interrupt
                                     vector

        STI
        POP     DS

        MOV     DX,OFFSET INT_END   ; LWA of resident
                                      portion
        INT     27H

CODE    ENDS
        END     BEGIN
```

Continued from Page 68

Listing 4 — "octnum.c" Function

```
/* prints a character in the range 0 - 255 as an octal
                                              number. */

octnum(c)
int     c ;
{ /* octnum */
        (unsigned int)c ;
        putchar('\\') ;  /* puts initial slash of
                                        octal format */
        putchar(c/64 + '0') ; /* initial digit is 0 - 3 */
        c %= 64 ; /* assignment operator */
        putchar(c/8 + '0') ;
        c %= 8 ;
        putchar(c + '0') ;
}    /* octnum */
```

Line 45 "#includes" the standard library supplied by The Software Toolworks. I have found it lacks many of the common functions that are frequently needed. However, the people at The Software Toolworks have made it very easy to add to the library. Listing 5 shows "fputs.c", "isascii.c", and "isprint.c". I just put each between "#ifneed" and "#endif" compiler directives and then inserted them into "stdlib.c". This causes no trouble because they do not call other functions in the library. "efopen .c" I did not insert into the standard library because it calls on "fputs" which is also in the library. I do not want to get the compiler confused with what is "if needed" so I avoid the opportunity for error.

As I promised, "view" is a useful program, gives additional insight into the use of the C/80 compiler, and provides some interesting functions to add to your library.

Listing 5 — Functions To Be Added To "stdlib.c"

```
/* fputs.c */

fputs(s,iop)
char    *s ;
register FILE   *iop ;
{       /* fputs */
        register   int  c ;

        while (c = *s++)
        putc(c, iop) ;
}       /* fputs */

------------------------------------------------

isascii(c)
char    c ;
{
        return((c < 128) ? 1 : 0) ;
}

------------------------------------------------

isprint(c)
char    c ;
{
        return((c >= ' ' && c <= '~') ? 1 : 0) ;
}
```

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
885-3031-37	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/Z100 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
885-8049-37	MSDOS Accounting System	20.00	85

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

MSDOS

Part Number	Description of Product	Selling Price	Vol. Issue
885-3026-37	MSDOS SMALL C Compiler	25.00	65

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
885-8047-37	CP/M Accounting System	20.00	85

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
885-8048-37	ZDOS/MSDOS Accounting System	20.00	85

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
885-8034-37	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
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

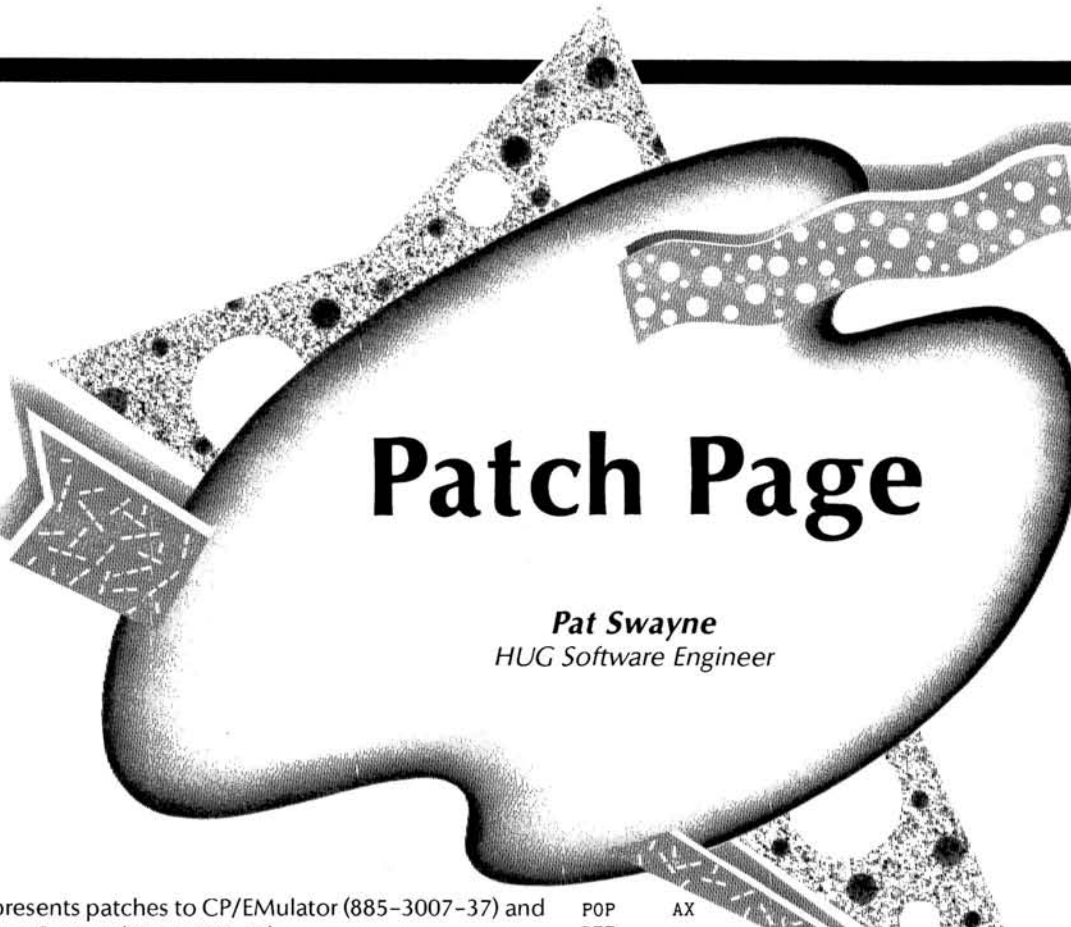
MSDOS H/Z100 - H/Z150 PC

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	

Continued on Page 83☛



Patch Page

Pat Swayne
HUG Software Engineer

This article presents patches to CP/EMulator (885-3007-37) and the HUG Menu System (885-3020-37).

CP/EMulator Patch

If you have the latest version of CP/EMulator (version 2.1), programs that send data to a printer running under it will not work properly if your CPM.COM file is dated earlier than 1-06-87. You can correct the problem by making a file called CPMPCH.DAT that contains these lines:

```
E131E
90 90
 0C
W
Q
```

Notice that there is a space before the "0C" in the lines above. Copy CPMPCH.DAT and DEBUG.COM to your CP/EMulator disk, log on to the disk, and enter

```
DEBUG CPM.COM <CPMPCH.DAT
```

and hit RETURN. When the DOS prompt reappears, your file will have been patched.

If you wish to correct the source code, locate these lines in the file CPM.ASM:

```
        ORG         4BH
BIOS_PRNFUNC        LABEL    FAR
```

Add two new lines above them like this:

```
        ORG         0CH
BIOS_PRINT          LABEL    FAR
```

Now, locate these lines:

```
PRTO:
        PUSH    AX
        MOV     AL,DL
        MOV     AH,CHR_WRITE
        CALL    BIOS_PRNFUNC
        POP     AX
        RET
```

and change them to:

```
PRTO:
        PUSH    AX
        MOV     AL,DL
        NOP
        NOP
        CALL    BIOS_PRINT
        POP     AX
        RET
```

Then reassemble the file to get CPM.COM.

HUG Menu System Patch

The HUG Menu System does not work correctly under MS-DOS version 3 because of a slight difference between that system and MS-DOS version 2. The problem affects the manual entry option. If you enter the name of a program to run, and do not enter a drive code because it is on the default drive, you will get a "Program load error" message. You can get around the problem by always specifying a drive code. If you have the Microsoft assembler (MASM), you can edit and reassemble MENU.ASM as follows. First, locate these lines:

```
          lodsb
          add      al,'@'
          stosb                 ; drive letter
```

Change them to look like this:

```
          lodsb
          OR       AL,AL        ; DEFAULT DRIVE?
          JNZ      NOTDEFD      ; NO
          MOV      AH,19H
          INT      21H          ; ELSE, GET DEFAULT DRIVE
          INC      AL           ; MAKE IT 1-BASED
NOTDEFD:add        al,'@'
          stosb                 ; drive letter
```

Then reassemble the file to get MENU.COM.

HUG NEW PRODUCTS

HUG P/N 885-6010-37

HAM HELP $20.00

Introduction: The program, HAM HELP, makes use of the personal computer to do a task that is, first, of potentially great use to the serious amateur radio operator and second, something that he could do only with great difficulty, if at all, without his computer. Accepting data that are available each hour throughout the day on the National Time Station, WWV, the program calculates the MUF (Maximum Useable Frequency) for the path between two geographical locations selected by the computer operator, for each 30 minute period of that day. Calculated results represented on the computer's video display terminal in a chart form. In addition to the MUF chart, the computer displays in tabular form such information as the great circle azimuth of the line connecting the two geographical points . . . the antenna azimuth . . ., an optimal antenna elevation, the path length, estimated radio signal attenuation over that path, and an estimate of the likely propagation conditions as a function of the existing electromagnetic environment. At the option of the computer operator, if the two locations are more than 4000 kilometers apart, the program will calculate the exact times of sunrise and sunset for each, will check for any unusual possibilities such as 'Grayline' longpath openings (defined below) or preferred paths to take advantage of or to avoid certain good or bad polar cap propagation phenomena.

Requirements: HAM HELP requires MS-DOS version 2.0 or greater on any Heath/Zenith PC compatible computer.

The following files are included on the HUG 885-6010-37 HAM HELP disk:

HAMHELP .EXE	FILEFIX .EXE
FOREIGN .LOC	HAMHELP .DOC
NOAMER .LOC	README .DOC

Author: Raymond S. Isenson (N6UE)

Program Content: To calculate the MUF, HAM HELP requires information as to the SF (Solar Flux) conditions for the most recent five days and the current geomagnetic value (the 'A' value.). Every hour, at 18 minutes after the hour, an announcer on station WWV (Boulder, CO) reports the SF for the day, the 'A' value and a current 'K' value. The latter is not used in this program. Obviously, the user must keep a record of the SF for a period of at least five days, including that of interest. The 'new' day for SF information purposes begins at 1800 hours, GMT. The SF tends to vary up and down on a short term cycle of about 28 days and a long term cycle of many years. By keeping long term records of the SF, the user can develop the potential to estimate what the SF will be for some future date and have the computer generate MUF curves on the basis of that estimate. (To use the program for this purpose, input the same SF for each of the five days. Input any number for the 'A' value. The estimated Quality Factor will be meaningless as it de-

Continued on Page 83☛

TABLE C
Product Rating

10 - Very Good
9 - Good
8 - Average

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

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

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

ORDERING INFORMATION

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

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

NOTE

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

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

BUGGIN' HUG

Hopes For Suggestions

Dear HUG:

I have a question that I would appreciate you publishing with hopes that there are some readers that have some suggestions.

I have a Z-100 and have written various menu driven programs in BASIC that prompt the user, during boot, on what printer options to set and so on. These options are set by sending the printer character strings, chr$(34), etc., through BASIC.

What I am having a hard time doing is writing these routines in 'C' since I have yet figured out how to write a 'C' string equivalent to a BASIC statement chr$(34) and the likes, nor have I figured out how to send these mysterious statements to my printer.

Sure would appreciate any suggestions out there on this topic. Thank you very much.

Sincerely,

Bradford R. Humphrey
340 South Street
Pittsfield, MA 01201

Clipper Under ZPC

Dear HUG:

I have been doing some work with the PC Emulator, ZPC, on my H-100 attempting to make more IBM PC software function. One program that can be used with the emulator is the DBASE III compiler CLIPPER. The compiler and linker will run as they are in the Z-100 mode. In fact, they work better in this mode. Once the programs have been compiled, they require four changes. Two of the changes are together at the same location. Once the program is patched, it will run in the PC mode.

To make the finished .EXE program work, first rename the program with a .BIN extension. Then run debug and read in the program.

```
DEBUG PROGRAM.BIN
```

Next search for BA DA

```
S0 FFF0 BA DA
```

The result of this command to debug should be one (1) address. Next use the unassemble command to unassemble starting at this address.

```
U address
```

The code will look like this:

```
BADA03
EC
A801
75FB
FA
EC
A801
74FB
8BC3
AB
```

Use the enter command to change this code starting at the 'EC' right after the 'BADA03'. This is three words after the address found in the search command. Replace:

```
EC
A801
75FB
```

with NOP which is 90. Replace:

```
EC
A801
74FB
```

with NOP.

Next, search for 'BA D8 03', for the search use a range of 0 to FFF0. There should be two addresses found. At the first address given use the assemble command to put in the following code:

```
A ADDRESS1
MOV AL,20
INT 51
NOP
NOP
```

Use the unassemble command to unassemble beginning at the second address returned by the previous search. The code should look like this:

```
BAD803
B029
EE
B403
```

Replace the 'EE' with an NOP (90) and that is the last change. Be sure to write the file back to the disk before exiting from DEBUG. Now change the name of the program back to an .EXE file and it is ready to run. The program needs to be on the same disk and partition as the database or other files used.

The programs are also very long. A DBASE III program that had only one executable line, an '@ SAY' command, was over 100,000 words long. Adding about 2000 lines of program code increases the size about 35,000 words to about 140,000. I have compiled 16 programs comprising about 4000 lines of program code, both separately and together with good results.

One thing that I have found with CLIPPER is that it does not like the DO CASE command structure. Programs having DO CASE in them have gotten many errors, most of them were not valid. When I replaced the DO CASE with a series of IF ENDIF statements then the program compiled correctly.

One other note. I attempted to change the compiler to compile programs that would run directly under ZPC. The code that needs to be changed is in the file CLIPPER.LIB. It is the same as found in the compiled program. The problem is that when the changes are made to the .LIB file, the linker (PLINK86) gets two checksum warnings and the .EXE file will not work. The file can be patched to work, but the changes required are not predictable so this method is more trouble than the first. If anyone finds a way around this, I would like to know about it.

Timothy Wilmarth
C/O TIG Foxboro Co.
P.O. Box 647
Dammam, Saudi Arabia

"Use It Or Lose It!"

Dear HUG:

Recently, I had corresponded with HUG while I renewed my HUG membership. I mentioned that I hoped that REMark would continue to provide future articles for CP/M-80 in addition to the MS-DOS articles.

As secretary, Margaret Bacon, pointed out to me in her return letter, those type articles are really dependent on the authors of REMark articles, and not just the HUG staff! Thus, I am following up with this letter to the authors via the Buggin' HUG column.

Basically, I would like to call to the attention of such authors the widespread and popular theory that there remains a large group of the computer community that continue to use CP/M-80. That, generally speaking, they either use "Plain Vanilla" CP/M and others using enhanced CP/M-80 (ZCPR3, and similar) systems.

Although I do not have supporting statistics. It is commonly voiced that this user group is large. Which is why I feel the CP/M and 8-bit groups should not only be recognized, but additionally should be provided article consideration by authors/editors.

As a matter of interest, many of the so-called "chippers and hackers" are enthused about recent advances in the 8-bit computer technology. New 8-bit enhanced processor chips, i.e., Hitachi HD64180, etc.; all the dynamic software, i.e., ZCPR3, etc., are providing the user community with ENHANCED systems that run with large managed RAM memory, fast running speeds, even high resolution color graphics, i.e., MicroMint's SB180 8-bit computer with HD64180.

As is the case with any language, CP/M included, if it is NOT USED it will DISAPPEAR in time. That is the point! If we wish to keep CP/M and 8-bit systems around, we must work at it. Surely there is enough room for both MS-DOS and CP/M.

Assuming favorable market economics, I would like to see some of the Heath/Zenith support vendors market a kit for updating the H-8, H/Z-89/90 systems with newer enhanced Z-80 type chips, i.e., HD64180.

Thanks for the recent articles in REMark on the ZCPR3 operating system. Hopefully such articles will make the Heath/Zenith community more aware of the facts as I have mentioned. So keep it up! There are many of us out here who appreciate it, and that are OPTIMISTIC about the future of CP/M-80 and ENHANCED 8-bit CP/M systems. When submitting articles for MS-DOS why not include CP/M as well?

Very truly,

Albert F. Bjorling, CMfgE
P.O. Box 216
Circleville, NY 10919-0216

Controlling Electronic Relays Through The Parallel Output Port

Dear HUG:

I have a hobby project whereby my HS-151 computer will control a large number of electronic relays through the parallel output port. This part of the program is done in assembly language to conserve memory space and function faster. On the video monitor is displayed graphics in BASICA and with a moving cursor will show just exactly which relays are operating.

I just can not get the two languages to communicate with each other. There will be over 100 constant exchanges back and forth between the two languages.

It would be nice if there was a program to merge GW-BASIC with the Assembly language program and create one file. The one file could then be saved on disk and could be called any time the file was needed. I haven't seen anything that comes close to this in REMark or any other magazine. What I have seen leaves much to be desired in completeness and clarity of articles and usually for another system. I dread the thought of doing my graphics in Assembly language.

Cordially yours,

Cletus J. Schneider
R1, Box 109
Vergennes, IL 62994

FULL PATH: 1 Solution For Many 'Fixes'

Dear HUG:

In reference to the article "Part 1: Getting Started With Hierarchial Directories" by Erik L. Pang in October 1986 Issue of REMark, HUG users will find an additional product, not mentioned at the end of the article regarding software products solving the problem of searching alternate directories whenever a file does not exist on the originally requested device or path. A product called FULL PATH operating as an extension to MS-DOS, gives ALL application programs the ability to search for data files in a similar fashion that DOS uses to search for COMMAND and BATCH files. All the fuss over using WordStar directory changers and other fixes to get around the problem of searching directories for data files is solved by including one extra command in your autoexec.bat file! Also, using commands SUBST and JOIN in PC-DOS 3.1 are a poor excuse for solving this problem.

FULL PATH can be purchased for a measly $24.95 from P. R. Glassel and Associates, Inc., 30255 Fir Trail, Stacy, MN 55079, (612) 462-1337. A 30-day satisfaction guarantee is also included.

Sincerely,

Jerry R. Schnetzer
8301 Bloomington Avenue South
Bloomington, MN 55420

DEBUG "Bug"

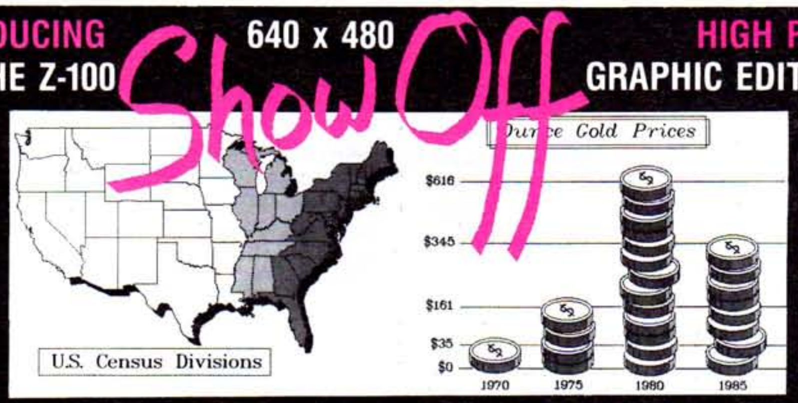
Dear HUG:

I have more information on the "bug" reported by Jerry Furst in his article, "DEBUG" (REMark, October 1986).

In his example of the use of the (s)earch command, Jerry pointed out that DEBUG will not accept the command "s000 ffff ec a8 01", which one would expect to search the entire default segment (the value of the DS register) for the consecutive bytes ec, a8, and 01. While it is true it will not accept this command, DEBUG will accept any of the following commands to search for the listed bytes:

Command	Comments
`s000 fffe ec a8 01`	Searches all but the last byte in default segment
`sXL0 ec a8 01`	Searches entire default segment starting at X (0-FFFF hex)
`sX Y ec a8 01`	Searches entire default segment starting at X if Y = X (0-FFFF hex)

INTRODUCING **FOR THE Z-100** 640 x 480 HIGH RES **GRAPHIC EDITOR**



Now, you can improve your presentations and reports. Use ShowOff, THE hi-res, graphics program for the Z-100. Easy to use, ShowOff will help you draw, paint and add text to create outstanding graphics.

ShowOff minimum requirements: Z-100 • color video memory • 384K RAM • MS-DOS 2.0 or higher

ShowOff $79 • demo disk $3 • ShowOff with Logitech Mouse $169
also available for digitizing pads

HOGWARE COMPANY
470 BELLEVIEW • ST. LOUIS, MO 63119
(314) 962-7833

With ShowOff, your Z-100 will outperform all popular PC graphic systems.
ShowOff offers:
- 640 x 480 resolution
- 92 fill colors
- 92 patterns
- 25 text styles
- compatibility with other graphic programs

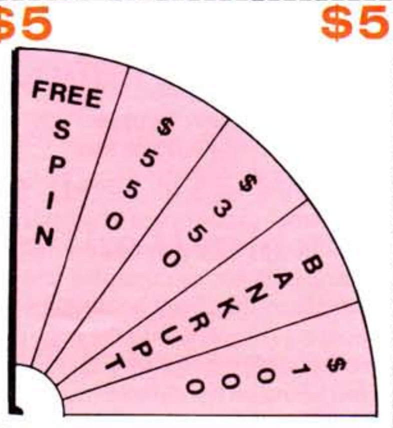
ORDER DIRECT • VISA • MASTERCARD • CHECK • Dealer Inquiries Welcome

From a practical standpoint, the first command is adequate to search the entire default segment. How often do you search for less than two bytes? If you were searching for the bytes ec and a8 and they happened to be the last two bytes of the segment, you would find them with the command "s0 fffe ec a8". This is so because fffe is the last byte searched for the first byte in the list. Note that I left out the trailing zeros in the starting address. One zero does the job.

The format for the second command is documented in the definition of "range" in Chapter 15 (DEBUG) of the Zenith Data Systems manual for MS-DOS 2.0. The L indicates that the hex number that follows is the number of bytes to be searched, including the byte corresponding to the start address. The number cannot exceed four digits (16-bits). Since the number of bytes in a segment is 10000 hex and you would never search zero bytes, 0000, 000, and 0 are interpreted as 10000 hex. Not documented is the fact that the entire segment is searched regardless of the start address, X. The only difference is where the search starts and ends. For example, the command "sFFFFL0 ec a8 01" looks at bytes in the following order: FFFF, 0000, 0001 . . . FFFE. There really is no reason

SPECIAL INTRODUCTORY OFFER!

$5 $5



SPIN-FOR-MONEY

Just like the popular TV-game!
List Price $24.95
WITH COUPON
ONLY $19.95
(Does not include Vanna White!)

Versions available for H89, Z100, and Zenith PCs.

$5 $5

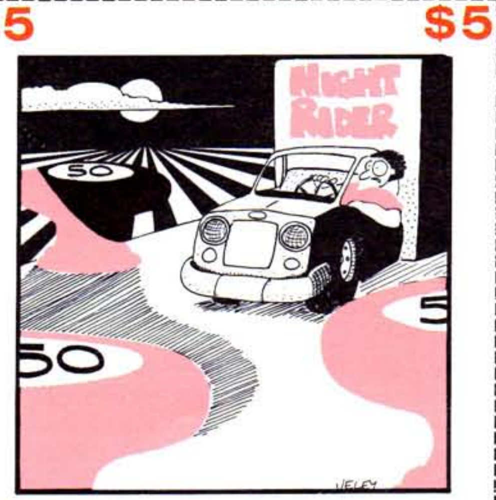
Add $3.00 per order for shipping and handling.
Michigan residents add 4% sales tax.
C.O.D. shipments available for $2.00 additional.
Specify computer model and operating system when ordering.

VISA **For faster delivery, call** master charge
906-249-9801
DEPT. 37R, POB 790
MARQUETTE, MI 49855

G.C.P.I.
GENERIC COMPUTER PRODUCTS, INC.
COMPUTER PRODUCTS YOU CAN RELY ON

CLIP 'n SAVE

$5 $5



NIGHT-RIDER PINBALL

So real . . . you'll forget you're using a computer!
LIST PRICE $29.95
WITH COUPON ***ONLY $24.95***
Requires Zenith-PC and Color Monitor

$5 $5

to use a value of X other than zero, as in the command "sOLO ec a8 01".

I discovered that the third command works by accident. Since there never is a need to search for only one byte, setting Y = X is the way you indicate you want to search an entire segment in the start address, space, end address, format. The entire segment is searched regardless of the value of X and Y, as long as Y = X. The affect of different values of X is the same as for the second command. Again, there is no reason to use a value of X and Y other than zero, as in the command "s0 0 ec a8 01".

In summary, the entire default segment can be searched for bytes listed with the command "sOLO list" or the command "s0 0 list". Any other segment can be searched with the command "sSEG:0LO list" or the command "sSEG:0 0 list". Take your pick.

One other point. If the list of bytes you are looking for ends beyond the current segment, it won't be found if the next segment you search is the current segment + 1000 hex. For example, assume the current segment is 1A90, as in Jerry's example. If the list ec a8 01 occurs starting at FFFE, it will not be found when segment 1A90 is searched since the last byte is not in segment 1A90. If the next segment searched is 2A90 (1A90 + 1000) the list will not be found since the first two bytes are not in segment 2A90. The solution is to overlap the searches. The list would be found if segment 2A8F (1A90 + FFF) were searched. DEBUG will not accept a command line longer than 80 bytes. If you always add FFB (4091 decimal) to the last segment searched, you will never miss the list you are searching for since the overlap is 80 bytes (5 × 16). You can use the DEBUG (h)ex command to do the necessary arithmetic.

Sincerely,

John E. Hill
1117 Haral Place
Cherry Hill, NJ 08034

"1.2 Megabyte Drives For The H/Z-100"

Dear Pat:

I am a Z-100 owner, and have followed your articles in REMark since '83. Your CRTSAVER program is still in my AUTOEXEC.BAT, and ZPC goes a long way toward easing one of my biggest frustrations as a Z-100 owner, the lack of IBM PC compatibility.

I am writing in regard to your article in the June Issue of REMark, "1.2 Megabyte Drives For The ET-100". Another frustration of being a Z-100 owner is Zenith's pricing policy, which makes upgrading the Z-100 with Zenith hardware or software economically unfeasible for me in most cases. For example, I have long wanted to add a hard disk to my system, but I refuse to pay $1000 for an obsolete 10M system. And the Zenith 8" drive lists for $800/$1400 for one/two drive units in kit form. On the other hand, 5-1/4" 1.2M TEAC drives (AT style Model FD-55GFV) can be purchased around here for as low as $115. I believe that you could provide a real service to your readers if you would write an article on interfacing 1.2M drives to the Z-100. I have to believe that there are a whole lot more Z-100 users than ET-100 users, and dissemination of this kind of information is exactly what a users' group is all about.

I have just completed interfacing two 1.2M TEAC drives to my Z-100 and they work great, so I don't have a need myself for such an article, but I feel strongly that this is the kind of information that should be provided to HUG members. I suspect that you know a lot more about the subject than I do, but I will give you my experience on this project. If for some reason you cannot or are not allowed to write an article as I suggest, you may be able to pass this on to your readership.

The Z-100 is designed to handle 8" DSDD drives which have a capacity of 1.2M. The TEAC FD-55G drive is designed so that it can be configured (via jumpers) to look like an 8" DSDD drive. The reason you can't connect the TEAC directly to the floppy disk controller on the Z-100 is that the connector on the controller is a 50-pin connector while the TEAC has a 34-pin connector. Therefore, to interface the TEAC to the Z-100 you must connect the appropriate pins on the 50-pin connector on the Z-100 floppy controller, For both connectors, all odd numbered pins are ground. The connections for the even number pins for the two connectors are given below:

50-Pin Connector (P50)

Pin	Connection	Pin	Connection
2	No Connection	28	Pin 12 of P34
4	No Connection	30	Pin 14 of P34
6	No Connection	32	Pin 6 of P34
8	No Connection	34	Pin 18 of P34
10	Ground	36	Pin 20 of P34
12	No Connection	38	Pin 22 of P34
14	Pin 32 of P34	40	Pin 24 of P34
16	No Connection	42	Pin 26 of P34
18	Pin 4 of P34	44	Pin 28 of P34
20	Pin 8 of P34	46	Pin 30 of P34
22	Pin 34 of P34	48	No Connection
24	No Connection	50	No Connection
26	Pin 10 of P34		

34-Pin Connector (P34)

Pin	Connection	Pin	Connection
2	No Connection	20	Pin 36 of P50
4	Pin 18 of P50	22	Pin 38 of P50
6	Pin 32 of P50	24	Pin 40 of P50
8	Pin 20 of P50	26	Pin 42 of P50
10	Pin 26 of P50	28	Pin 44 of P50
12	Pin 28 of P50	30	Pin 46 of P50
14	Pin 30 of P50	32	Pin 14 of P50
16	No Connection	34	Pin 22 of P50
18	Pin 34 of P50		

The method I used was to build a small circuit board which has two connectors on it, a 50-pin and a 34-pin. One side of a normal 50-conductor cable plugs into the Z-100 floppy controller and the other side plugs into the connector on the circuit board. Similarly, one side of a normal 34-conductor cable plugs into the edge connectors of the TEACs and the other side plugs into the 34-pin connector on the circuit board. The wiring on the circuit board makes the connections described above.

The relative positions of the jumpers or straps on the circuit board of the TEAC are shown below:

```
X-X
FG
                                ML  RE  DC  RY
                 X  X  X  X     X-X X-X X-X X-X
         X X     ¦  ¦  ¦  ¦
       HG¦ ¦I    X  X  X  X
         X X     U1 U2 HL IU
       LG¦ ¦II
         X X       X  X  X  X
                   ¦  ¦  ¦  ¦
                   X  X  X  X
                  DS DS DS DS
                   Ø  1  2  3
```

The straps needed are FG,HG,II,HL,ML,RY, and DS0/DS1 for drive C/D respectively. The drives have a resistor pack installed in an IC socket on the circuit board. If two drives are to be used, only the last drive on the cable should have the resistor pack installed. I have no expansion boards installed in my Z-100, therefore, I am using power from the Z-100 power supply to power the TEAC drives.

I hope you will consider expanding this information with your own knowledge on the subject and giving your readership the opportunity to add affordable memory storage to their Z-100s.

Sincerely,

Ken Goto
1306 Bottle Brush Lane
San Jose, CA 95118

H-150 Speed-up Mod

Dear HUG:

In the December '86 Issue of REMark a question was posed concerning my H-150 speed-up modification and some possible problems the answers to which may be of interest to many. In particular, error messages when using the COPY command with floppy disks at 6.67MHz were observed. This problem arises when the /V switch (for verify) or when the MS-DOS command VERIFY ON is used. The same problem surfaces when either FORMAT or DISKCOPY is used at high speed (FORMAT and DISKCOPY verify automatically). It should also be mentioned that this problem is present no matter whether my modification is being used or some other (or for that matter whether it is an H/Z-150 or IBM PC). If the CPU clock is increased, the problem occurs. While I do not know exactly why it happens, I do know it only occurs when the BIOS disk service is used to verify a sector. There are a couple of programs floating around on bulletin boards that get around the problem by trapping the BIOS call and changing it from verify sector to read sector. FORMAT and DISKCOPY will then work at high speed but the verify function is disabled.

The second part of the question concerned the use of a 5MHz 8087 at higher clock speeds. The long term reliability of a semiconductor device is related to its operating temperature and as such components are derated as their operating temperature increases. Anyone who owns an 8087 knows that they run VERY warm (usually too hot to touch) and a heat sink certainly would not hurt the overall reliability of the IC even at normal speeds. If you are fortunate enough to have a 5MHz 8087 that will perform properly at higher speeds I can see little reason not to use it. As far as documented effects of running at a higher speed, only Intel knows for sure. A particular 8087 is not rated at 5MHz because it will melt at higher clock speeds but rather because it will not meet the 8MHz specification. Attempting to predict the actual effect (if any) of running at a clock speed greater than 4.77 MHz would be at best speculation on my part.

Dante Bencivengo
P.O. Box 234
Wyandotte, MI 48192

Hook Up An OKIDATA Printer To The H11

Dear HUG:

A couple of years ago, I sent this information to you but you did not print it. Now I see that you have printed a letter asking for this kind of information. (REMark, December 1986) The question was how to hook up an OKIDATA printer to the H11. I have not hooked up an OKIDATA, but I have hooked up an Epson FX-80, with the optional FX-80 serial interface board installed, and I think the connections may be the same, or only slightly different. Anyway, this information may give a needed clue. It was not easy to figure out how to hook up the FX-80 to the H11. I had to disassemble part of the LP.SYS driver by hand to figure it out. I found the place in LP.SYS where the hand-shaking is done. It appeared to have been patched quite a lot — probably by Heath, to modify DEC's LP handler to accommodate the H14. I figured out a way to repatch it to make it work with the FX-80. Use the H11 software patch facility, and select BLOCK NUMBER 00001 of LP.SYS. Words 026, 030, 032, 072, 102, and 104 must be changed in BLOCK NUMBER 00001. The new contents are 000240, 000240, 000240, 000240, 100375, and 000411, respectively. It is advisable to make a backup copy of LP.SYS before attempting the patching, of course, unless one is well experienced in making patches. The JNOR-JREV jumper on the FX-80 serial board is set to JNOR for normal polarity of the handshake signal, and the following cable connections are made from computer (the WH11-51 adapter) to the 25-pin RS232 connector on the printer.

```
     Computer Pin
3      4     7     1
3     11     7     1
      Printer Pin
```

Our H11 still works fine, but I don't use it much any more because I am constantly zapping the system disk by forgetting to remove it on power up or power down. I would be interested in replacing those old 8" drives with nice new 5" 800KB drives, but I suspect that it would be quite a project, what with the drive controller being partly hardware and partly ROM. Also, I would dread digging into the software modifications, not having any of the source code. If some quaint hacker among the REMark readers has made such a conversion, I would like to hear about it.

Sincerely,

Phillip L. Emerson
Department of Psychology
Cleveland State University
Cleveland, OH 44115

Programs For MPI Printers?

Dear HUG:

I am using a H/Z-100 computer with MPI 150 and MPI 99 printers. I have the AP-PAK for the H/Z-100 from MPI, but I never have figured out how to really use it to change font styles. I have seen a program for IBM called Stylewriter. If anyone knows where I can get a copy of this for the H/Z-100 computer or any other programs for MPI printers, please let me know.

Sincerely,

Michael L. Dettmer
95 Duquesne Drive
Napoleon, OH 43545

H-8, H/Z-88/89/90 Newsletters

Dear HUG:

I would like to pass some important information along to the 8-bit side of the membership. This concerns a pair of newsletters spe-

ZENITH | data systems

COMPUTERS

RAM Technology carries a full line of Zenith computers, peripherals, and software. Call us toll-free for competitive prices on systems and accessories.

VIDEO

Paradise Auto Switch EGA	$429.00
Video 7 EGA Deluxe	$429.00
Video 7 EGA High-Resolution Color Card	$369.00
Video 7 MGA High Res. Monochrome Graphics	$189.00
NEC Multisync High Res. Color Monitor	$599.00
C. ITOH CM1000 Color Monitor	$399.00
Mitsubishi EGA High Res. Color Monitor	$429.00
Premier Technologies Z148 1½-Slot Board	$109.00
ZVM-1220-A: Non-Glare, Amber Monitor	$ 99.00
ZVM-1230-A: Non-Glare, Green Monitor	$ 99.00
ZVM-1240: Non-Glare, Amber Monitor, TTL	$159.00
ZVM-1200-1: Tilt Swivel Base	$ 17.00
ZVM-1330: High-Resolution Color, RGB	$459.00
ZVM-1360: High-Resolution Color, RGB, Long-Persistence Phosphor	$609.00

DYNAMIC RAM MEMORY

4164	64K 150 NS.DRAM	QTY: 1-50:	$1.28
41256	256K 150 NS.DRAM	QTY: 1-50:	$2.90
41256	256K 120 NS. DRAM	QTY: 1-50:	$3.25

MODEMS

EVERCOM INTERNAL MODEM, 300/1200 Baud	$153.00
US ROBOTICS SPORTSTER 1200 (EXT)	$136.00
US ROBOTICS PASSWORD 300/1200	$199.00
US ROBOTICS MICROLINK 1200 W/Telpac	$239.00
US ROBOTICS MICROLINK 2400 W/Telpac	$389.00
US ROBOTICS COURIER 1200	$239.00
US ROBOTICS COURIER 2400	$389.00
US ROBOTICS VAR MODEM 1200 (INT.)	$136.00
US ROBOTICS VAR MODEM 2400 (INT.)	$198.00
US ROBOTICS COURIER 2400e	$439.00
US ROBOTICS COURIER HST (9600 baud)	$699.00

PC COMPATIBLE UPGRADE ACCESSORIES

INTEL ABOVE BOARD PC, 64K Expandable to 2 MB	$299.00
EVEREX MAGICCARD	$159.00
NEC V20 8 MHz Processor	$ 14.95
INTEL 8087-3 5 MHz. Co-Processor	$109.00
INTEL 8087-2 8 MHz. Co-Processor	$169.00
SMARTWATCH Clock/Cal. Module	$ 44.00
PRINTER CABLES, STANDARD PARALLEL, 6 Ft.	$ 22.00
SWITCHBOXES: 2 or 4 Position, 5 Yr. Warranty	$ 99.00
AST RAMPAGE EMS/EEMS w/256k, Software	$337.00
AST ADVANTAGE EMS/EEMS w/128k Serial, Parallel Ports, Software	$398.00

Z100 UPGRADE ACCESSORIES

C.D.R. Z100 Speed Module	$ 39.95
FBE RESEARCH ZMF-100A: Install Up To 768K On Z100 Motherboards #181-4917 Or Lower	$ 59.95
UCI Ramboard for Z-100, OK Expandable to 2MB	$ 299.00
SEAGATE 10 MB Winchester Kit, COMPLETE W/Zenith Contr., Hardware & Software	$ 999.00
W/CDR Contr., Hardware & Software	$ 899.00
SEAGATE 20 MB Winchester Kit, COMPLETE W/Zenith Contr., Hardware & Software	$1150.00
W/CDR Contr., Hardware & Software	$ 999.00
SYQUEST 10MB Removeable Media HD w/CDR Contr., Hardware, Software, and One Cartridge	$1248.00

PRINTERS

EPSON DX10 Daisy Wheel	$ 189.00
CANON LBP-8A1 Laser Printer	$2290.00
CANON A40 NLQ Dot Matrix, 140 cps, QUIET	$ 279.00
JUKI 6100 Daisy Wheel, 18 cps, parallel	$ 399.00
JUKI 6300 Daisy Wheel, 40 cps, ser/par	$ 749.00
OKIDATA ML182 Dot Matrix, 120 cps	$ 269.00
OKIDATA ML192 Dot Matrix, 160 cps	$ 399.00
TOSHIBA P321 3-in-1 par., 216 cps	$ 549.00
TOSHIBA P341 3-in-1 par., wide carriage	$ 899.00
C. ITOH Pro Writer Jr.	$ 259.00

Z200 UPGRADE ACCESSORIES

INTEL ABOVE BOARD AT, 128K, Expandable to 2 MB	$459.00
INTEL ABOVE BOARD AT PIGGYBACK, 128K; Expandable to 2 MB; Expands AB AT to 4 MB	$229.00
INTEL 80287 6 MHz. Co-Processor	$195.00
INTEL 80287 8 MHz. Co-Processor	$318.00
INTEL 80287 10 MHz. Co-Processor	$397.00
SEAGATE ST4026 20MB, 40 ms. avg.	$599.00
SEAGATE ST4038 30MB, 40 ms. avg.	$769.00
SEAGATE ST4051 40MB, 40 ms. avg.	$899.00
NEWBURY NDR 1085 85 MB, 30 ms. avg	$1799.00

Nashua Box of 10 With Lifetime Guarantee

Nashua Diskettes: DS/DD, 48 TPI, Soft-Sectored or Hard-Sectored (Specify)	$9⁵⁰
Nashua Diskettes: DS/DD, 96 TPI, Soft-Sectored or Hard-Sectored (Specify)	$21.50
Nashua Diskettes: DS/HD, SS 600 Oersted Media for AT and Compatible 1.2 MB Drives.	$29.95

HARD DISK DRIVES

With One-Year Replacement Warranty

SEAGATE 10 MB Hard Disk With OMTI	$399.00
SEAGATE ST 225 20MB Hard Disk	$339.00
SEAGATE 20 MB HD w/OMTI	$459.00
OMTI/LAPINES 20 MB Hardcard	$479.00
SEAGATE ST238 30 MB w/OMTI RLL Controller	$559.00
SEAGATE ST251 50 MB Half HT HD w/OMTI Controller	$938.00
SYQUEST 10 MB Removeable Media Hard Disk Drive With OMTI Controller, Mounting Hardware, Cables and One Cartridge	$759.00
Additional Cartridges	$109.00

MITSUBISHI DISK DRIVES

FUJITSU M2551A: 5¼, Halt Ht, 48tpi, DS/DD	$112.50
MF501A: 5¼" HalfHeight, 48 TPI, DS/DD	$125.00
MF504A: 5¼" Half Height, 96 TPI, 1.2 MB	$159.00
M2896: 8" Half Height, 48 TPI, DS/DD, Shugart 850/860 Compatible (Limited Quantities)	$199.00

TERMS

Prices and specifications subject to change. Minimum 90-day warranty on all hardware. Personal checks held 10 working days; money orders accepted as cash. Please add 2% (minimum $2.00) for shipping; any surplus will be refunded. COD orders accepted; cash or cashier's check only. VISA/MC accepted; please add 2.8%. Purchase Orders accepted from businesses and educational institutions. User group inquiries invited. We appreciate your business!.

RAM Technology

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

ORDER LINE: 1-800-662-0070
INFORMATION: (603) 889-0633

HOURS OF OPERATION: Monday-Friday, 9:00 A.M. - 5:00 p.m. EST

MasterCard

VISA

cifically oriented toward the H-8 and H/Z-88/89/90. Their vital statistics are as follows:

SEBHC Journal
c/o Lenny Geisler, Editor
895 Starwick Drive
Ann Arbor, MI 48105
Rate of Appearance: Monthly
Annual Subscription: $12.50
Size of December Issue: 18 pages
Estimated Subscription Base: 50+

The Staunch 8/89'er
c/o Hank Lotz, Editor
2024 Sampson Street
Pittsburgh, PA 15221
Rate of Appearance: Quarterly
Annual Subscription: $5.00
Size of December Issue: 8 pages
Estimated Subscription Base: 90+

If you anticipate hanging on to your 8-bit Heath/Zenith machine for some time, as I do, I strongly urge you to subscribe to one or both. Unfortunately, with the increasing dominance of the IBM-compatible even in our hardware-specific magazines, the currently-available information resources about our equipment are fast dwindling.

However, it is also my contention that more articles would appear here and in Sextant if more users would write about their machines. The magazines and newsletters can't publish anything about the '8 and '89 if no one submits anything! So support for Heath/Zenith's 8-bit equipment, as with any other discontinued model, comes down to the willingness of the owner (you and me) to contribute to the national publications. I'm doing my part; how about you?

Go to it!

Sincerely,

Kirk L. Thompson
#6 West Branch Mobile Home Village
West Branch, IA 52358

Continued from Page 69

pends upon 'A'.) The program will accept values for solar flux that vary between a low of 60 and a high of 400. The 'A' value could vary between 1 and 100, but will likely be in the range 1 to 20. To try the program, key in values of 150 for the SF for each day and a value of 6 for 'A'.

Listed at the right edge of the chart, as the last item, is a relative figure of merit, 'Estimated Propagation Quality'. This estimate is based upon many factors; the 'A', whether or not the path crosses the equator, the zenith distance of the midpoint of the path, and some proportionality constants, to name a few. Although of little use to the operator, initially, its value will grow with experience. You will find, for example, that QRN will be higher and there will be more signal flutter with lower 'Quality Estimates'. Therefore, if you learn through experience that a quality of 4 and a 'Q3' contact with London went together, you have reason to expect that the next time the program estimates a 4 for the path to London, you will have the same results; a 3, not so good, a 5, perhaps a 'Q5' contact. To another DX station an estimate of 6 might mean only a fair contact. In general, however, for two different paths at any given time, that with the higher quality estimate should offer much easier copy. The indicated 'Path Attenuation' also varies from day to day and from path to path. Its basis is somewhat different from that of the Quality Estimate. The two should be considered jointly in determining when to try for a specific DX or what antenna azimuth and when you should get a good response to a 'CQ'. Remember, the closer your operating frequency is to the MUF, the better will be your signal propagation and the more valid will be the information in the table.

All of the results of the calculations are presented on the video display tube. The program supports a hard copy printout that has somewhat less information than shown on the CRT but does include the MUF curve, beam azimuth, station location identification and date. Because we currently are near the low end of the "11 year" cycle and MUF will seldom exceed 20 to 25 MHz, a scale factor was chosen that limits the ordinate to less than 38 MHz.

Comments: none

TABLE C Rating: 10

885-3026-37 Small-C Compiler Update

Originally released in the June 1985 issue of REMark, the Small-C compiler, P/N 885-3026-37, has been updated by the author. A third disk has been added to the original two. This third disk contains the accessory packages. The first package is the Small-C Standard Library in both source (C) and object (OBJ) formats. Some of the files in the Standard Library include, ABS.C, ATOI.C, FGETC.C, RAND.C etc. The primary documentation, LIBRARY .DOC, is also included. This package is a collection of 30 subroutines which perform "grunt work" for C programs. These functions are a subset of the ANSI Standard Library Set as defined in the DRAFT Proposed ANSI Standard for the C Programming Language. The Utilities Set package consists of 16 programs which provide UNIX-like software tools. Most of the tools are based on the utilities in the book Software Tools by Kernighan and Plauger. Overall documentation is included for each of the programs. This new three disk set is now $30.00 and previous owners can obtain this three disk package by returning their original two disks, along with a check for $5.00 made out to HUG (NO credit card orders for this update), to Nancy Strunk, HUG, Hilltop Road, Saint Joseph, MI 49085.

Continued from Page 74

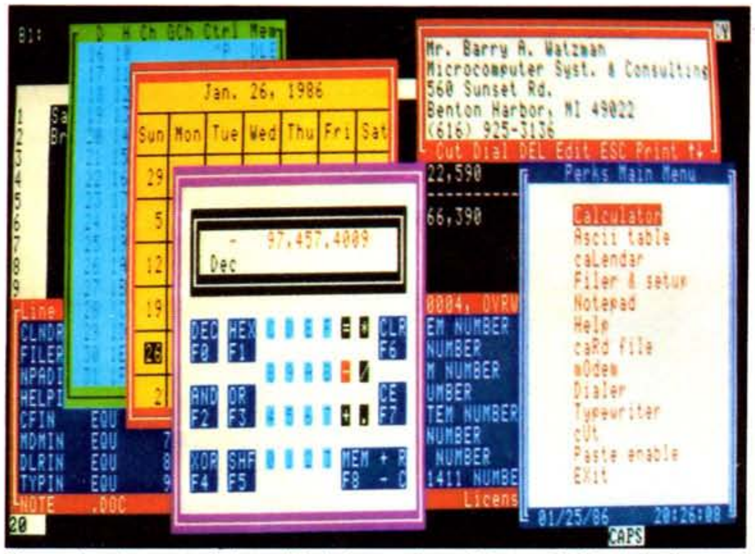
Part Number	Description of Product	Selling Price	Vol. Issue
885-4600	Watzman/HUG ROM	45.00	41
885-3015-37	ZDOS Skyviews	20.00	55
885-3036-37	MSDOS TREE-ID	20.00	77

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.

Announcing

Perks™-PC

Now the Perks desktop utility is available for both the Z-100 series and the Z-100 PC series computers as well as compatible computer systems. Perks-PC includes all Perks version 2 features with command and screen formats identical to Z-100 Perks version 2.



Mr. Barry A. Watzman
Microcomputer Syst. & Consulting
560 Sunset Rd.
Benton Harbor, MI 49022
(616) 925-3136

Shown above is an actual photo of Perks in operation, with Lotus 1-2-3 in the background. The main menu and windows for the notepad, calculator, ASCII table, calendar and card file modules are visible.

- ☑ Calculator
- ☑ ASCII Table
- ☑ Notepad w/Variable size buffer (to 64K)
- ☑ Perpetual Calendar
- ☑ Appointment Calendar
- ☑ Context Sensitive Help
- ☑ Cut/Paste
- ☑ Dialer
- ☑ Typewriter
- ☑ DOS Functions
- ☑ Alarm timers (8)
- ☑ Screen Saver
- ☑ Import/Export
- ☑ Removable w/o rebooting
- ☑ Runs with MS-DOS versions 1, 2 & 3
- ☑ Modem w/XMODEM file Xfer
- ☑ NOT copy protected
- ☑ Multi-Key ISAM Card File
- ☑ Small Size (under 80K)

Both Perks and Perks-PC are priced at $69.95 each.* Telephone credit card orders (VISA and MasterCard) are now accepted without a prior signed license agreement.

BARRY A. WATZMAN Microcomputer Systems & Consulting

560 Sunset Road • Benton Harbor, Michigan 49022-7142 • (616) 925-3136

Perks is a trademark of Barry Watzman.

*Plus $4.00 for S&H, Michigan Residents add 4% sales tax also.

Heath/Zenith Users' Group

Hilltop Road
Saint Joseph, Michigan 49085

BULK RATE
U.S. Postage
PAID
Heath Users' Group

POSTMASTER: If undeliverable, please do not return.