REMark®

Volume 9, Issue 4 • April 1988

P/N 885-2099 Issue 99

$2.50

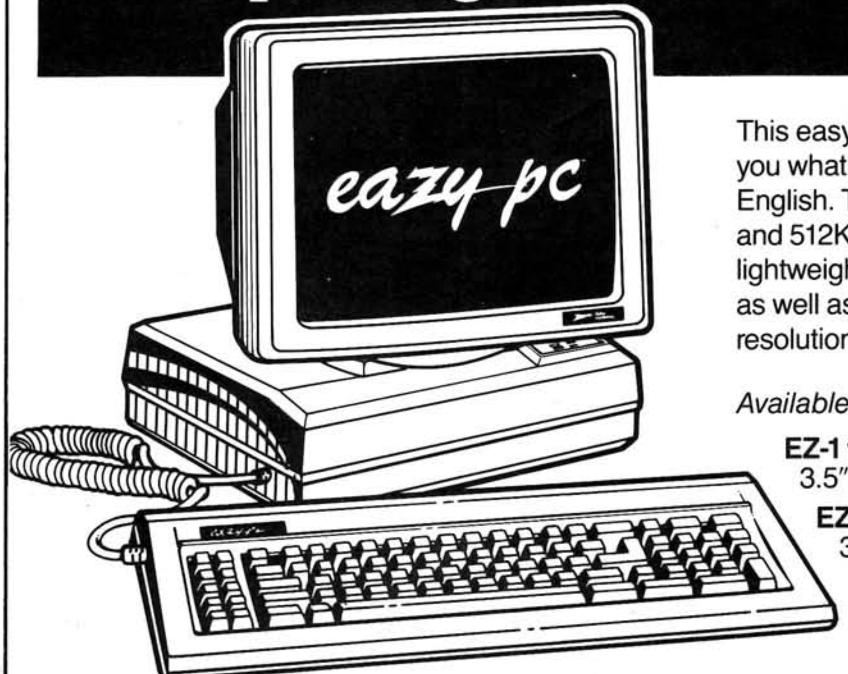
Planning A Beginner's Column
See Page 7

A Publication of Heath/Zenith

"Why Haven't You Called?"

"I've stayed up 24 hours a day waiting to hear from you. I know that for most of you, this may be your first time, but I promise, I'm very gentle! If you're still unsure, let me tell you what I have to offer. First, there's my message base. Here's where my users exchange ideas, receive assistance, and sell things. Next, is my database. Most of my users will tell you, it's second to none! With over 30 megabytes of hand-picked software it caters to all Heath/Zenith computers, but mostly PC compatibles. Finally, I have something new for you, my 'Bargain Centre'. Here you can buy surplus software and hardware, at unheard of prices. Interested? I hope so! Set your modem to either 300, 1200, or 2400 baud, and call me right now at (616) 982-3956, and register today. If you're still a bit shy, you can still call my human at (616) 982-3837 and register with him. Although he talks at 150 baud, he's gentle, too!"

Zenith Data Systems eaZy pc
Computing has never been easier



This easy-to-use IBM PC compatible tells you what to do and when – on-screen in plain English. The EaZy PC comes with MS-DOS and 512K RAM expandable to 640K. Plus it's lightweight and has a professional keyboard, as well as a 14″ monitor that provides high resolution text and graphics.

Available in three models:

EZ-1 with one 720K 3.5″ drive	ONLY **$799**
EZ-2 with dual 720K 3.5″ drives	ONLY **$959**
EZ-3 with 20 MB Winchester	ONLY **$1359**

Also comes with our FREE Partnership Pack support package worth $500 in savings!

HUG DISCOUNTS
DO NOT APPLY.

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

ARIZONA – Phoenix, 85017
2727 W. Indian School Rd.
602-279-6247

Tucson, 85711
5616 E. Broadway
602-745-0744

CALIFORNIA – Anaheim, 92805
330 E. Ball Rd.
714-776-9420

San Jose (Campbell), 95008
2350 S. Bascom Ave.
408-377-8920

El Cerrito, 94530
6000 Potrero Ave.
415-236-8870

San Diego (La Mesa), 92041
8363 Center Dr.
619-461-0110

Los Angeles, 90007
2309 S. Flower St.
213-749-0261

Pomona, 91767
1555 N. Orange Grove Ave.
714-623-3543

Redwood City, 94063
2001 Middlefield Rd.
415-365-8155

Sacramento, 95825
1860 Fulton Ave.
916-486-1575

Woodland Hills, 91364
22504 Ventura Blvd.
818-883-0531

COLORADO – Denver (Westminster), 80003
8725 Sheridan Blvd.
303-429-2292

FLORIDA – Hialeah, 33012
4705 W. 16th Ave.
305-823-2280

Jacksonville, 32211
9426 Arlington Expressway
904-725-4554

Plantation, 33317
7173 W. Broward Blvd.
305-791-7300

Tampa, 33614
4019 W. Hillsborough Ave.
813-886-2541

GEORGIA – Atlanta, 30342
5285 Roswell Rd.
404-252-4341

HAWAII – Honolulu (Pearl City), 96782
98-1254 Kaahumanu St.
808-487-0029

ILLINOIS – Downers Grove, 60515
224 Ogden Ave.
312-852-1304

Skokie (Chicago), 60076
3606 W. Dempster
312-583-3920

INDIANA – Indianapolis, 46220
2112 E. 62nd. St.
317-257-4321

KANSAS – Kansas City (Mission), 66202
5960 Lamar Ave.
913-362-4486

KENTUCKY – Louisville, 40243
12401 Shelbyville Rd.
502-245-7811

LOUISIANA – New Orleans (Kenner), 70062
1900 Veterans Memorial Hwy.
504-467-6321

MARYLAND – Baltimore, 21234
1713 E. Joppa Rd.
301-661-4446

Rockville, 20852
5542 Nicholson Lane
301-881-5420

MASSACHUSETTS – Peabody, 01960
242 Andover St. (Rt. 114)
617-531-9330

Wellesley, 02181
165 Worcester Ave. (Rt. 9)
617-237-1510

MICHIGAN – Farmington Hills, 48018
29433 Orchard Lake Rd.
313-553-4171

East Detroit, 48021
18149 E. Eight Mile Rd.
313-772-0416

St. Joseph, 49085
2987 Lake Shore Drive
616-982-3215

MINNESOTA – Minneapolis (Hopkins), 55343
101 Shady Oak Rd.
612-938-6371

St. Paul, 55106
1645 White Bear Ave.
612-778-1211

MISSOURI – St. Louis (Bridgeton), 63044
3794 McKelvey Rd.
314-291-1850

NEBRASKA – Omaha, 68134
2311 N. 90th St.
402-391-2071

NEW JERSEY – Ocean, 07712
1013 State Hwy. 35
201-775-1231

Fair Lawn, 07410
35-07 Broadway (Rt. 4)
201-791-6935

NEW YORK – Amherst, 14226
3476 Sheridan Dr.
716-835-3090

Jericho, L.I. 11753
15 Jericho Turnpike
516-334-8181

Rochester, 14623
937 Jefferson Rd.
716-424-2560

N. White Plains, 10603
7 Reservoir Rd.
914-761-7690

NORTH CAROLINA – Greensboro, 27407
4620-C W. Market St.
919-299-5390

OHIO – Cincinnati (Springdale), 45246
131 West Kemper Rd.
513-671-1115

Cleveland, 44122
28100 Chagrin Blvd.
216-292-7553

Columbus, 43229
2500 Morse Rd.
614-475-7200

Toledo, 43615
48 S. Byrne Rd.
419-537-1887

OKLAHOMA – Oklahoma City, 73139
7409 South Western
405-632-6418

OREGON – Portland (Tigard), 97223
10115 S.W. Nimbus Ave.
Nimbus Center
503-684-1074

PENNSYLVANIA – Frazer, 19355
630 Lancaster Pike (Rt. 30)
215-647-5555

Philadelphia, 19149
6318 Roosevelt Blvd.
215-288-0180

Pittsburgh, 15235
3482 Wm. Penn Hwy.
412-824-3564

RHODE ISLAND – Warwick, 02886
558 Greenwich Ave.
401-738-5150

TEXAS – Dallas, 75218
12022C Garland Rd.
214-327-4835

Fort Worth, 76116
6825-A Green Oaks Rd.
817-737-8822

Houston, 77008
1704 W. Loop N.
713-869-5263

San Antonio, 78216
7111 Blanco Rd.
512-341-8876

UTAH – Salt Lake City (Midvale), 84047
58 East 7200 South
801-566-4626

VIRGINIA – Alexandria, 22303
6201 Richmond Hwy.
703-765-5515

Virginia Beach, 23455
1055 Independence Blvd.
804-460-0997

WASHINGTON – Seattle, 98109
505 8th Ave. N.
206-682-2172

WISCONSIN – Milwaukee (Wauwatosa), 53226
845 N. Mayfair
414-453-1161

Phone orders accepted.

Heath Zenith Computers & Electronics

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

Heath/Zenith
Users'
Group

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

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

Production Coordinator .... Lori Lerch
(616) 982-3794

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

HUG Bulletin Board ..... (616) 982-3956

HUG Parts Ordering ..... (616) 982-3838

Contributing Editor .... William M. 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
P.O. Box 217
Benton Harbor, MI 49022
(616) 982-3838

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

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

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

REMark®

Official magazine for users of HEATH ZENITH computer equipment.

Volume 9, Issue 4 • April 1988

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.***
164	*D-G Electronics*	*46*
160	*Ecosoft, Inc.*	*16*
104	*FBE Research Company, Inc.*	*59*
*****	*HEPCAT*	*22*
*****	*HUG Authors*	*30*
*****	*HUG MCP*	*68*
*****	*HUG Members Only*	*67*
*****	*HUG PBBS*	*2*
107	*Paul F. Herman, Inc.*	*36*
137	*Jay Gold Software*	*10*
111	*KEA Systems Ltd.*	*21,40*
114	*Micronics Technology*	*42*
117	*Payload Computer Services*	*6*
132	*RAM Technology*	*60*
119	*S&K Technology, Inc.*	*56*
122	*Secured Computer Systems*	*29*
*****	*Veritechnology Elec. Corp.*	*3*

On The Cover: *Pictured is the eazy pc, a useful tool for people who are not particularly interested in learning the intricacies of DOS and how a computer works. See On The Leading Edge on Page 17.*

★★★ 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	$10.00
MS-463-7	Multiplan	$195.00	$10.00
MS-253-1	BASIC-80 (8-bit)	$175.00	$10.00
CD-463-2	Condor File Manager	$299.00	$10.00

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

HIGH DENSITY 1.2 MEG DRIVES. External floppy drive set-up complete with drive, power supply, case and cable. Ready to connect to your 8" floppy controller. $277.00. Dual Drive Unit $424.00

COLOR GRAPHICS UPGRADE. All memory chips (16 pieces 150 ns) required to update Z-100 Series computer for color. Installation instructions included. Order Memory Kit #100-64-16 .... $36.00

MEMORY UPGRADE. All memory chips (9 pieces 150 ns) required to upgrade from 128K to 192K RAM. Installation instructions included. Order Memory Kit #100-64-9 .... $20.25

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 $182.50 with 27 256K RAM chips included.

UCI MEMORY UPGRADE PAL CHIP SET. For Z-100's with the newer motherboard (181-4918 or greater). This set permits installation of 256K RAM chips on the motherboard. PAL set $64.00 or $187.00 with 27 256K chips required for 768K of memory on the motherboard.

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 Scftware 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 $408.00, One MEG $490.00, Two MEG $653.00 Add $35.00 for EasyDrive RAM Drive Software if desired.

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 $578.00, 31 MEG $598.00 System without Drive and Controller $239.00

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

★★★ 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

Magnolia Microsystems Memory Upgrade Card. Installs in your H-89 to increase memory from 48K to 64K. Payload price $50.00

INTERNAL DUAL DRIVE SETUPS. Includes two half height double sided disk drives and all hardware and connectors required to mount inside your H-89.
MITSUBISHI MF501 Setup, 48 TPI, 6 MS seek, 320K $220.00

★★★ 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	$152.00
MITSUBISHI MF353	3.5" in 5.25" frame720K	$109.00
TOSHIBA ND04	5.25" 48 TPI DS/DD 320/360K	$89.50
TOSHIBA NDO4	with grey faceplate instead of black	$89.50
TOSHIBA NDO8	1.2 MEG for Z-200 or AT Computer	$119.00

★★★ UPGRADE ACCESSORIES FOR Z-148/150 ★★★ SERIES COMPUTERS

VMM 150 Video Card Eliminator for Z-150/160 computers. Allows use of EGA color card $45.00

SmartWatch from FBE Research. Installs in ROM Socket on CPU Board in Zenith computer series Z-100/148/150/158/160. This 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. For Z-150/160 only $99.00

Winchester Hard Disk Drive Internal Set-up. Includes Winchester drive, controller/interface card, cables and all hardware. With 20 MEG (formated) drive $339.00 30 MEG $359.00. 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. $118.00

Winchester Hard Disk Drive External Set-up. Includes Winchester drive, controller/interface card, power supply and case with fan.
With 20 MEG (formated) drive $528.00
With 30 MEG (formated) drive $548.00

★★★ SEAGATE HARD DISK DRIVES ★★★

ST-225 20 MEG Winchester Hard Disk	$295.00
With Western Digital Controller & Cables	$339.00
ST-238 30 MEG, Requires RLL type controller	$309.00
With RLL Controller & Cables	$359.00
ST-4038 30 MEG High Speed for Z-200	$544.00
ST-4051 40 MEG High Speed for Z-200	$629.00
ST-4096 80 MEG High Speed with Software	$929.00
ST-251 40 MEG High Speed Z-150/200	$489.00

★★★ POWER SUPPLIES AND CASES FOR DISK DRIVES ★★★

Rugged steel construction with heavy duty power supplies. Purchase with drives and we will install drives in case.

Single 5.25" unit for Full Height Drive	$68.00
Dual 5.25" unit, for Half Height Drives	$92.00
Face plate for single drive in dual case	$8.00
For WINCHESTER Drive, with fan	$191.00

★★★ COMPUTERS ★★★

DTK 1000 XT TURBO Computer System. These are assembled and tested in our shop using the highest quality components - Mitsubishi floppy drives, Seagate hard disk drives, 120 ns RAM chips, 150 Watt power supply, expanded 101 key AT style keyboards, AT junior slide cases with keylock, etc. Each unit is burned in and fully tested prior to shipment. The front panel has "power on" and "hard disk access" indicator lights together with a Reset button and a lighted Turbo Speed button. In Turbo Speed the computer operates at 10MHz. We can prepare one customized to your specifications with Tape Backups, Hard Disk Drives, CGA or EGA Color, 3.5 or 5.25 inch floppy drives, Etc. Please call or write and we will prepare a quote for you. Two of our most popular configurations are quoted below. Add only $5.00 for UPS shipping within Continental U.S.

Package #1. Computer with one floppy drive, 1 Serial and 2 parallel ports, 512K RAM, Clock/Calendar and amber or green Monitor with tilt swivel base $689.00

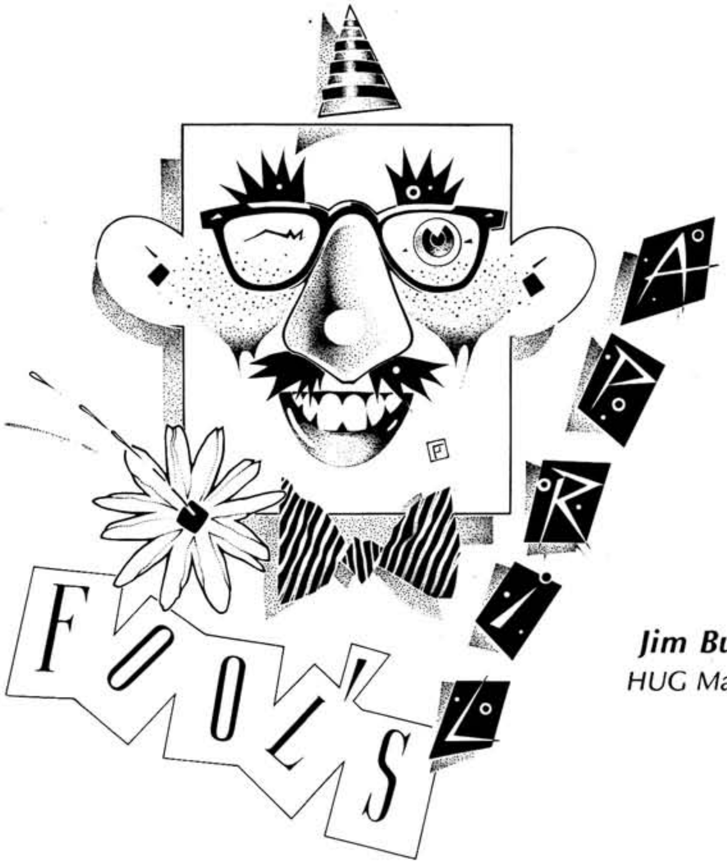
Package #2. Computer and Monitor as above but with two floppy drives, a 30 Meg hard disk drive and 640K RAM $1096.00

ZFL-181-93 LAPTOP PORTABLE with the amazing Supertwist LCD backlit screen 640K RAM, two 720K 3.5" disk drives, clock, P & S Ports, 8 MEG. Retail $2399.00. Payload $1660.00

PAYLOAD
COMPUTER SERVICES

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 8.0% sales tax. We accept purchase orders from schools, government and approved accounts.



Jim Buszkiewicz
HUG Managing Editor

No APRIL FOOLS this year! Everything I have for you this month is pretty serious, so read on!!

First, I would like to announce that Nancy Strunk, our Software Coordinator, will be leaving The Heath Users' Group for a job back at the Heath Company main plant. We all hate to see her leave, but want to wish her the best of luck.

Wow! I hadn't realized there was so much interest in a beginners' column. Well, you talked, and I listened. Starting in June of this year, we're going to start a new monthly column specifically for beginners. The content of this column will specifically address your computer, the operating system, and peripherals. It will be written by one of our contributing editors, Bill Adney. I think you'll find Bill's approach most logical and down-to-earth.

I would also like to start a second column for beginners. This column would be entitled: "Getting Started With..". The idea being, a beginner's approach to application software such as: dBase III, GWBasic, WordStar, Word, Multiplan, etc. These articles would **not** be reviews of these particular products and would be complete in of themselves. Assuming the reader has very little, or no previous knowledge of the product, each article would be a tutorial in getting that reader started with that certain piece of application software. The main idea is 'get the reader started and productive'. Much of the correspondence I've received, indicate that there's a lot of good software out there just sitting idle, because the level of the users' manual for that software is aimed at the moon instead of the earth!

This second column is 'open' to anyone interested in writing for it, and HUG's Major Article Program will apply. The reason for it being open to anyone, is that, I believe no one single person can have all the many types of application software available. Secondly, when an individual has been using a piece of software for some time, he gets to 'know' the application quite intimately. The success of this column will depend on REMark's more knowledgeable readers. C'mon let's share the knowledge!!

Apparently there's been some problem in obtaining the HV-2000 Computer Voice kit. In considering this, I would like to extend the current "HV-2000 Applications" contest (see Feb 1988 REMark) deadline from May 31st to July 31st 1988.

I would like to tell you about a single item we have for sale on the HUGPBBS 'Bargain Centre'. We have a limited quantity of ZA-180-45 battery packs. These are 12 volt 2.5 amp/hour NiCd battery packs for the Z-180 series laptop computer, but can be used for just about **any** 12 vdc application. One user is using these packs for his portable 'HAM' transceiver. These are brand new surplus items, but being sold 'as-is'. In the Heath Catalog, they're $119.95. If you buy them from the Bargain Centre, they're $30.00!! You can also call in your order to (616) 982-3838.

Finally, we've got two super neat things for you this month; a super RTTY program for all you radio amateurs (see the new products section), and for those HUG members that built an H-200 (241/248), a coupon worth $1200 off the regular price of the HUG-386 Upgrade Kit (see the ad elsewhere in this issue)!!!

*

BUGGIN' HUG

Assembler for the NEC V20

Dear HUG:

In his article, "Installing and Programming the NEC V20," in the June 1987 issue, Kevin Lerch gives an excellent and helpful review of the features of the V20.

In speaking of the unique NEC instructions, he says, "(They) are not supported by widely available software. Therefore, they will have to be hand-coded, and relocatable addresses cannot be used."

There is, however, a widely-available assembler, A86 by Eric Isaacson, 416 E. University Street, Bloomington, IN 47401, which not only supports these instructions but is highly desireable on other grouds, including speed, lack of complexity and red tape, support for object libraries, the nicest macro facility ever seen, and too many other things to mention.

It is being sold as shareware for the time being, but it is not some amateur program. I think it is quite likely to replace MASM as the standard 80 x 86 assembler, and anybody in HUG who either programs in assembler or writes modules to be called from high-level languages will be doing himself a favor by getting it, whether he has a V20 or not.

Sincerely,

David Nord
Route 3, Box 94
Caledonia, MN 55921

Dear HUG:

Webster Definition: STODGY...Moving in a slow plodding way...Dull...Boring...EXTREMELY OLD FASHIONED...Drab

I just couldn't take it any longer, I MUST speak my opinion on the subject of the "KEYBOARD WAR". The word "STODGY" describes my feeling toward the whiners who are complaining about the new 101 style keyboard being supplied with our new machines. I personally have one at home and one at work, so I'm not speaking wthout experience. Sure, I felt a little lost at first. Sure, I hit the shift key instead of the control key a few times. But now that I have become accustom to the layout, I find it much easier to use.

If any of you remember the old Z-100, with the function keys across the top, the computer industry heralded the incredible keyboard layout. The new function layout across the top was the thing to have.

Then low and behold the IBM PC came out. Oh no, the function keys are in the wrong place, they're along the side... everyone complained. Well the Z150 computer came out and it also had the function keys on the side. After a year or so things quieted down and everyone seemed happy once they became familiar with the new layout.

Now the new ZKB-2, 101 style keyboard raises the same old complaints. The function keys are back where they were to begin with, like the old Z100. I'm sorry, but it just doesn't take me that long to get acquainted to this new keyboard to warrant such whining.

In response to Mr. Adney's article, "On The Leading Edge" in the Nov. REMark, give the thing a chance! In six months the only sobbing you'll hear will be from those people that don't have the 101 keyboard. In response the Mr. Buszkiewicz's "Monkey See..." article, besides voiding the warranty on the unit, I would not want to put my brand new $4000 unit under the knife.

Well, I've made my point I guess, so I'll sign off for now. One note to keep in mind, if you don't like change, don't play with computers. I LOVE MY 101!

Live Long and Prosper,

Kevin Lerch
Heath Computer Consultant

Dear HUG:

How many members of the Science and Engineering community are out there? I've been a HUGGIE (and H-120 user) for over four years and **occasionally** see a letter from someone of that persuasion. I've used my H-120 occasionally for analysis in my profession as a physicist/applied mathematician. This work involves algorithm development and is usually implemented in FORTRAN. I'm usually developing/testing a method that is soon turned over to a programmer for running on a mainframe or mini after I've verified its correctness and usefulness.

I use advanced math methods for things other than science -- notably stock market analysis. For this, I love to use ZBASIC because of the graphics capability and the quickness of debugging codes using the interpreter mode. I then compile the codes (if they contain time-consuming calculations) to achieve the much higher speed of running machine-language code. The drawback is that compiled ZBASIC does not support my 8087 coprocessor (my FORTRAN compiler does).

In the last few months I've begun to overcome this problem by learning some assembly language (I just received my Heath MACRO-86 diploma). I've been writing assembly language subroutines that use the 8087 and that can be called from and linked to ZBASIC main programs. I've combined the subroutines into a library file (named it B8087.LIB) so that to use any of them I need merely give the linker the name of the library. My Fourier integral transform runs eight times the speed that it ran as a compiled ZBASIC program. The fast Fourier (Cooley-Tukey) algorithm runs twice as fast as the identical algorithm in FORTRAN (that supports the 8087). Currently, the library has the trig functions, Fourier integral and FFT, plynomial routines (value, derivative and synthetic division to eliminate roots) and a green-plane pixel getting routine for screen bit maps. I plan to keep expanding it to meet new needs.

I'm wondering if anyone knows of a similar library -- am I reinventing the wheel? (I'm aware of and using material from the book by Richard Startz, "8087 Applications for the IBM PC and other PC's.") Is anyone interested in obtaining such a library if it were made available? What routines would you like in such a library? I invite your letters.

Sincerely,

Ken Granzow
1079 Haverhill Place
Colorado Springs, CO 80919

Continued on Page 36

New Tool For The H/Z-89/90

Kenneth L. Owen
312 Main Street
P.O. Box 47
Six Mile, SC 29682

The H/Z-89/90 came stocked with a removable hard top and a fan. Now, it can also sport windows! (No, this is not another article cooling the H-89.)

We have all heard the complaint that the H/Z-89/90 was dead, and the arguments that indeed they still do live and are working in New York, Atlanta, Houston, Six Mile, . . . Where? Obviously, Spectre Technologies believes they still live and the users will want a good software tool. I'm talking about their package named PRESTO PLUS!, Version 3.

I have just received the software and have not even fully utilized the functions in my system. I have flexed the muscles of the package and found them to be both strong and gentle. That is to say, the software works! And it works with everything I've tried to date without disturbing other system operations.

PRESTO! is a disk based, memory co-resident utility. Let me see if I can make that a little clearer. The core module of PRESTO! is loaded in high memory and also replaces the users original CCP with its own Z-80 based CCP. That means it does consume some memory, and since we only have 64k, you want to know "How much memory?"

The answer is: "Not much (about 9k) for the features gained." The program uses disk resident overlays to accomplish the various tasks. When you invoke the utility, it stops the program in progress, jumps to the core program, and presents a window on screen to prompt for the needed task you select. It loads the necessary overlay and executes your request. When done, you are returned to the original program with not so much as a cursor-hair out of place. PRESTO! can be removed at the system prompt without having to reboot if the space is needed.

The program is a good housekeeper, also. It automatically saves its own environment, thus 'remembering' where you were when. Not only that, it also will customize its operation for the program you are currently running. So much for the general stuff, let's see what it can do.

PRESTO! is invoked by a trigger key sequence, the CONTROL key and some other key pressed simultaneouly. The trigger can be user defined so that it does not conflict with the current application.

The core module opens a window on screen and presents a menu of tasks:

```
+-----------------+
|     PRESTO      |
|                 |
| K - Key defns   |
| N - Notepad     |
| T - Timepad     |
| C - Calculator  |
| D - Disk utils  |
| S - scrn utils  |
|                 |
+-----------------+
```

The overlays are called by typing the letter for the desired function.

<K> Define Keys — This function allows redefining a key to do a function. This can be a simple key substitution, or a string of keystrokes to accomplish a task (macro). I am writing this using a word processor, and used a key assignment to quickly insert the centering commands for the chart above.

This function may be used to insert text, or issue a command with a simple user defined key sequence. The key definitions may be saved in a file, and if properly named, will automatically be loaded with the application program for which they are to be used. The speed of execution of the macro may be adjusted for various application programs. If Mr. Katz were still working on the H/Z-89/90, he would LOVE this macro feature. Some really wild possibilities come to mind with this kind of utility.

<N> Notepad — The notepad function opens the lower half of the screen and presents a simple, but quite functional, word processor. The commands are similar to those used in WordStar and function to blank the notepad, insert, delete (character and line), find, print, and a nifty one called Undo. The last one will undo all changes for the current access.

The notepad can write to both the disk or the application program screen and recall saved notepads (and other text files) on demand. The file name of the current file is displayed on a status line.

<T> Timepad — The time pad for the H/Z-89/90 displays the current date, and either the current month (default) or the requested month for years 1950 through 2049. The current date can be written to the application program screen, or to the notepad. The calendar can also be copied to the upper left-hand corner of the notepad.

If there were any disappointments in the package, it was here. I had hoped that the BIOS clock option was supported to allow time and alarm functions, as well. These functions were available in the versions for other brands of computers which have clock modules installed. Spectre said they want us to notify them of other clocks in use, and that they may add support. The letter is already in the mail!

<C> Calculator — This option presents a calculator overlayed on the screen. It is a true general purpose, eight digit floating point calculator and programmer's calculator, performing math and conversions in-between decimal, octal, hexadecimal, and binary.

It won't compute hyperbolic-sines, but you do get basic add, subtract, multiply, and divide, along with exponentiation, inverse (floating point only), remainder (programmer mode only), bit wise and/or, complement and negate. There are six memories, three for floating point and three for programmer's mode. The calculator 'display' may be written to either the notepad or the application program screen.

<D> Disk Utilities — The utilities provided are just the standard CP/M commands. The boon is that you may now use them from within ANY program. Some good software packages I use allow doing a disk directory, file renaming, etc. Now, even the programs that I write have disk utilities! You may perform DIRectory, View, Copy, Rename, Erase, and even change User.

<S> Screen Utilities — This panel allows sending the current screen to the printer, to a text file, or to a graphics file without aborting the application program. Another feature is the ability to blank the screen and lock the keyboard with a user defined password. This will not keep someone from turning off the power and rebooting the machine, but it does a good job of putting things away and out of sight with some security. Retyping the password returns the system to where you were.

PRESTO! and now my old workhorse acts a little younger. The boss will appreciate the increased productivity by not having to exit and re-enter various programs just to find a file name or retrieve a bit of data. He will be dazzled by how fast I can get out documents full of difficult-to-type repeating terms and phrases. And maybe now I can plan and stay on a schedule and improve my documentation with the Timepad and Notepad functions. All in all, PRESTO! is a class act. First class software supplied with good documentation is usually appreciated.

PRESTO PLUS! is available for the nominal price of $39.95, plus shipping and handling from:

Spectre Technologies, Inc.
22358 Ventura Boulevard., Suite E
Woodland Hills, CA 91364

CP/M is a registered trademark of Digital Research, Inc.

WordStar is a copyright by MicroPro International Corporation. *

Using Data Compaction In Programs

About The Author

Ronald Perrella is a 19 year old sophomore at Rose-Hulman Institute of Technology in Terre Haute, Indiana, studying Computer Science. He is the author of SPELL5 and SPELL5F. the HUG spelling checkers. He enjoys working with symbols much more than with numbers and is a fanatic about learning programming languages. He also would welcome any comments about his article.

Ronald Perrella
Rose-Hulman Institute of Technology
5500 Wabash Avenue
Terre Haute, IN 47803

Data compaction techniques vary in both type and usage. In this article, I will present some data compaction algorithms which you can use inside of programs to give yourself greater storage capacity. Of course, de-compaction will also be treated so that you can effectively manipulate your data.

First of all, why use data compaction? After all, most computer users today own 16-bit machines with at least 256k of memory, sometimes 1 megabyte, and even more in the case of the extended or expanded memory of the PC clones. Well, let's look at some common programming languages to find the answer to data usage.

Take a look at ZBASIC (or MBASIC-80 for you 8-bitters out there). It allows you to make 64k programs. That's all. You can CHAIN, RUN, and COMMON all you want, but you only get to use 64k at a time. BASIC does not care how much memory you really have. So, obviously, every byte of memory you use in a BASIC program must be wisely spent. Many programmers, however, have long since stopped using BASIC as their programming language, using Turbo Pascal instead. Well, guess what . . . Turbo only allows you to have a single 64k data segment. Of course, you can use the Turbo Pascal heap and stack, but that is a different story. And the situation is quite similar for those people who program in C using a Small-Model C compiler. You only get a 64k data segment.

So what are your solutions? Make your program smaller and thus less capable; stop using speed-efficient data structures, replacing them with space-efficiency ones; or use data compaction, thus retaining speed-efficiency and space-efficiency. In this light, data compaction is an attractive alternative. But is it both feasible and easy to program? My answer is "Well, almost." The data compaction method you use depends largely on the data you are dealing with. Along with this article, you will find several listings in Turbo Pascal that should be easily converted to whatever Pascal dialect you are using. I chose Turbo Pascal because it is a common dialect and a pure pleasure to use.

For my first example, I have chosen the symbol table of a compiler. For those of you who have never written a compiler, here is a quick resume of what the symbol table is used for. First of all, the symbol table contains the various symbols used in the source program, as well as additional data pertaining to the symbol type (procedure, function, constant, or variable, for example), symbol size, and the address where the symbol is allocated (its beginning). Let's assume that the symbols to be stored in this table are always upper-case and belong to the following character set: A-Z. In addition, each symbol may be anywhere from 1 to 32 characters long and the table must be able to accommodate 2000 symbols. Of course, one could just allocate 32 characters per symbol and just make a large table. But the size of a 2000 element table (in bytes) would be '2000 * (32 + 1)' because 1 byte is used to keep the length of the string. Obviously, this solution is unacceptable because it would require 66000 bytes, and we are only allowed 65536 or 64k bytes of random access memory.

Let's think about what we can do here. The letter range is 26 long. And we need some way to mark the end of a word, possibly without using an extra byte (whether it be a length byte or end-of-string marker). An easy way to mark the end of a word is to use the 8th-bit of the last letter. Just using this simple solution could save us 2000 bytes in the example above. But what we need is to compact the symbols themselves. We can only pack two 26 number sequences in a single byte. This is easy to show. Let's look at the bit-map of one byte (see Figure 1).

How many bits do we need to represent

Figure 1

```
Most Sig. Bit -\                    /-Least Sig. Bit
               v                      v
               8  7  6  5  4  3  2  1    <- bit number
              [0  0  0  0  0  0  0  0]   <- byte
             128 64 32 16  8  4  2  1     <- assoc. values
```

```
                                       Figure 1

Most Sig. Bit -\              /-Least Sig. Bit
           v            v
           8 7 6 5 4 3 2 1  <- bit number
          [0 0 0 0 0 0 0 0]  <- byte
          128 64 32 16 8  4  2  1   <- assoc. values
```

26 letters? Four bits (a "nibble") can represent any number up to 31 (=16+8+4+2+1). Thus, any four bits can represent a letter. By packing two letters per byte, we have halved the storage need to store one symbol. This is very nice; however, how do we represent the end of word marker now? If this problem is not yet apparent to you, take a look at this example.

ABCDEFG is compacted into [AB] [CD] [EF] [G] which in binary form gives this:

```
[AB] => [0001][0010]
      [CD] => [0011][0100]
[EF] => [0101][0110]
      [G ] => [0111][0000]
```

Which takes up four bytes. The last byte should contain the 8th bit end-of-string flag and should look as follows:

```
[G ] => [1111][0000]
        - end-of-string flag.
```

By masking in the flag off, you can get the value for 'G'. However, let's suppose the last letter was a 'P'. The letter 'P' is the 16th letter of the alphabet. Let's see what happens:

```
[P ] => [1000][0000]
```

A priori, it looks O.K. But notice that it occupies the 8th bit; the same bit we are using to mark the end-of-string! We must, therefore, give up using the 8th bit as an end-of-string marker. However, there is still a way to save space. Use a null byte to represent the end-of-string. But make it optional. That is, if there already is a null nibble in a byte, it represents the end-of-string. I have used single quotes to delimit strings and square brackets to delimit bytes in the following example:

```
'ABCDEF' =>[AB][CD][EF][  ] = 4 bytes
'ABCDE'  =>[AB][CD][E ]     = 3 bytes
```

Finding the end of a string is very easy. Here is the algorithm in pseudo-code.

```
len = 1
while (s[len] and $0f) <> $00 do len=
   len+1;
```

The 'and' instruction used here is the bitwise 'AND' operation. The 'and $0f' causes the upper nibble of a byte to be set to zero without changing the values in the lower nibble. When the lower nibble is all zeros, you have found the end of the string.

For the compaction and decompaction algorithms, see the accompanying listings (i.e., Listing 1 and Listing 2).

So far, we have walked through a nice and relatively easy compacting method. It compacts your alphabetic strings into half of their original length. What effect does this have on your program? Three effects come to mind. First, more room to store data is obvious. The second, increase complexity, comes from reading the listings. The third, increases speed, may come as a surprise. But it suffices to think about what the computer has been doing in your program all along. Although there is increased storage time, because of the compacting, there is increased retrieval time since there is half the amount of data to compare and move or do anything you wish. However, the fourth effect is much sneakier. You can now perform comparisons as easily as before. For example:

```
'ABCDE' > 'ABCAA' is equiv. to
   [AB][CD][E ] > [AB][CA][A ]
```

Because of the way numbers work, you can now compare two characters at a time! You have decreased comparison times by exactly one half! The algorithm to perform the comparison is easy: "while not end of string, compare each element numerically until a difference occurs and

Listing 1

```
PROGRAM Compactor(input,output);
{ Public Domain 1987 Ronald J. Perrella }
{          *** Listing # 1 ***       }

Type
Symbol       = String[32];
CompactSymbol = Array[1..17] of Char;

Var
i       : Integer;
str     : Symbol;
compstr : CompactSymbol;

{
The following is a puny binary number printer.  It was not meant
to be fast nor fancy. I just needed it.
}

Procedure PutBin(m:Integer);       {PRINT M AS A BINARY NUMBER}
Var
i      :Integer;
b      :Array[1..8] of Char;
Begin
   m := m * 2;
   For i:= 8 Downto 1 Do Begin
      If odd(m div 2) Then b[i]:='1' Else b[i]:='0';
      m:=m div 2;
   End;
   Write('[');   For i:=1 to 4 Do Write(b[i]);
   Write('][');  For i:=5 to 8 Do Write(b[i]);
   Writeln(']');
End;

{
The following is the compaction routine.  It packs two letters
to a byte and marks the end by a null nibble.  It assumes that
the Symbol to be packed has already been checked and found valid
(i.e. all characters in ['A'..'O'] only.)
}
```

```
Procedure Compact(t:Symbol; var s:CompactSymbol);
Var
len,i,j    : Integer;
C          : Char;
m,n        : Integer;

Begin
    len := Length(t);
    i := 1;           {index t[]}
    j := 1;           {index s[]}
    If odd(len) Then Begin
      While (i <= len - 1) Do Begin  {leave the last one alone...}
           m := ord(t[i]) - ord('A') + 1;   {make code 1..26}
           n := ord(t[i+1]) - ord('A') + 1;
           s[j] := chr(m*16+n);
           i := i + 2;
           j := j + 1
     End;
     m := ord(t[i]) - ord('A') + 1;  {prepare the last char.}
     s[j] := chr(m*16)           {mark end of string}
  End Else {If even(len) Then } Begin
      While (i <= len) Do Begin
           m := ord(t[i]) - ord('A') + 1;   {make code 1..26}
           n := ord(t[i+1]) - ord('A') + 1;
           s[j] := chr( m * 16 + n );
           i := i + 2;
           j := j + 1
     End;
     s[j] := chr(0)         {mark end of string}
  End
End {Compact};

{
The following is the decompaction routine.  It unpacks two letters
from each byte.
(i.e. all characters in ['A'..'O'] only.)
}

Procedure DeCompact(s:CompactSymbol; var t:Symbol);
Var
i        : Integer;
hi,lo     : Integer;

Begin
     i := 1;                   {index s[] source}
     t := '';                   {destination}
     repeat
          lo := (ord(s[i]) and $0f);                 {Low nibble}
          hi := (ord(s[i]) and $f0) div 16;           {High nibble}
          If hi <> 0 then t := t + chr(hi-1+ord('A'));
          If lo <> 0 then t := t + chr(lo-1+ord('A'));
          i := i + 1
     Until lo = 0;
End {DeCompact};

Begin
     Write('Enter all alphabetic upcase string to compact ->');
     buflen := 17;        {makes input string 17 chars Max. (TURBO-ONLY)}
     Readln(str);
```

```
     Compact(str,compstr);   {assumes str is a valid symbol}

     Writeln;
     Writeln('The compacted string consists of the following bytes:');

     i := 1;              {index the compstr[]}
     While (ord(compstr[i]) And $0F <> 0) Do Begin   { $0F = 15}
         Write ('#',i:2,' ');
         PutBin(ord(compstr[i]));
         i := i + 1;
     End;
     Write ('#',i:2,'  '); PutBin(ord(compstr[i]));

     Writeln;
     Writeln('The same string decompacted gives the following:');

     str:='xxxxxxxxxxxxxxxxxxxxxxxxxx'; {wipe out previous values for proof.}
     DeCompact(Compstr,Str);

     Writeln(' <',str,'>');
End.
```

Listing 2

```
Program Compactor(Input,Output);
{
 This program compacts a text file using the
      'White-Space common elimination scrunching'
 Algorithm.
}

{          *** Listing # 2 ***    }
Label 99;

Const
Max       = 1000;
Tab       = #08;
Cr        = #13;
Lf        = #10;
Whitespaces : Set of Char = [' ',Cr,Lf,Tab];

Var

outfilename,
infilename : String[255];
infile    : Text [5000];   {[5000] means have buffer contain 5000 chars}
outfile   : Text [5000];

IBuf,OBuf : Array[1..Max] of Char; {buffers for i/o}
Last,
C,D       : Char;

Count     : 0..256; {1 byte}
Searching : Boolean;

Begin
    Writeln('Enter input file name.');
    Readln(infilename);
    Writeln('Enter output file name (must be different).');
    Readln(outfilename);
    if infilename=outfilename then halt;

    Assign(infile,infilename);
    Assign(outfile,outfilename);
    Reset(infile);
    ReWrite(outfile);

    Searching := false;

    While not eof(infile) do Begin
        Read(infile,c);
99:
        If c in whitespaces then begin
            last :=c;
            Read(infile,c);     {Next Char}
            if last=c then begin
            Searching:=true;
            count:=2;      {have already two of them...}
            Repeat
                Read(infile,last);
                if last = c then Count := Count + 1;
                if count = 256 Then searching:=false;
            Until (Last <> C) or eof(infile) or (not searching);
            D := c;
            C := chr(ord(C) + 128); {set 8th bit on.}
            Write(Outfile,c);
            C := Chr(count);
            Write(Outfile,C);
            if last <> d then Write(outfile,last);
          end else begin {c := next char and last := prev. char}
            Write(outfile,last);
            Goto 99; {check to see if c in whitespace}
          end;
      End else Write(outfile,c) {write normal character out}
    End;
    Close(infile);
    Close(outfile);
End.
```

record this difference."

Also, you can still perform a binary search, you can still perform exchange sorts or any other kind of sorts, and you can perform data movement twice as fast as before! So, packing data into bytes can be rewarding both in data movement, as well as retrieval time. However, it requires that characters belong to a very restrictive 15 character range of symbols. That's all folks! Now, wasn't that FUN? Well, we are not finished. Oh boy, here we go again . . .

What we need is a more generalized compaction method, one that is suitable for the 7-bit ASCII standard character set.

Now, let's suppose that you want to send a file to a friend by way of a MODEM connection. The trouble is your friend lives in Hawaii and you live in Maine. You do not want to pay too great a phone bill for such a long, long distance call. Your file takes up 30k and is a geographical report, but contains only the printable ASCII codes and the space, newline, tab and carriage return characters a finished text file should contain. What you need is a simple text compacting program. While many such programs exist in the Public Domain, you are reading this article to learn how to do such things. So, you set about to write your 'scrunch' program. There are several ways you can scrunch your text files. I have given my own names for the first two of the following methods:

1. 'white-space common elimination scrunching'.
2. 'general common elimination scrunching'.
3. binary code compaction.

The first way comes from a study of the English language and computer text files, in general. It seems that spaces occur everywhere. Indeed, they seem to be the most common character occurrence. While we are at it, we can also notice that carriage-returns and line-feeds are also common occurrences. All of these characters are termed "White-Spaces", because they do not show up in printed text. So why not find a way to pack the white-spaces into less space (no pun intended)? This first compacting method consists of replacing all white-spaces that appear in multiples by a single character followed by a count indicating how many characters were originally there. This is a simple and effective way to compact data, since white-space occurs so often in texts. Here is an algorithm to accomplish this:

```
set mode to 'not searching for multiples'.
get a character from the text.
if this character is a white-space then
   get the next character from the text.
   if it is the same character as before then
      set mode to 'searching for multiples'.
      get the next character until a different character
       is found and remember the number of characters
       read.
      set the upper (8th) bit of the search character on.
      write the search character followed by the count.
   endif
endif
repeat the above until end of file.
```

Of course, this algorithm is somewhat incomplete. Several points need clarification. For example, we have not explicitly given the size of the count which would succeed the white space. In the example program (Listing 3), I have decided upon using a 1 byte count. In other words, a string of 400 spaces will take up 4 bytes: 2 representing a space and a count of 256 followed by 2 bytes representing another space and a count of 144. The only difference this makes in the algorithm is that it must set the mode to 'not searching for multiples' and have the loop continue as if nothing had happened. This is conveniently illustrated in the example program.

The second compaction method consists of generalizing the above algorithm so that any character that occurs more than once is compacted. This, for example, means that the series of asterisks that occur so often in program comments are packed also, as well as the inevitable spaces. The algorithm actually becomes a bit simpler, which really does not surprise me because it is an empirical result that many programmers (and scientists) have found: generalizing a solution often makes it simpler. Looking for generalities and using them is a good way to approach just about any problem. My solution is found in Listing 4.

For both the first and second methods, a single decompaction program is needed, as show in Listing 3. This was my first clue as to the effectiveness of not just compacting the white-spaces, but also any other frequently repeated character. This program just reads a character. If the 8th bit is not set, the character is written out "as is". Otherwise, the character's 8th bit is set to zero, the count is read from the next character, and the character is written out that many times. All of this is done

Listing 3

```
Program DeCompactor(Input,Output);
{
 This program decompacts a text file using the
     'White-Space common elimination scrunching'
 Algorithm. It also decompacts the 'General common elimination scrunching'
 files too !
}

{        *** Listing # 3 ***    }

Var

outfilename,
infilename : String[255];
infile     : text [5000]; {[5000] means make buffer contain 5000 chars}
outfile    : text [5000];

C,D       : Char;
i         : Integer;

Begin
    Writeln('Enter input file name.');
    Readln(infilename);
    Writeln('Enter output file name (must be different).');
    Readln(outfilename);

    if infilename=outfilename then halt;

    Assign(infile,infilename);
    Assign(outfile,outfilename);
    Reset(infile);
    ReWrite(outfile);

    While not eof(infile) do Begin
        Read(infile,c);
        if ord(c) and 128 <>0 then begin     {8th bit was set}
           c:=chr(ord(c) and 127);          {remove 8th bit}
           Read(infile,d);                  {get the count}
           For i:=1 To ord(d) Do
               Write(outfile,c);            {write the character out}
        End Else Write(outfile,c);
    End;
    Close(infile);
    Close(outfile);
End.
```

```
Program Compactor(Input,Output);
{
 This program compacts a text file using the
     'General common elimination scrunching'
 Algorithm.
}
```

Listing 4

```
{       *** Listing # 4 ***    }

Const
Max       = 1000;
Tab       = #08;
Cr        = #13;
Lf        = #10;

Var

outfilename,
infilename : String[255];
infile     : Text[5000];
outfile    : Text[5000];

IBuf,OBuf  : Array[1..Max] of Char;  {buffers for i/o}
Last,
C,D        : Char;

Count      : 0..256;  {1 byte}
Searching  : Boolean;

Begin
    Writeln('Enter input file name.');
    Readln(infilename);
    Writeln('Enter output file name (must be different).');
    Readln(outfilename);

    if infilename=outfilename then halt;

    Assign(infile,infilename);
    Assign(outfile,outfilename);
    Reset(infile);
    ReWrite(outfile);

    Searching := false;

    Read(infile,c);
    While not eof(infile) do Begin
        last :=c;
        Read(infile,c);    {Next Char}
        if last=c then begin
          Searching:=true;
          count:=2;      {have already two of them...}
          Repeat
              Read(infile,last);
              if last = c then
                Count := Count + 1;
              if count = 256 Then
                searching:=false;
          Until (Last <> C) or eof(infile) or (not searching);
          d := c;                              {save copy of 7-bit C}
          C := Chr(ord(C) + 128); Write(Outfile,c); {set 8th bit}
          C := Chr(count); Write(Outfile,C);        {write count}
          if last <> d then
             c := last; {If found new char, start to check it}
        end else write(outfile,last);
    End;
    Close(infile);
    Close(outfile);
End.
```

until the end of the input file. Quite simple.

The third way consists of disregarding the fact that each character is a byte and instead, representing each character as a unique bit-stream. This approach is based on the work of a mathematician by the name of Hoffman, I believe. This form of compaction is very, very efficient, as well as very, very difficult. This method also causes the compactor to take quite a long time to analyze each character's occurrence in the file to determine which character is associated to which bit-stream, and this table sent along with the text-file by way of the MODEM. But it often gives the best results. I will not go into the details of such a compactor. Its complexity is far beyond the scope of this article. However, Figure 2 shows how the tables might look and what the output could be for a short string.

Figure 2

```
Let:
 'A'  101
 'B'  111
 'C'  1101

Then:
                 <A><B>< C ><A><B> < C>< C>
 'ABCABCABCC' becomes [10111111][01101111][11011101]
   ^- 10 bytes        ^- 1st byte ^          ^
                            ¦-2nd byte     ¦
                                       \ 3rd byte
```

The savings are three-fold in the Figure 2 example. Of course, I could have chosen simply two-bit patterns (i.e., A = 00, B = 01, C = 11) and the savings would have been five-fold. But for a true 127 position character set, three- and four-bit streams would be more common.

There is just one little (big?) problem with all of the above compaction routines: They use the 8th bit. Well, some MODEM connections do not accept the 8th bit. All along, I have made the assumption that we were dealing with a connection which does accept the 8th bit, which we have sometimes been using as the end of string markers in the first symbol compaction system. The way around this would be to decompose each 8-bit byte into a bit stream which can be sent out seven bits at a time and the remaining bit sent out with the next byte, and so on. However, it is not in the scope of this article to cover 8-bit to 7-bit conversion.

Throughout this article, I have been using text as the primary compaction material. Unfortunately, text does not always contain many repeated strings of characters. However, there is a material which does contain many repeated strings of characters: graphics displays. If you take a similar approach to graphics data, you will find that 'general common elimination scrunching' work extremely well. Of course, you must deal with the fact that the 8th bit is part of the data and cannot be used as a flag of some sort. In this case, just follow each and every character by a one byte count. At first, this seems that it will be expensive in terms of data storage, but this compaction method ends up being quite effective.

Another effective way to compact graphics data is to place the count byte only after bytes containing $00 or $FF, since these are the two bytes that occur most often in a stream. I leave it up to you to try out this kind of compaction on your own data. All you need to do is get a graphics dump of an image, modify my "common elimination scrunch" program and away you go. Don't forget to change the decompactor, too!

I hope that you have found this article to be both enjoyable and useful. The study of compaction methods can by quite interesting and enjoyable. But be prepared to work in binary notation quite often . . .

On The Leading Edge

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

Zenith's eazy pc, MS-DOS Manager

As I pointed out in one column some time ago, one of the reasons that you have a "personal" computer is to help you improve productivity in one way or another. For most of us, another reason they are called "personal" computers is that, if someone makes a derogatory comment about OUR computer or manufacturer, we take it real personal.

What you want or need in a personal computer is strictly dependent on your specific application. It is silly to spend $5,000 or more for a 386 system with a hard disk when all you need is a word processor that sees occasional use on a weekend. Perhaps you have a spreadsheet for budgeting, and since taxes are due this month, you may have also bought a tax preparation program of one kind or another.

There is a significant trend in all personal computers to simplify their operation. From an operational perspective, the kindest thing you can say about most 8-bit CP/M computers is that they are difficult to learn. The DOS-based computers aren't any easier, and there is a good argument that they are more difficult because DOS has more commands and features than CP/M ever did. And so, many manufacturers, including Zenith, have developed a product line that is easier to use than most of the others including the IBM PC. It's called the eazy pc.

Zenith's eazy pc

The eazy pc was developed for a specific market -- it was intended to be used by people who are not particularly interested in learning the intricacies of DOS and how a computer works. Many of today's computer users want to DO something useful, like write a letter, and they don't particularly care exactly how that is done, so long as they can write and print a letter. In other words, more people are looking at a computer as a tool instead of a hobby. That's no particular surprise because computers were always intended as a tool to help you perform some specific task, although some of us have developed that into a hobby and profession as well. The point is that an inexpensive electric drill (or computer) may not have all of the features of the expensive ones. It may have less power, but it still does the job. And so it is with the eazy pc.

The eazy pc is essentially available in three basic models. One has two 3.50" 720 KB floppy disk drives, and the other has a single 3.50" floppy drive with a 20 MB hard disk. You can also get a version with a single floppy drive (EZ-1), but that is really not a useful configuration -- you cannot add a second floppy drive because there is no "hole" in the cabinet for it. In order to keep the footprint of the unit small, you also cannot add a hard disk to a system with two floppy drives because there is no room. But if you like to fool around with various drives and other hardware like I do, you will want to get some other system that has more capabilities in that respect.

All eazy pcs have a standard 512 KB of system memory (RAM) and use a NEC V-40 CPU chip running at 7.16 Mhz. A 25-pin parallel port (for a printer) is standard, and a 9-pin connector is provided for a mouse as the COM2 serial port.

Perhaps the most interesting part of the eazy pc design is the fact that the 14" CGA monitor and system unit are physically combined into a single unit. The monitor is attached on a special base that tilts and swivels for ease of use. A single power supply serves both, and the system unit is intentionally difficult to disassemble since it is assumed that users will not want to be bothered with the internal hardware.

The eazy pc is still expandable through the use of some clever add-ons that are quite simple to install. You can upgrade the system to 640 KB of RAM with the optional 128 KB Memory Module. Or you can get the optional Modem/Memory/Serial Module that includes a modem, 128 KB for memory expansion, and a serial port with a 9-pin connector like the Z-200. You can even get a Zenith SmartWatch so that you don't have to reset the date and time every time you reboot the system. The manual suggests that the SmartWatch is a dealer installed option.

The eazy pc comes with the usual excellent Zenith documentation that consists of four manuals: the Owner's Manual, the MS-DOS Setup and User's Guide, a Quick Reference Guide (to MS-DOS commands), and a manual on Using MS-DOS Manager (more on that later).

The Owner's Manual has just six pages showing (with figures) how to get the system up and running. It is the simplest set of instructions that I've ever seen. It starts with suggestions for the physical placement of the computer (not near a window because of glare) followed by how to connect the keyboard, power, printer, and a mouse. Turn the power on, insert the boot disk, adjust the monitor position, and you're running the computer. There are a total of 10 steps, and I thought this particular section was extremely well done.

There is also a quick reference set of instructions affixed to the monitor screen with a removable plastic overlay sheet for those of us who don't like to read a lot of manuals. Still, first time computer users would be well advised to read (or at least scan) the small Owner's manual from cover to cover to see what's there. I have to admit that following these instructions is a whole lot simpler than trying to figure out how to program the new timer that I got for my yard's sprinkler system -- now THAT was a real challenge.

The MS-DOS Setup and User's Guide includes some additional information about the setup of the eazy pc plus the usual reference to the various DOS commands that is included in all of the Zenith manuals. For that reason, it is the largest of the four manuals (about an inch thick). The Quick Reference Guide is a 42 page booklet that includes a condensed command reference and other useful information. The MS-DOS Manager reference is about a quarter inch thick and includes everything you need to know about that software.

Using the eazy pc

Although I did get a complete set of the software and documentation to review for the eazy pc, the last thing I need around here is another computer. I did some testing at my local Heath store just to see how the eazy pc responded to some of my most commonly used programs. I now have a 3.50" drive in my '248, so I created some disks with some of these programs. I checked out these programs on an eazy pc with two floppy disk drives.

First, and foremost, I checked to see that my usual SuperKey and WordStar version 4 configuration worked on the eazy pc. It worked fine. I purposely did not change any of the color settings that I use on my NEC Multi-Sync version, and the eazy pc faithfully displayed these "colors" as shades of gray as expected. Then, I tried Reflex, and it worked fine too. Microsoft Word 3.10 also seemed to have no difficulty when it was also loaded with my special SuperKey configuration, and Word Perfect 4.1 also worked just fine as well. SuperCalc 3 version 2.1 ran fine, and Borland's new Quattro spreadsheet also ran fast as expected.

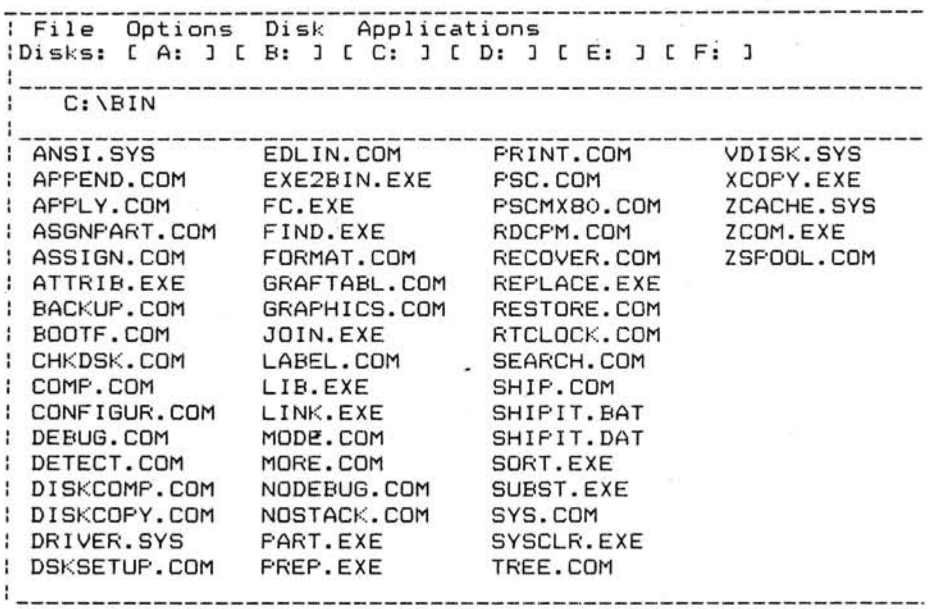
Some of these programs are probably more expensive and sophisticated than a usual eazy pc user would want, so I checked out some other programs too. Since I teach at UTA, I checked out PC-TYPE, PC-CALC, and PC-FILE as well as PC-Write. We use those programs in various computer courses, and they worked as expected on the eazy pc with no problems. But there is one other goodie included with the eazy pc that is very nice -- the MS-DOS Manager.

The MS-DOS Manager

The MS-DOS Manager software was developed by Microsoft, although I understand that Zenith provided some input on the general requirements for the software. At first glance, Manager looks suspiciously like Windows, but it is really a shell program. If you have never used one, a shell program usually provides a directory display with various commands that you can use on the files such as Delete, Rename, Copy, and so on. Manager provides all of that plus a capability to run application programs from the shell. The basic shell function displays a directory as shown in Figure 1.

Although it isn't shown on this figure, the

```
************************************************************
```



```
 ---------------------------------------------------------------
 : File   Options   Disk   Applications
 :Disks: [ A: ] [ B: ] [ C: ] [ D: ] [ E: ] [ F: ]
 :
 :--------------------------------------------------------------
 :   C:\BIN
 :
 :--------------------------------------------------------------
 : ANSI.SYS        EDLIN.COM       PRINT.COM      VDISK.SYS
 : APPEND.COM      EXE2BIN.EXE     PSC.COM        XCOPY.EXE
 : APPLY.COM       FC.EXE          PSCMX80.COM    ZCACHE.SYS
 : ASGNPART.COM    FIND.EXE        RDCPM.COM      ZCOM.EXE
 : ASSIGN.COM      FORMAT.COM      RECOVER.COM    ZSPOOL.COM
 : ATTRIB.EXE      GRAFTABL.COM    REPLACE.EXE
 : BACKUP.COM      GRAPHICS.COM    RESTORE.COM
 : BOOTF.COM       JOIN.EXE        RTCLOCK.COM
 : CHKDSK.COM      LABEL.COM     . SEARCH.COM
 : COMP.COM        LIB.EXE         SHIP.COM
 : CONFIGUR.COM    LINK.EXE        SHIPIT.BAT
 : DEBUG.COM       MODE.COM        SHIPIT.DAT
 : DETECT.COM      MORE.COM        SORT.EXE
 : DISKCOMP.COM    NODEBUG.COM     SUBST.EXE
 : DISKCOPY.COM    NOSTACK.COM     SYS.COM
 : DRIVER.SYS      PART.EXE        SYSCLR.EXE
 : DSKSETUP.COM    PREP.EXE        TREE.COM
 :
 :--------------------------------------------------------------
```

Figure 1 -- MS-DOS Manager Screen

```
************************************************************
```

current disk (drive C in this case) is shown in reverse video on the Disks line. You can always press the F1 key to see help information. If you don't have a mouse, the four menu selections -- File, Options, Disk, and Applications -- are activated by pressing the Alt key. Once the menu selection is activated, you can press the first letter to bring up the appropriate menu: F for File, O for Options, D for Disk, and A for Applications. The File menu shown in Figure 2 was activated by hitting the Alt key and then pressing F.

```
************************************************************


 -----------------------------------------------------------
 |
 | File  Options  Disk  Applications
 |_________________: ] [ C: ] [ D: ] [ E: ] [ F: ]
 | Run...          |
 |________________ |________________________________________
 | Copy...      F4 |
 | Get Info        |
 |________________ |________________________________________
 | Delete...    F5 |DLIN.COM       PREP.EXE        TREE.COM
 | Rename...    F7 |XE2BIN.EXE     PRINT.COM       VDISK.SYS
 | Move...         |C.EXE          PSC.COM         XCOPY.EXE
 | Locate...       |IND.EXE        PSCMX80.COM     ZCACHE.SYS
 |________________ |ORMAT.COM      RDCPM.COM       ZCOM.EXE
 | MS-DOS          |RAFTABL.COM    RECOVER.COM     ZSPOOL.COM
 |________________ |RAPHICS.COM    REPLACE.EXE
 | Exit         F3 |OIN.EXE        RESTORE.COM
 |________________ |ABEL.COM       RTCLOCK.COM
 | COMP.COM        LIB.EXE         SEARCH.COM
 | CONFIGUR.COM    LINK.EXE        SHIP.COM
 | DEBUG.COM       MAN             SHIPIT.BAT
 | DETECT.COM      MODE.COM        SHIPIT.DAT
 | DISKCOMP.COM    MORE.COM        SORT.EXE
 | DISKCOPY.COM    NODEBUG.COM     SUBST.EXE
 | DRIVER.SYS      NOSTACK.COM     SYS.COM
 | DSKSETUP.COM    PART.EXE        SYSCLR.EXE
 |__________________________________________________________


Figure 2 -- MS-DOS Manager File Selection

************************************************************
************************************************************


 -----------------------------------------------------------
 |
 | File  Options  Disk  Applications
 |Disks_________________________ D: ] [ E: ] [ F: ]
 |_____| Split Screen        F9 |___________________________
 |   C:|________________________|
 |_____| Screen Colors...       |___________________________
 | ANSI| File View Options...   |  PREP.EXE        TREE.COM
 | APPE| Delete Options...      |  PRINT.COM       VDISK.SYS
 | APPL| Copy Options...        |  PSC.COM         XCOPY.EXE
 | ASGN| Other Options...       |  PSCMX80.COM     ZCACHE.SYS
 |_ASSI|________________________|  RDCPM.COM       ZCOM.EXE
 | ATTR| Date/Time...           |  RECOVER.COM     ZSPOOL.COM
 | BACK|________________________|  REPLACE.EXE
 | BOOT| Save Options           |  RESTORE.COM
 | CHKD|________________________|  RTCLOCK.COM
 | COMP.COM        LIB.EXE         SEARCH.COM
 | CONFIGUR.COM    LINK.EXE        SHIP.COM
 | DEBUG.COM       MAN             SHIPIT.BAT
 | DETECT.COM      MODE.COM        SHIPIT.DAT
 | DISKCOMP.COM    MORE.COM        SORT.EXE
 | DISKCOPY.COM    NODEBUG.COM     SUBST.EXE
 | DRIVER.SYS      NOSTACK.COM     SYS.COM
 | DSKSETUP.COM    PART.EXE        SYSCLR.EXE
 |__________________________________________________________
Figure 3 -- MS-DOS Manager Options Selection

************************************************************
```

Now that you have a function menu, most of the selections are performed by pressing the first letter (or a function key) for that command function. The appropriate letter is highlighted so that you can see what letter is associated with each command function on the menu, and this is usually called the "point and shoot" technique. To see how this works in practice, let's go through an example of how you would delete a file.

First, select the file to be deleted on the main screen (Figure 1) by moving a reverse video block around with the arrow keys on the keypad. Then, hit Alt to activate the menu, press F for Files, and press D for Delete. Manager asks you if you are sure that's the correct file, and there are two confirmations to be sure you really want to delete that file. All of the other functions work the same way on all of the menus.

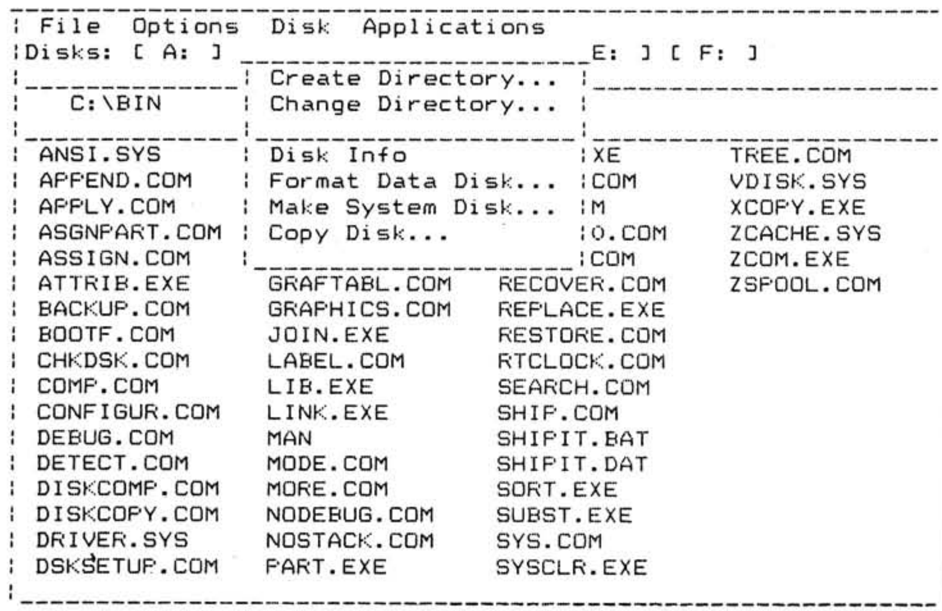
You can also change a number of the Manager display options with the Options menu, and you can even split the screen to look at two drives and/or subdirectories side-by-side. The Options menu is shown in Figure 3.

Although Manager ran just fine on my '248 with a MultiSync monitor, I found that the standard colors chosen for the "highlight" on the eazy pc were difficult to read. I had to use this menu to adjust the screen "colors" (shades of gray) on the eazy pc, and I think Zenith should change the default configuration to make it easier to read.

The Disk selection, shown in Figure 4, allows you to perform various disk-related commands so you don't have to remember the DOS commands for these functions. I thought it was interesting to note that Manager has used the same terminology for system disk and data disk that I have previously mentioned in this column.

The Applications selections are shown in Figure 5. This is a menu that you can customize according to your own requirements using the "Modify Application List" function as shown. You can update the application list to include any program on your disk, and this list includes a complete path specification with the drive letter. I have added WordStar version 4 to the list as shown.

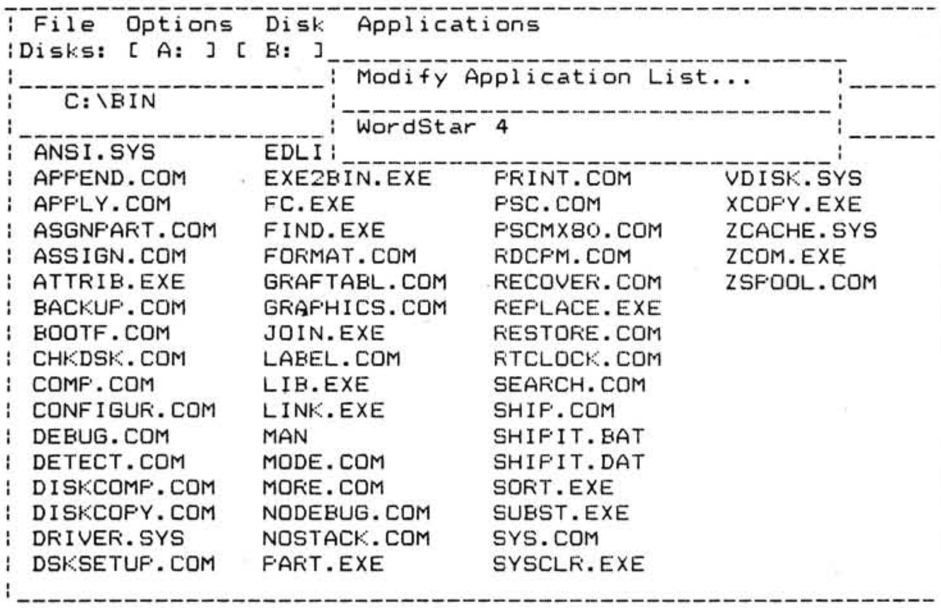
****



```
-------------------------------------------------------------------
¦ File  Options  Disk  Applications
¦Disks: [ A: ] ______________________E: ] [ F: ]
¦______________¦ Create Directory... ¦______________________
¦   C:\BIN     ¦ Change Directory... ¦
¦______________¦_____________________¦______________________
¦ ANSI.SYS     ¦ Disk Info           ¦XE        TREE.COM
¦ APPEND.COM   ¦ Format Data Disk... ¦COM       VDISK.SYS
¦ APPLY.COM    ¦ Make System Disk... ¦M         XCOPY.EXE
¦ ASGNPART.COM ¦ Copy Disk...        ¦O.COM     ZCACHE.SYS
¦ ASSIGN.COM   ¦_____________________¦COM       ZCOM.EXE
¦ ATTRIB.EXE     GRAFTABL.COM   RECOVER.COM     ZSPOOL.COM
¦ BACKUP.COM     GRAPHICS.COM   REPLACE.EXE
¦ BOOTF.COM      JOIN.EXE       RESTORE.COM
¦ CHKDSK.COM     LABEL.COM      RTCLOCK.COM
¦ COMP.COM       LIB.EXE        SEARCH.COM
¦ CONFIGUR.COM   LINK.EXE       SHIP.COM
¦ DEBUG.COM      MAN            SHIPIT.BAT
¦ DETECT.COM     MODE.COM       SHIPIT.DAT
¦ DISKCOMP.COM   MORE.COM       SORT.EXE
¦ DISKCOPY.COM   NODEBUG.COM    SUBST.EXE
¦ DRIVER.SYS     NOSTACK.COM    SYS.COM
¦ DSKSETUP.COM   PART.EXE       SYSCLR.EXE
¦__________________________________________________________________
```

Figure 4 -- MS-DOS Manager Disk Selection

****
****



```
-------------------------------------------------------------------
¦ File  Options  Disk  Applications
¦Disks: [ A: ] [ B: ]__________________________________
¦___________________¦ Modify Application List...     ¦______
¦   C:\BIN          ¦________________________________¦
¦___________________¦ WordStar 4                     ¦______
¦ ANSI.SYS     EDLI ¦________________________________¦
¦ APPEND.COM   EXE2BIN.EXE    PRINT.COM     VDISK.SYS
¦ APPLY.COM    FC.EXE         PSC.COM       XCOPY.EXE
¦ ASGNPART.COM FIND.EXE       PSCMX80.COM   ZCACHE.SYS
¦ ASSIGN.COM   FORMAT.COM     RDCPM.COM     ZCOM.EXE
¦ ATTRIB.EXE   GRAFTABL.COM   RECOVER.COM   ZSPOOL.COM
¦ BACKUP.COM   GRAPHICS.COM   REPLACE.EXE
¦ BOOTF.COM    JOIN.EXE       RESTORE.COM
¦ CHKDSK.COM   LABEL.COM      RTCLOCK.COM
¦ COMP.COM     LIB.EXE        SEARCH.COM
¦ CONFIGUR.COM LINK.EXE       SHIP.COM
¦ DEBUG.COM    MAN            SHIPIT.BAT
¦ DETECT.COM   MODE.COM       SHIPIT.DAT
¦ DISKCOMP.COM MORE.COM       SORT.EXE
¦ DISKCOPY.COM NODEBUG.COM    SUBST.EXE
¦ DRIVER.SYS   NOSTACK.COM    SYS.COM
¦ DSKSETUP.COM PART.EXE       SYSCLR.EXE
¦__________________________________________________________________
```

Figure 5 -- MS-DOS Manager Applications Selection

****

All in all, the MS-DOS Manager is a nice shell. You can run it from the keyboard or use a mouse, but I found it more convenient to run from the keyboard. In many respects, it resembles Windows, but it doesn't have all of the Windows' features such as the notepad, calendar, etc. I found the program to be easy to use, and it is recommended if you are looking for a shell program. Although it comes with the eazy pc, you can also obtain it as a separate program from the Heath catalog.

Comments on the eazy pc

With one exception, the eazy pc has generally received pretty good reviews in the context of its intended market -- beginners and occasional users. The one exception appeared in the September 28, 1987, InfoWorld (page 76). In most cases, I generally agree with the InfoWorld reviews for products that I have worked with, but this particular review is an exception. I found a number of inaccuracies and misunderstandings in this particular review that seem to be a result of some bias on the part of the reviewer. While it is always difficult to be totally objective, I think this particular review completely lost sight of the facts and objectives of the eazy pc.

For example, the review states that "its nonstandard insides mean that service is harder to find for this unit than it is for most PC clones". One could assume this means it is easier to find service for Fred's TurboWhacker pc than it is for the eazy pc. I somehow doubt that since you can get service at Heath stores (70 or so), 600 ZDS dealers, and ComputerLand stores.

This review also criticizes the user expandability of the unit since the eazy pc is difficult to disassemble. Perhaps the reviewer missed the point that this system is not intended for a lot of user upgrades - it is designed for a "plug in and go" concept so the user does not have to be immediately concerned with all of the trivia such as video boards, memory cards, asynchronous port adapters, and other cryptic computer things. You can easily add some of the Zenith options that I mentioned earlier in a couple of minutes because they fit on the back of the system.

It is too bad that the InfoWorld review clearly missed the purpose of the eazy pc. One of the problems pointed out was the fact that they had difficulties loading the memory intensive Multimate Advantage 1.0 program. Their problem was simply that too much memory was required by both Multimate and MS-DOS Manager, and I think it is unfair to criticize a system on that basis. Using that same criteria, I could also add that I might never buy a Chevrolet four-door family sedan because it won't exceed 120 miles per hour. And if you want to get downright technical, you could also criticize the Z-248 or IBM AT for the same reason since they also come with a standard 512 KB of memory. When you exceed the known limitations of a computer system, you should expect that things will not work properly, and the reviewer should have known that.

Is the eazy pc for everyone? Of course not. Is the eazy pc something that all

HUG members will need? Probably not, because many HUG members want or need more capabilities that are available in the other product lines. Not everyone needs a '158, '248, or '386 system, and the eazy pc offers an alternative to these other choices.

From my teaching experience, I would suggest that most beginning users would not even consider a program like Multimate Advantage -- they would be more inclined to use something simple like PC- Write or perhaps even WordStar. In most cases, eazy pc's standard 512 K of memory is quite sufficient for nearly all applications, and I doubt that most users would have any problems with that.

All in all, I think that the eazy pc is a nice little system based on its intended market. It is easy to set up, and it includes all of the documentation that a beginner needs to get the system up and running in very short order.

Upgrading Your '150

With all of the emphasis on newer and faster computers, there are a remarkable number of "accelerator" boards available in the market that can add an 80286 or 80386 CPU to your Z-150 series computer. Based on some information that was published in PC Magazine some time ago, they concluded that many of these upgrades did not work as well as expected. In most cases, they reported that some software was not compatible with these upgrade boards even though it ran just fine on the original system. Different boards had problems with different software, and it wasn't clear why the software problems occurred. If you have any Z-150 series computer, I just received a press release that you should know about.

First Capitol Computer

First Capitol Computer has announced that they will upgrade your 150 series computer to the new Z-286 configuration that appears in the latest Heath catalog. They remove the disk drives and any add-on boards (e.g. memory) from your computer and install them into a new Zenith Z-286 chassis, test it, and ship it back to you. You have, in effect, a brand new 286 computer with your original disk drives. You also get an AT style keyboard, video controller equivalent to the video board installed in your '150, and a replacement 16-bit hard disk controller if you have a hard disk installed. And you can get this upgrade to a 286 configuration for $995 (plus 2% for shipping) which I think is a good deal.

One of my friends has a '151, and he has been considering this upgrade for improved speed and performance. He has two floppy drives and a 20 MB hard disk, and if he decides to get his system upgraded, I will let you know about the results.

There is one potential problem area related to this upgrade that you should know about. Perhaps the most important point is that this upgrade will take you to a Zenith compatible IBM AT style system, and there are a few hazards in that as I have discovered. In many cases, existing PC compatible memory boards will simply not work in a faster system because they weren't designed for the 8 Mhz clock speed. Other add-on boards may also not work in the new system for the same reason. If the existing add-on boards will not work in the new system, First Capitol will give you a trade-in credit on replacement boards that work in the new system.

Another important point is that the keyboard for the 80286 and later systems has significant internal differences from the PC compatible keyboards. In other words, you can't use a PC compatible keyboard with an AT compatible machine and vice-versa. From a technical perspective, the AT keyboards generate a three-byte scan code instead of the PC's two-byte scan code. That's why you'll get a new keyboard with the new system -- the old keyboard will simply not work.

If you decide to upgrade your system, you might let me know how things worked out. If you need any more information about this upgrade, you can contact First Capitol as listed at the end of this article.

Powering Down

There are all kinds of new products available in the computer arena, but I think the most exciting is Borland's new Quattro spreadsheet. In fact, I think it is so exciting that I am writing a separate article about it which will appear next month. When you see all of the things it can do, it is a truly spectacular piece of software. I think its most amazing capability is that it can out-Lotus the 1-2-3 spreadsheet. And you can buy Quattro for about the same price that Lotus wants for their upgrade to a new version. Watch for this article in addition to my regular column.

If you have any questions about anything in this column, be sure to include a self-addressed, stamped envelope (business size preferred) if you would like a personal reply to your question, suggestion or comment.

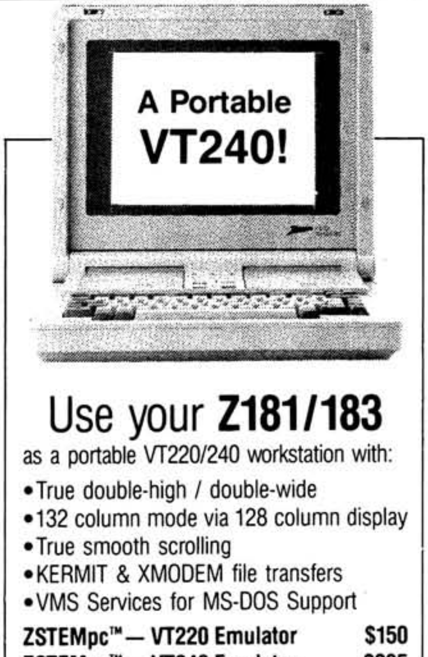
Products Discussed

Z-286 Upgrade $995.00
First Capitol Computer(plus 2% shipping)
#16 Algana Drive
St. Peters, MO 63376
(314) 447-8697

eazy pc

Single floppy drive (EZ-1)	$995.00
Two floppy drives (EZ-2)	1199.00
Floppy/hard disk (EZ-3)	1699.00
128 K Memory Module	99.00
Modem/Memory/Serial Module	399.00

Heath/Zenith Computer Centers
ZDS Dealers
ComputerLand Stores
Zenith Data Systems
1000 Milwaukee Ave.
Glenview IL 60025
(800) 842-9000



The other cats get to sing along!

That's because HEPCAT runs **with** your other programs, not **over** them. HEPCAT (HUG Engineer's and Programmer's CAlculation Tool) is a powerful pop-up calculator for all Heath/Zenith MS-DOS and Z-DOS based computers. Unlike other pop-up calculators, HEPCAT does not stop the currently running program while it is popped up. That means that you can do calculations while your computer is busy with something else. For example:

- While Lotus (tm) is loading a huge spreadsheet, you can check your kid's math homework.
- While Dbase (tm) is sorting a large database, you can add up some grocery prices.
- While your computer is busy compiling one program, you can work on number base conversions needed for another program.

HEPCAT is safe to pop-up during just about any running program — even during disk activity. And HEPCAT has other features the other guys can't touch.

HEPCAT gets along with everyone . . .

HEPCAT supports more video configurations than any other pop-up, and always pops up in the current video mode, rather than forcing the screen into a text mode as other pop-ups do. It also works properly with more programs than any other pop-up. You can pop up HEPCAT over Microsoft Windows (tm) and many other programs that other pop-ups can't work with, and even over some other pop-ups.

HEPCAT works harder . . .

HEPCAT provides a multi-function floating point calculator and a programmer's binary calculator that work together to do more than the basic four (+, -, *, /). The floating point calculator includes the following built-in functions: powers, pi, factorial, square root, sine, arc sine, cosine, arc cosine, tangent, arc tangent, log (natural and base 10), e^X and 10^X. It also includes the following conversions: degrees-radians, radians-degrees, Celsius-Fahrenheit, Fahrenheit-Celsius, centimeters-inches, inches-centimeters, meters-feet, feet-meters, kilometers-miles, miles-kilometers, grams-ounces, ounces-grams, kilograms-pounds, pounds-kilograms, milliliters-fluid ounces, fluid ounces-milliliters, liters-quarts, quarts-liters. The binary calculator works in these number bases: binary, tetral (base 4), octal, split octal, decimal, and hexadecimal; and it supports these operations: MOD, AND, OR, XOR, SHL, SHR.

The HEPCAT floating point calculator supports 8 significant digits and can display numbers four ways: floating point, fixed point, scientific notation, and engineering notation. Numbers are handled internally in BCD format to eliminate binary round off errors in addition and subtraction.

HEPCAT eats less . . .

HEPCAT uses less than 16k of memory — less than any other pop-up calculator that we know of. It also uses less than 16k of disk space, so you don't have to worry about where to put it on a small system. The HEPCAT window uses less screen space, too. It shows you more real information than other pop-up calculator displays, but it doesn't waste space by showing you a keypad layout. You already know what your keypad looks like! HEPCAT is easier to learn, too, with commands that make sense.

If you are tired of pop-ups that can only sing solo, give HEPCAT a try. HEPCAT is available from HUG as part no. 885-3045-37 for $35.00. It works on any Z-100 PC, Z-200 PC, or Z-100 (not PC) system and any version of MS-DOS or Z-DOS.

C__Power Part 11

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

I'm using the editor which is the object of the current program source code being featured in the C__Power series to write this article. I can't think of a better test for an editor than to use it to do what it was designed to do. I'll have to admit that this is my second start at writing this paragraph, the first attempt uncovered some problems with the program that led me to do some extensive troubleshooting before achieving the present version.

I'm really quite pleased with the performance of the program as it stands, but there are some features to be added before calling it complete. My present objective is to complete the editor in C__Power, Part 13. The version being used to write this article will be featured in Part 12 of the C__Power series.

You are probably wondering what is included in this issue if two more articles are needed to finish the program. The source code found in the included listing will provide the reader with an editor capable of data input, character insertion or deletion, line insertion or deletion as well as file I/O. A major enhancement as you can see and, as a consequence, we are adding quite a bit of code to that found in the previous article. Some of the new routines are a bit complex so they will bear a little scrutiny in order to fully understand the logic involved. I will dwell on the more sophisticated code fragments and attempt to explain how each routine works so that you may derive the maximum benefit from this article. The major advantage to be gained from following this series is not the programs themselves (although they are quite useful) but the experience acquired in "hands on" programming. When you get your "sea legs" (read "C legs") you will be creating your own programs with an ease only experience can provide.

Future references to this and succeeding articles will refer to the source code by version number. For instance: Edtxt, version 1.11 will refer to the current listing. Version 1.12 will refer to the source code included in the next article and so on. I am using this terminology in an effort to reduce the possibility of ambiguity and also to lend a bit of charisma to our program! If you don't understand the significance of the numbers you get an "F" in Observation - 101. O.K., O.K., the numbers following the decimal will correspond to the article number, i.e. version 1.11 is listed in C__Power, Part 11.

How do I use this editor? I was hoping you would ask. After invoking the program and choosing option one from the menu, the screen will clear and the cursor will move to row 1, col 1. Text may then be entered as with other editors. The arrow keys will enable cursor movement to any location on the screen WITHIN the text. If a key is struck in error, the offending character may be deleted by moving the cursor under the offender and pressing "delete". The line will be reformed by excluding said miscreant. By the same token, a character may be inserted by moving the cursor as before and pressing the "insert" key when appropriate followed by one or more characters. The new letter or letters will be inserted at the cursor position and the rest of the line will be shifted to the right to make room for those being inserted. Pressing any arrow key will switch off the "insert" mode. Lines may be deleted (the entire line) by placing the cursor on the proper row (any column) and pressing "F8" (function key). Lines may be inserted by placing the cursor as before and pressing "F9". If the cursor is in the middle of a line, the text following the cursor will be printed on the line below after shifting the remaining lines down to provide room. The spaces to the right of the cursor will be erased, leaving room for the text to be inserted. When done with your masterpiece, press "F10" to exit, furnish the requested filename and your work will be saved to disk. Menu option two will give you access to any ASCII text file.

One feature which is essential to a word processor is the ability to "pick up" the line below and append it to the current line. Edtxt ver. 1.12 has this feature and quite a bit more polish but as in any other area, we must progress through this program in a logical manner, a step at a time, in order to acquire a thorough comprehension of the methodology involved. I mentioned it now so as to assure you that we will include the ability to reformat a paragraph with our editor. Other additions will be a printing module accessible from the menu, a very attractive menu, a status indicator (insert/overstrike) and a cursor/line number indicator. The latter will be in reverse video. Included in the source code for version 1.12 will also be routines which you can include in other programs for printing in several modes including blinking, bold (intensified) and underline. As you can see, we are rapidly approaching a very useful program.

Before I get into the actual listing, I want to pass along a shortcut that can be a big help in program development. The following will be a little help to those of you with hard disks, no help if you have insufficient computer RAM but a big help to those who are using two floppy drives with 640k of memory and a version of MS-DOS which includes one of the RAMdisk utilities. It may be called Vdisk or something similar, but will reduce the time needed for code compilation drastically. Create or modify your existing config.sys file to include: a:device = RAMdrive.sys 200. Using MS-DOS version 3.2 with the above residing on the default drive causes a virtual disk to be created with 200k of RAM (consult your MS-DOS manual if using a different version). Of course the 200k is subtracted from your available memory, but when using Turbo C you should have adequate space. Next create a batch file to copy all of the files needed by the compiler for your application to the virtual disk (don't forget the source code you will be working on). Finally, transfer to the virtual disk and read Turbo C into memory. When you get through with the project you are working on be sure to copy the new version back to your floppy or Winchester before powering down. I use this technique whenever I plan to spend more than a few minutes in front of the computer - even when I'm using the hard disk. My batch file looks like this:

```
copy stdio.h d:
copy scrnlib.c d:
copy dos.h d:
copy cpl3.c d:      /* source code for C_Power part 13 */
copy stdarg.h d:
copy c0s.obj d:   /* that;s a zero between the c and s */
copy *.lib d:    /* make sure only needed files are on default disk */
copy fcntl.h d:
d:               /* transfer control to drive d: (virtual disk) */
md\sys            /* create a \sys directory */
copy c:\sys\stat.h d:\sys
c:tc             /* Read Turbo.C into memory */
```

I named the batch file "Turbo.bat" so I simply type "turbo" on the keyboard and I'm ready to utilize the virtual disk which the computer thinks is drive d.

Alright, let's look at the listing to see what is new and (hopefully) interesting. Two more include files are listed, both of which are needed for the file I/O which is new to this version of "Edtxt". Please note the first character inside the brackets which designates the drive where the file "stat.h" is to be found. Be sure the file is where the compiler can find it, otherwise you will be bombarded with error messages (something I'm quite familiar with). Also added is the definition used by "C" for EOF (end of file).

Perhaps we better look at the library file "scrnlib.c" next so we may examine the routines which have been added. The first line of new code deals with one of these routines.

Notice the "scroll_up(row)" and "scroll _dn(row)" functions. They are quite similar to the "cls()" function. By passing row 1 to either of these functions they will scroll, up or down respectively, the entire screen (excluding the 25th line) one line. The only other addition to our screen handling library file is the "prnt(string)" function. This is a first cousin to the "puts" function found in most "C" libraries. The difference is that "prnt (string)" doesn't print a newline character following the string. Its chief reason for existence is in the pursuit of speed. I used it (in lieu of "printf") in some of the functions that required quite a bit of printing to the screen in hopes of expediting things a bit. I'll let you be the judge of the results.

Now let's look at the first "switch(chr)" routine within the "get_ch()" function. Move on down to "case 72:" (up arrow). Here we have added the ability to check for the top of the screen and scroll the text down one line while printing a new line of text if the current line is not the first line (ptr->prev != 0). Look at "case 80:" (down arrow), this is nearly the inverse of the above. It checks for the last row and whether or not the stored line number (ptr->lineno) is less than the total number of lines in the file being worked on. By using these two functions in conjunction with the library routines we just covered, we have added the ability to scroll the screen up or down within the confines of our file. You may say "Elementary dear Watson" but we couldn't get very far without them!

Since the above was so easy, we will look at something a bit more complicated next. The "case 67:" (insert line) routine should satisfy your appetite on that score. Here we check to see if the cursor is on row twenty four. If it is, then cause the screen to scroll up. Then the row is incremented (make up your mind, which is it), "i" is adjusted and finally "insline(ptr->strng,i)" is called. The first order of business within the function, after declaring the calling variables, is to provide temporary storage for two lines along with some integer variables which will be needed as counters. Next, "clean_it(string,pos, numchars)" is called to ensure there is no garbage lurking in the storage area waiting to sabotage our efforts. Now the current line is placed in the first storage area (strcpy (string1,stng)). Using the position passed to it ("j"), "stng" is cleaned out from there to its end, resulting in the current line being chopped off at the cursor position. After storing the line number (lyne=ptr-> lineno) the current row is cleared from the cursor position to the screen border by printing spaces over whatever may have been there, resulting in a clean area for additional text entry. The screen image now matches the memory image down to the cursor.

We have ignored the second part of the current line up to now, even erasing it as if it didn't exist. The next operation will take care of that problem. A copy of the unaltered original line still exists in "string1" and we have the subscript where the break occurred. We'll just copy the second half of the original line into the second storage area previously allocated (string2). The code fragment:

```
for(i=0; i <= 79-pos; ++i,++j)
string2[i]=string1[j];
```

should do it nicely. All that remains is to print the second part of the current line on the next row while clearing any other garbage in the process. We employ "printf()" to take care of this job for us by directing it to print our new string in a space eighty characters wide. Actually, we're not quite done since the screen below the fragmented line must be updated as well as a link provided for the new entity, not to mention performing an update on the line numbers. Each line must contain its own number, and since a line was inserted into this maze, all the numbers following must be incremented. The "update_lineno(val)" function is passed a value which is used to adjust each line number in turn.

The "del_line()" function owes its apparent complexity, in part, to the fact that three different situations must be provided for. The line being deleted may be the first line of text, an intermediate entity or the last line. Each instance dictates a different treatment and when taken individually, each instance is quite simple. The line to be deleted is pointed to by the preceding line as well as the succeeding line (excluding the first and last line provision). All we need to do is make these two pointers point to each other, effectively excluding the line to be deleted. By saving the pointer (fptr=ptr) we can free the memory allocated to the deleted line (free(fptr)), thus making a contribution toward memory management. In the case of first or last line deletion, only one link needs to be provided since in either case one of the pointers (prev/next respectively) is not "looking at" its neighbor. Take a second look at the routine element that handles the deletion of the first line with reference to the manner in which we preserve the integrity of the base pointer.

O.K., I think you are ready for "case 83:", the delete character routine. The first operation is to call "fix_line()" in case the character to be deleted has been appended to an existing line. Next we set "i" equal to the cursor position (col) minus one and finally call the "del_ch()" function. You will notice a striking resemblance between this function and the "insline()" function in the first few lines, that is because we do essentially the same thing, create a storage area for our string to be operated on and then store the line of text up to the cursor position in this newly created territory. A test is performed on the character to see if it is not a zero (VACANT). Here you will encounter what appears to be some redundant code. Just ignore it for now and continue on. The strange line will come into play in "Edtxt version 1.12". Have I got your curiosity piqued? I hope so! At this moment we have stored the line of text, up to the cursor position, in "string1". Now if we increment the string subscript in "stng" (ptr->strng) and copy the remainder into "string1" we will have the original line minus the character in the cursor position. Since that is what we are trying to achieve, let's do it. All that remains to do is copy our newly altered line back into "ptr-

```
/* **************** SCRNLIB.C by John P. Lewis ************ */
/* **************** Created 8/1/87 ********************** */
#define VIDEO 0x10
union REGS regs;

void locate(x,y)    /* positions cursor at row, col */
int x,y;
{
  union REGS regs;
  x-=1;y-=1;  /* enable first = 1 */
  regs.x.ax=(2 << 8)+00; /* set cursor position */
  regs.x.dx=(x << 8)+y;
  regs.h.bh=0; /* video page = 0 */
  int86(VIDEO, &regs, &regs);
}

void cls()
{
  union REGS regs;
  regs.h.ah=7;
  regs.x.cx=(00 << 8) + 00; /* top left of screen */
  regs.x.dx=(23 << 8) + 79; /* bottom right of screen */
  regs.x.bx=(7 << 8) + 00; /* 80 X 25 monochrome */
  regs.h.al=0;              /* does not clear the 25th line */
  int86(VIDEO, &regs, &regs);
  locate(1,1);
}

void clr_dn(row1,row2)
int row1, row2;
{
  union REGS regs;
  row1-=1;row2-=1;  /* enable usage first row = 1 */
  regs.h.ah=7;
  regs.x.cx=(row1 << 8) + 0;
  regs.x.dx=(row2 << 8) + 79;
  regs.x.bx=(7 << 8) + 00;
  regs.h.al=0;
 int86(VIDEO, &regs, &regs);
}

 scroll_up(row)
 int row;
 {
 union REGS regs;
 regs.h.ah=6;
 regs.x.cx=(row-1 << 8) +0;
 regs.x.dx=(23 << 8) + 79;
 regs.x.bx=(7 << 8) + 0;
 regs.h.al=1;
 int86(VIDEO, &regs, &regs);
 }

 scroll_dn(row)
 int  row;
 {
 union REGS regs;
 regs.h.ah=7;
 regs.x.cx=(row-1 << 8) + 0;
 regs.x.dx=(23 << 8) + 79;
 regs.x.bx=(7 << 8) + 0;
 regs.h.al=1;
 int86(VIDEO, &regs, &regs);
 }

prnt(string)          /*  similar to the ''C'' library ''puts'' function */
char *string;
{
while(*string)
putchar(*string++);
}
```

>strng", print it and relocate the cursor. That was almost too easy wasn't it?

The next routine is even simpler and quite familiar by now. The "Case 82:" (ins__ch(ptr-strng,col-1)) splits the current line of text at the cursor position, gets the character to be inserted from the keyboard, appends it to the left side of the line and finally appends the right side. After sending the cursor to column one, the newly modified string is printed in the place of its parent. Lastly, the cursor is relocated under the next character, ready for additional insertions if any.

Any program that purports an ability to manipulate data and to be truly valuable must have a file I/O capability. So, you guessed it, that is our next topic. We will provide a means of reading and writing to the disk as well as file creation. One of the routines was imported from a previous "C__Power" article pertaining to a database program (the file opening and creation mechanism). Here we have an example of the advantages of having (and using) a library of "C" functions. Just what you are building now.

Scan the listing down to "open__file()". This function, although possibly overly sophisticated, is now a standard in my own library of "C" functions. It does the job in every conceivable situation that I can conjure up so I can devote my programming efforts to new algorithms. That, after all, is the "Name of the Game". Let's look it over before going to the actual disk reading and writing procedures (oops! functions). First the user is asked for a filename and then a check is made for the existence of same through the use of the library function "access(filename, 0)". If "stat" is found to be unequal to zero, the file does not exist, the user is so informed and given the opportunity to create one in the write (w) mode. The return of a zero indicates that the file exists and it is then opened in the update (r+) mode.

The "read__file()" function first creates dynamic storage for the characters to be retrieved and then forms a link with the first element before utilizing a "do - while" loop to input characters from the file. A check is made for the existence of either an "EOF" character or a line feed/carriage return combination ('\n') during each retrieval. When a newline character is discovered, it is stripped (ptr->strng[i-1]=VACANT), "i" is reset, more storage is created, a link is formed and the loop is reentered. If an "EOF" character is retrieved, a "loop fall out" is executed.

That brings us to "write__file()", a function which utilizes the previously created "base" pointer to find the beginning of our file in memory. Before continuing, note that the file is closed if already opened so that we can open it in the "w" mode, ensuring that we'll start writing to a file of zero bytes length (the "w" mode truncates an existing file to byte zero). A "for" loop is used to write the memory image of our file to disk, a character at a time. Each byte is examined for the presence of a '\0' which is a signal for the substitution of a newline character ('\n'). Each time the end of a line is reached, the pointer is incremented (ptr=ptr->next) and a test is made to see if the last line has been reached. When "ptr->next=0" the last line is being written so each character is "looked at" before being written to the disk. When a '\0' character is encountered a MS-DOS end of file character is written to disk and the file is closed.

We have reached the point where you have the source code for a useable, if rather clumsy, editor. Please don't regard this as a finished program, there are (at least) two more articles dealing with "Edtxt". Each will add some sorely needed enhancements. As stated previously, I'm using "Edtxt version 1.12" to write this article and I'm pretty fussy about editors.

If you would like to follow this series through to fruition but lack the time and or patience to enter the program listing, I'll furnish the program source code and an executable file on YOUR formatted disk. The disks must be either 5 1/4" 360K or 3 1/2" 720K floppies. Please enclose $5.00 for shipping and handling.

If you are having problems with some facet of the program that you do not understand, write me about your predicament and I will try to help. Be sure to enclose a SASE. You will find my address at the beginning of this article.

Before closing, I'd like to comment on D.C. Shoemaker's "Letter to Hug" in the January ('88) issue of REMark. He very eloquently put into print some of my thoughts about a magazine which I consider an indispensable reference. I even have a computer program (a high speed card catalog emulator) to help me look up useful articles I've found in past issues. I hope you find it to be as helpful as I have.

"C" you soon.

```
/* ************ Edtxt by John P. Lewis ************* */
/* ************ ver 1.11 last update 1/28/88 ************* */
#include <dos.h>          /* header file used with int86 interuppts */
#include <stdio.h>         /* standard header file (I/O) */
#include "scrnlib.c"       /* our own screen handling file */
#include <c:\sys\stat.h>  /* used with file I/O, change drive to suit */
#include <fcntl.h>         /* as above */
#define VIDEO 0x10
#define TRUE 1
#define FALSE 0
#define VACANT 0
#define SPACE 32
#define EOF (-1)          /* conforms to  "C" protocol for end of file */

int flag, init, row, col, i, end, in, skip, lines, flag_long;
char string[81], chr, filename[13], buff[160];    /* global variables */
FILE *fd, *f2;

 typedef struct line
    {                          /* structure declaration used with */
    char strng[82];             /* linked_list */
    struct line *next, *prev;
    int lineno;
    } list;
 list *ptr, *runptr, *base, *last, *nxtptr, *tmptr, *fptr;

 create()
 {
 int i;
 ptr = (list*)malloc(sizeof(list));   /* create storage area for above */
 for(i=0;i <= 79; ++i)              /* clean out the garbage */
 ptr->strng[i]=VACANT;
 }
```

```
go()            /* see commented code in text, this code */
{               /* is similar */
union REGS regs;
regs.h.ah=0;
int86(0x16, &regs, &regs);
}
get_ch()
{                        /* see above */
union scan {
      int c;
      char ch[2];
        } sc;skip=flag=0;
       sc.c=go();
       if(sc.ch[0]==0)
       { chr=sc.ch[1];flag=1; }
       else chr=sc.ch[0];
       if(flag==1)
       {
       switch(chr)   /* further decoding of input done using */
        {            /* switch-case function */
        case 75:if(col > 1) col--;i=col-1;  /* left arrow, move cursor */
        locate(row,col);in=0;break;   /* left, decrement string subscript */
        case 77:if(col < 79) col++;i=col-1; /* right arrow, move cursor */
        locate(row,col);in=0;break;   /* right, increment string subscript */
        case 72:fix_line();if(row > 1) {row--;/* up arrow, move cursor up */
        ptr=ptr->prev;}               /* get new pointer */
        if(ptr->prev != 0 && row==1)   /* check for top of screen */
        {ptr=ptr->prev;scroll_dn(row);
        locate(1,1);prnt(ptr->strng);}      /* print updated screen */
        locate(row,col);in=0;i=col-1;break;
        case 80:fix_line();                    /* down arrow */
        if((row < 24) && ((ptr->lineno) < lines))   /* test for file */
        {row++;ptr=ptr->next;      /* & screen position, get new pointer */
        locate(row,col);in=0;}
        if(row==24 && ptr->lineno < lines)
        {ptr=ptr->next;scroll_up(1);locate(24,1); /* cause a screen */
        prnt(ptr->strng);locate(row,col);}i=col-1;break;  /* scroll */
        case 67:if(row==24) {scroll_up(1);row=23;}
        row+=1;i=col-1;insline(ptr->strng,i);  /* insert line */
        refresh_screen();break;
        case 66:del_line();break;                     /* delete line */
        case 83:fix_line();i=col-1;
        del_ch(ptr->strng,col-1);in=0;break;  /* delete character */
        case 82:in=1;ins_ch(ptr->strng,col-1);break; /* insert char */
        }
       }
         else
          switch(chr)                 /* further decoding of input */
          {
          case 13: locate(row,col);printf("\n\n");/* carriage return */
          col=1;fix_line();if(row==24) clr_dn(row,row);
          if(row < 24) row++;
          if(ptr==last )          /* test for ptr pos in file */
          {i=80;                    /* force loop exit if last line */
          locate(row,col);}   /* increment no of line(s) */
          else
          {skip=TRUE;chr=VACANT;        /* if not last line */
          ptr=ptr->next;if(row==24)    /* get new ptr, skip linker */
          refresh_screen();} break;
```

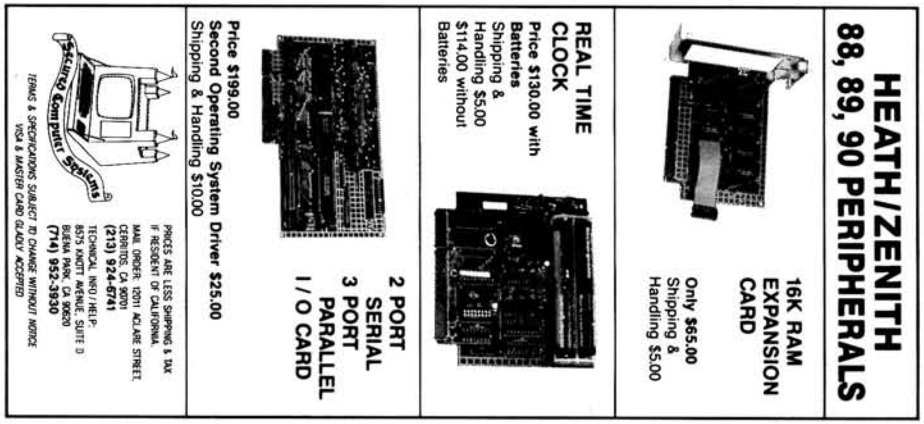
```
          case 8:putchar(8);putchar(32); /* backspace char */
          putchar(8);col--;in=0;         /* move cursor back, print */
          ptr->strng[col-1]=; '' /* space, move cursor back, decrement */
          i-=1;break;        /* string subscript */
          default:i=col-1;       /* must be alpha/numeric character */
          ptr->strng[i]=chr;           /* add character to string */
          putchar(chr);               /* print it to scrn */
          col++;break;
          }
       if(flag==1 && chr != 68) chr=; '' /* null special keys */
       return chr;
      }

first()
 {
 base=ptr;ptr->prev=0;            /* first line, initialize next */
 ptr->next=0;runptr=ptr;
 init=TRUE;last=ptr;lines=1;   /* & prev to 0, create ptr to 1st line */
 ptr->lineno=lines;/* update_screen(); */
}

rest()
{
runptr->next=ptr;ptr->prev=runptr; /* link previous line (runptr) */
ptr->next=0;runptr=ptr;last=ptr; /* to this one (ptr), link prev ptr to */
skip=FALSE;lines++;          /* previous line - see text */
ptr->lineno=lines;
}

insline(stng,pos)
char stng[80];                         /* inserts a new line at */
int pos;                              /* cursor position */
{
int temp, j, i, lyne;
char string1[80], string2[80], dummy[2];fix_line();
clean_it(string1,1,80);clean_it(string2,1,80);
strcpy(string1,stng);j=pos;              /* copy current line to storage */
for( ; j <=79; ++j) stng[j]=VACANT; /* clean out garbage from cursor */
locate(row-1,col);lyne=ptr->lineno;      /* position to end, store line no */
for(j=pos; j <=79; ++j) putchar(32);j=pos;/* erase (on screen) from cursor */
for(i=0; i <=79-pos;++i,++j)             /* store balance of line in */
string2[i]=string1[j];locate(row,1); /* string2 */
/* if(flag_long==1) {string2[i]= '0';
strcat(string2,buff);flag_long=0;} */
printf("%-80s",string2);                 /* move to new line & print it */
if(ptr->next!=0)
{nxtptr=ptr->next;tmptr=ptr;create();    /* establish link with new line */
tmptr->next=ptr;ptr->prev=tmptr;
ptr->next=nxtptr;nxtptr->prev=ptr;}                       /* see text */
else
{ tmptr=ptr;create();tmptr->next=ptr;last=ptr;
ptr->prev=tmptr;ptr->next=0;nxtptr->prev=ptr;}
strcpy(ptr->strng,string2);ptr->lineno=lyne+1;
runptr=ptr;                             /* since we have inserted a */
update_lineno(lyne+2);lines++;col=1;     /* new line, line numbers must be */
}                                      /* updated from this pos to end */

refresh_screen()                          /* prints updated screen from */
{                                      /* current row to line 24 */
```

```
int temp;
tmptr=ptr;temp=row;clr_dn(row,24); /* store pointer and row info, */
while(ptr->next != 0 && row < 25)  /* clear screen from current row dn */
{
locate(row,1);prnt(ptr->strng); /* print new line */
ptr=ptr->next;row++;              /* get pointer to next line */
}if(row < 25 ) {locate(row,1);prnt(ptr->strng);}
ptr=tmptr,row=temp;locate(row,col);    /* restore all values */
}

update_lineno(val)               /* gets line num, increments it, */
int val;                       /* & stores new value */
{
tmptr=ptr;               /* save pointer */
while(ptr->next != 0)
 {
 ptr=ptr->next;ptr->lineno=val;    /* store new line no */
 val++;
 } ptr=tmptr;                  /* restore pointer */
}

del_line()                      /* deletes a line by removing */
{                      /* its link to others & reestablishing link */
int val;                    /* see text */
if(ptr->next==0 && ptr != base)
{fptr=ptr;ptr->prev->next=0;lines--;           /* deleting last line */
ptr=ptr->prev;if(row >1) row--;
last=ptr;ptr->next=0;runptr=ptr;refresh_screen();return;}
if(lines <= 2) { create();first();create();
rest();ptr=base;refresh_screen();return;}
if(base == ptr && lines >2) {fptr=ptr;
ptr=ptr->next;base=ptr;                       /* deleting first line */
refresh_screen();val=ptr->lineno=1;ptr->prev=0;
update_lineno(val+1);lines--;return;}
fptr=ptr;
ptr->prev->next=ptr->next;ptr->next->prev=ptr->prev; /* deleting inter- */
ptr=ptr->prev;row-=1;refresh_screen();val=ptr->lineno; /* mediate lines */
update_lineno(val+1);lines--;free(fptr);     /* free memory */
}

clean_it(string,pos,num)                /* removes garbage from string */
char string[];
int pos, num;
{
int i;pos-=1;
for(i=pos; i < num; ++i)                /* by substituting a zero for */
string[i]=0;                          /* character */
}

del_ch(stng,pos)                   /* deletes a character at cursor pos */
char stng[160];
int pos;                       /* pos = col-1 (i) */
{
int  i, j, len, len1;
char string1[160];clean_it(string1,1,160);
if(stng[0]==VACANT) return;
/* if(stng[0]==SPACE && stng[1]=='\0')*/   /* this code is used in Edtxt */
/* {string1[0]=SPACE;string1[1]='\0'; */    /* version 1.12, commented out */
/* join(string1,pos);goto joinit;}  */   /* for 1.11 */
```

```
for(j=0; j <= pos; ++j)
string1[j]=stng[j];                   /* string1 = ptr->strng up to cursor */
if(stng[j-1]==VACANT && ptr->next ==0)     /* is cursor at end of line? */
return; else if(stng[j-1]==VACANT && ptr->next !=0)
/* {string1[j]='\0';join(string1,pos);goto joinit;}*//* if so, join it */
return;
for(i=pos; i <= 79 ; ++i)            /* if not, get string to right of */
string1[i]=stng[i+1];joinit:   /* cursor & append it to that to the left */
/* if(flag_long==1) add_ch(string1);*/
strcpy(stng,string1);                  /* copy new string into storage */
locate(row,1);printf("%-80s",stng);   /* print new string & erase end of old */
locate(row,col);
}

ins_ch(stng,pos)                   /* inserts character from keyboard */
char stng[80];                    /* into current line at cursor pos */
int pos;
 {
 char string1[80], string[80], c;
 while(in==1)
 {
 strcpy(string1,stng);              /* copy existing string into string1 */
 for(i=0; i < pos ; ++i)
 string[i]=string1[i];              /* truncate string at cursor pos */
 c=get_ch();if(flag!=1 && c!=13 && c!=8) {string[i]=c;i+=1;}/* get char */
 else return;                       /* & test for carriage return */
 for( ; i <= 79; ++i)                  /* append rest of string */
 string[i]=string1[i-1];
 strncpy(stng,string,80);locate(row,1);
 prnt(stng);locate(row,col);              /* print new string */
 }
 }

read_file()           /* reads a file into memory, linking as */
{                    /* it does so and stripping newline chars */
char ch;              /* in the process */
int i;
create();first();i=0;
  do next:
  {
  ch=getc(fd);ptr->strng[i]=ch;++i; /* get char from file & add to list */
  } while(ch != '\n' && ch != EOF);  /* check for newline or end of file */
  ptr->strng[i-1]=VACANT;i=0;        /* zero out newline & reset i */
  if(ch != EOF){ create();rest();goto next;}/* get new storage & establish */
}                                       /* link, get next line if not end */

open_file()                   /* creates a file if filename does */
{                        /* not exist or opens in update mode if it does */
int handle, stat;
char dum[8], ch;
cls();locate(4,4);printf("Please enter filename ");
gets(filename);
stat=access(filename,0);
if(stat != 0)
  {
  locate(6,4);printf("Filename %s does not exist, create? y/n ",filename);
  gets(dum);ch=toupper(dum[0]);if(ch != 'Y') exit();
  handle=open(filename,O_CREAT,S_IWRITE);
  fd=fdopen(handle,"w");
 if(fd==0)
   { locate(6,6);printf("I can't open file, sorry.");exit(); }
 }
 if(stat==0)
   {
   handle=open(filename,O_RDWR,O_TEXT);
   fd=fdopen(handle,"r+");
   }
}

fix_line()                    /* parses current line & substitutes */
{                            /* spaces for zeros when found between */
int i, len;           /* text characters (as when appending existing line) */
for(i=0; i <= 79; ++i)
if(ptr->strng[i]!=VACANT) len=i;
if(len < 1) len=0;if(len > 80) len=0;
for(i=0; i <= len; ++i)
if(ptr->strng[i]==VACANT) ptr->strng[i]=SPACE;  /* ensures line is at least */
}                                      /* one character long (space) */

write_file()                            /* writes linked list to disk */
{
int i;fclose(fd);fd=fopen(filename,"w");ptr=base;
do
{
for( i=0; ptr->strng[i] != '\0'; ++i)
putc(ptr->strng[i],fd);putc('\n',fd); /* writes character from string to */
ptr=ptr->next;                       /* file, appending newline at end */
}while(ptr->next != 0);
for(i=0; i <= 79 ;i++)
{ if(ptr->strng[i] != '\0') putc(ptr->strng[i],fd);}
putc(0x1A,fd);fclose(fd); /* write EOF & close file */
}

build_line()                /* builds a line of text from keyboard input */
{
char c;
int ln;
if(skip==TRUE) goto skipl;
create();         /* create storage for structure (line) */
if(init==0)       /* see if first line */
first();         /* link as first */
else rest();      /* link as any other line */
skipl:
for(i=col-1; i < 80 && ((c=get_ch()) != 68 | | flag != 1 ); ++i);/* exit w/F10 */
 /* above line of code is input loop as well as a trap for exit char */
if(flag == 1 && c == 68)    /* F10 key to act as exit signal */
  {
  fix_line();end=TRUE;     /* exit condition, flag main */
  }
}

main()
{
int lev;
char ch, dummy[2];               /* initialize variables */
while(dummy[0] != '3')
{
cls();lines=0;row=lev=col=1;end=init=FALSE;
```

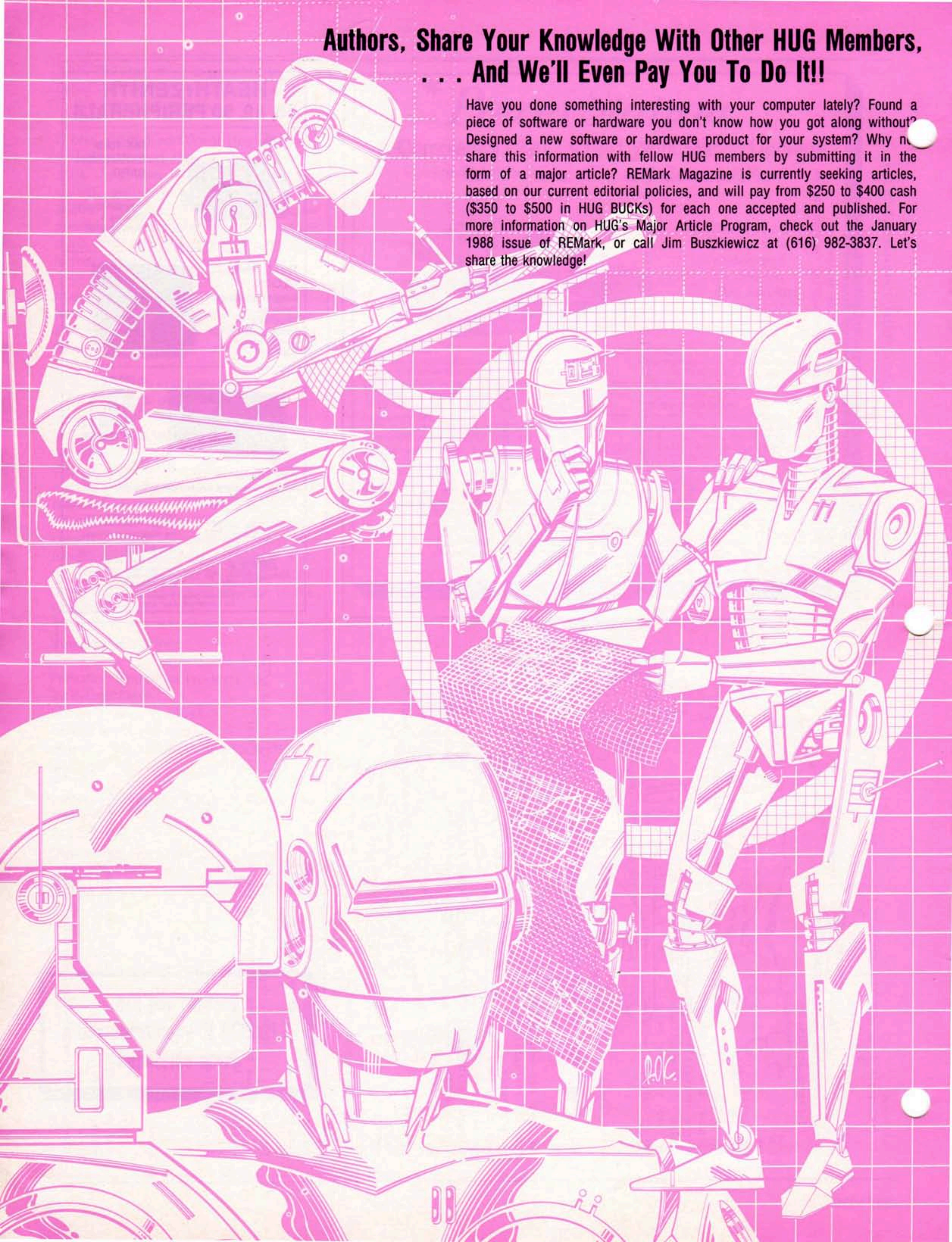
```
locate(2,37);/* print_string(BOLD,"Edtxt");drawbox(2,3,78,24);
drawbox(4,4,76,23);*/locate(6,22);
printf("Please enter...");locate(9,28);printf("1. Create new document.");
locate(11,28);printf("2. Edit existing document.");
locate(13,28);printf("3. Exit to operating system.");
locate(16,22);printf("the number corresponding to your choice ");
gets(dummy);
switch (dummy[0])
 {
 case '1': { cls();
while( end != TRUE)
{
build_line();
}
fix_line();cls();
open_file();write_file();
break; } /* end of case 1 */
case '2': { cls();open_file();read_file();ptr=base;cls();lev=1;
 do
 {
 locate(lev,1);prnt(ptr->strng);ptr=ptr->next;lev++;
 } while(ptr->next != 0 && lev < 25);
 if(lev < 25) {locate(lev,1);prnt(ptr->strng);}
 skip=TRUE;ptr=base;locate(1,1);row=col=1;
 while(end != TRUE)
   build_line();
 write_file();
 break; } /* end of case 2 */
 }
}/* end of while */
}
```

*

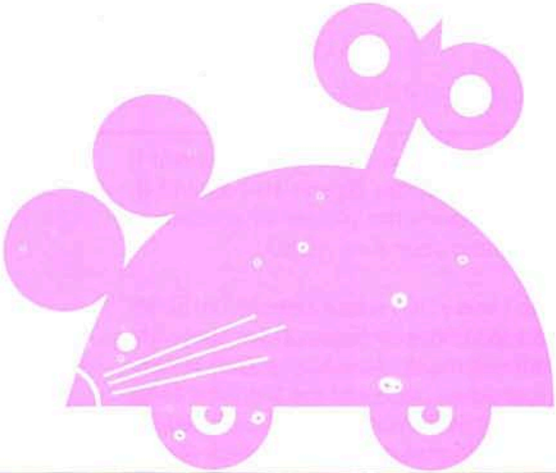


Authors, Share Your Knowledge With Other HUG Members, . . . And We'll Even Pay You To Do It!!

Have you done something interesting with your computer lately? Found a piece of software or hardware you don't know how you got along without? Designed a new software or hardware product for your system? Why n share this information with fellow HUG members by submitting it in the form of a major article? REMark Magazine is currently seeking articles, based on our current editorial policies, and will pay from $250 to $400 cash ($350 to $500 in HUG BUCKs) for each one accepted and published. For more information on HUG's Major Article Program, check out the January 1988 issue of REMark, or call Jim Buszkiewicz at (616) 982-3837. Let's share the knowledge!



Interfacing A C-7 Mouse To The H/Z-100



Robert F. Hassard
3466 Tice Creek Drive, #4
Walnut Creek, CA 94595

I was very disappointed when I received AutoCAD Version 2.18 (for the H/Z-100) to find that it did not support the Logitech C-7 LogiMouse. And I was furious when I discovered that Windows didn't support the C-7 either.

So I set out to learn what could be done about it. After several frustrating months trying to get help from Heath and from Autodesk, with no result whatsoever, and frustrating efforts by Logitech to help me, I finally figured out how to do it. And it is relatively simple.

The C-7 mouse is usually sold bundled with software for the PC. Of course, that software was useless to me, except that the manual for it told me one thing. To use the Mouse with AutoCAD, you must first command the mouse to change to 9600 Baud. All that was necessary was to write a piece of software that would do that on the H/Z-100. Writing the software was easy, but getting it to work was something else.

SUM.COM was the result and it is shown in Figure 1. The procedure is to first configure AutoCAD to use the Logitech R-5 driver, then connect the mouse to Port B (J2), then run SUM, then run ACAD. To make it easier, I prepared a batch file which first asks if the mouse is

Figure 1

```
       page   60,92
       title  Set Up Mouse (SUM.COM)
       subttl by Robert F. Hassard - Sept 1987

;      This program is intended to prepare C-7
; LogiMouse for use with AutoCAD by changing the
; Mouse Baud to 9600 - AutoCAD will reconfigur
; the Port.

PORT    equ    0ECH           ;Port B (J2)

code    segment
       assume  cs:code,ds:code

       org     0100H

start:  xor     al,al          ;First set Port to
       out     PORT[3],al     ;Mouse Default Baud
       in      al,PORT[3]
       mov     al,01001110b    ;1-stop, No parity
       out     PORT[2],al     ;8-bits, 16X rate
       mov     al,01111000b    ;1200 Baud
       out     PORT[2],al
       mov     ch,1
       call    delay          ;allow to settle
       mov     al,00110110b    ;enable transmit
       out     PORT[3],al     ;& maintain voltage
       mov     ch,2
       call    delay          ;more settling
       in      al,PORT        ;clear receiver
       in      al,PORT
       mov     ah,4
       mov     dl,'*'         ;9600 Baud command
       int     21H
       mov     ah,4
       mov     dl,'q'
       int     21H
       mov     ax,4C00H
       int     21h            ;return to MS/DOS
```

```
delay   proc    near
        xor     cl,cl
        mov     si,cx
        mov     ax,2C00H        ;read timer
        int     21H
        mov     bh,dh
        mov     bl,cl
dely1:  mov     ax,2C00H        ;read it again
        int     21H
        cmp     cl,bl           ;and see whether
        jz      dely2           ;its the same Min
        add     dh,60           ;if yes, add 60 Sec
dely2:  sub     dh,bh
        xor     dl,dl
        cmp     dx,si           ;if delay not reached
        jc      dely1           ;go again
        ret                     ;else return
delay   endp

code    ends
        end     start
```

connected, then runs SUM, then asks if I will use my plotter (Epson HI-80), and if yes, commands the plotter to emulate a Roland, and then runs ACAD.

While I was at it, I wrote a version of SUM called WSUM (see Figure 2) for use with Windows, thanks to a tip by Rich Hirsch contained on page 4 of the Summer 1987 edition of SHOWOFF NEWS published by Hogware Company. I enlarged on Rich's idea by first configuring Port B to suit the mouse and then commanding the mouse to emulate a Microsoft mouse. However, there is more to it than this.

The C-7 mouse in emulating the Microsoft mouse fails to do one essential thing. It does not identify itself by emitting an M. That would be asking too much. So, the query in the Windows Mouse Driver must be defeated. Fortunately, that is easy to do. Make a copy of the SETUP disk, then using DEBUG, change just one byte in MOUSE.DRV. At address 07C8, change 74 (JE) to EB (JMP). Now Windows will never know that it is using an alien mouse.

To use the C-7 mouse with windows, run SETUP with the revised MOUSE.DRV, then copy WSUM.COM to the STARTUP disk. To run windows, first be sure that the mouse is connected, then run WSUM, then run WIN.

Figure 2

```
          page    60,92
          title   Set Up Mouse (WSUM.COM)
          subttl  by Robert F. Hassard - Sept 1987

;         This program is intended to prepare C-7
; LogiMouse for use with WINDOWS by changing the
; Mouse format to emulate a Microsoft mouse.
; Also Port is reconfigured to Mouse default Baud.

PORT      equ     0ECH              ;Port B (J2)

code      segment
          assume  cs:code,ds:code

          org     0100H

start:    xor     al,al             ;First set Port to
          out     PORT[3],al        ;Mouse Default Baud
          in      al,PORT[3]
          mov     al,01001110b      ;1-stop, No parity
          out     PORT[2],al        ;8-bits, 16X rate
          mov     al,01111000b      ;1200 Baud
          out     PORT[2],al
          mov     ch,1
          call    delay             ;allow to settle
          mov     al,00110110b      ;enable transmit
          out     PORT[3],al        ;& maintain voltage
          mov     ch,2
          call    delay             ;more settling
          in      al,PORT           ;clear receiver
          in      al,PORT
          mov     ah,4
          mov     dl,'V'            ;Command for Micro-
          int     21H               ;soft emulation
          mov     ax,4C00H
          int     21h               ;return to MS/DOS

delay     proc    near
          xor     cl,cl
          mov     si,cx
          mov     ax,2C00H          ;read timer
          int     21H
          mov     bh,dh
          mov     bl,cl
dely1:    mov     ax,2C00H          ;read it again
          int     21H
          cmp     cl,bl             ;and see whether
          jz      dely2             ;its the same Min
          add     dh,60             ;if yes, add 60 Sec
dely2:    sub     dh,bh
          xor     dl,dl
          cmp     dx,si             ;if delay not reached
          jc      dely1             ;go again
          ret                       ;else return
delay     endp

code      ends
          end     start
```

*

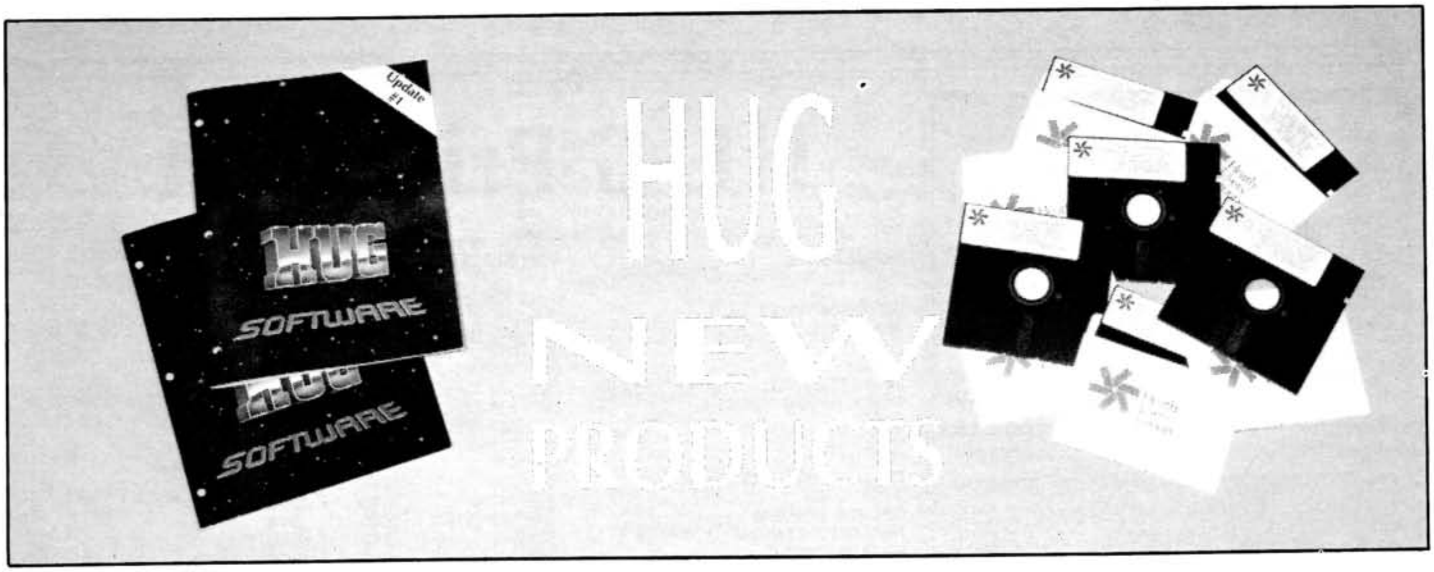


TABLE C
Product Rating

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 – 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/Z-89) 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., 885–1103 or 885–1211[-37] SEA BATTLE.

ORDERING INFORMATION

For VISA and MasterCard phone; telephone Heath/Zenith Users' Group directly at (616) 982-3838. Have the part number(s), description, and quantity ready for quick processing. VISA and MasterCard require minimum $10.00 order. By mail, send your order, plus 10% postage/handling ($1.00 minimum, $5.00 maximum) to: Heath/Zenith Users' Group, P.O. Box 217, Benton Harbor, MI 49022-0217. Orders may be placed, by mail only, using your Heath Revolving Charge account. Purchase orders are also accepted by phone or mail. No C.O.D.s accepted.

Questions or problems regarding HUG software or REMark magazine should be directed to HUG at (616) 982-3463.

NOTES

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).

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

HUG P/N 885-6012-37
ULTRA-RTTY.....................$20

Introduction: This software provides the radio amateur with the means of using a PC compatible computer for transmitting and receiving both RTTY Baudot and ASCII standard codes.

Requirements: This program is designed to run on an IBM (or Heath/Zenith) PC compatible computer running MS-DOS version 2.0 or later. Extensive use is made of the 10 function keys available on the PC. It also requires an RTTY Terminal Interface such as the Heath HD-3030.

Program Author: David E. Warnick

The ULTRA-RTTY disk contains the following files:

RTTY.EXE
RTTY.DOC
RTTY.PRM
RTTY.BAT
RTTYINST.BAT
README.DOC

The following is a list of features as outlined by the author:

- All popular speeds of RTTY Baudot
- Both 110 and 300 baud ASCII
- Transmit Receive Control from the Computer Keyboard
- Split-Screen Display
- User Choice of screen Colors
- User Selection of COM port
- Support of COM1 and COM2
- User Selection of Disk Drive
- Printer Control from the Computer Keyboard
- Save of Received Text to Buffer
- Save of Buffer to Disk
- Transmission of Buffer Contents
- Type Ahead Buffer with Editing Capability
- Preset CQ and 8 Other Messages (Changable from Disk on the fly)
- Instant Send of Above Messages By Function Key
- Send of File From Disk
- Send and Receive Marked on Printed Copy
- All Statuses of the System On Screen
- Limiting on Baudot Transmission to 72-Character Lines (assures compatibility with mechanical teletypes)

Comments: The documentation for this program appears to be excellent. A long awaited program finally here!

Table C Rating: 10

HUG Price List

The following HUG Price List contains a list of all products in the HUG Software Catalog and Software Catalog Update #1. For a detailed abstract of these products, refer to the HUG Software Catalog, Software Catalog Update #1, or previous issues of REMark.

PRODUCT NAME	PART NUMBER	OPERATING SYSTEM	DESCRIPTION	PRICE
		H8 — H/Z-89/90		
ACCOUNTING SYSTEM	885-8047-37	CPM	BUSINESS	20.00
ACTION GAMES	885-1220-[37]	CPM	GAME	20.00
ADVENTURE	885-1010	HDOS	GAME	10.00
ASCRITY	885-1238-[37]	CPM	AMATEUR RADIO	20.00
AUTOFILE (Z80 ONLY)	885-1110	HDOS	DBMS	30.00
BHBASIC SUPPORT PACKAGE	885-1119-[37]	HDOS	UTILITY	20.00
CASTLE	885-8032-[37]	HDOS	ENTERTAINMENT	20.00
CHEAPCALC	885-1131-[37]	HDOS	SPREADSHEET	20.00
CHECKOFF	885-8010	HDOS	CHECKBOOK SOFTWARE	25.00
DEVICE DRIVERS	885-1105	HDOS	UTILITY	20.00
DISK UTILITIES	885-1213-[37]	CPM	UTILITY	20.00
DUNGEONS & DRAGONS	885-1093-[37]	HDOS	GAME	20.00
FLOATING POINT PACKAGE	885-1063	HDOS	UTILITY	18.00
GALACTIC WARRIORS	885-8009-[37]	HDOS	GAME	20.00
GALACTIC WARRIORS	885-8009-[37]	CPM	GAME	20.00
GAMES 1	885-1029-[37]	HDOS	GAMES	18.00
HARD SECTOR SUPPORT PACKAGE	885-1121	HDOS	UTILITY	20.00
HDOS PROGRAMMERS HELPER	885-8017	HDOS	UTILITY	16.00
HOME FINANCE	885-1070	HDOS	BUSINESS	18.00
HUG DISK DUPLICATION UTILITIES	885-1217-[37]	CPM	UTILITY	20.00
HUG SOFTWARE CATALOG	885-4500	VARIOUS	PRODUCTS THRU 1982	9.75
HUGMAN & MOVIE ANIMATION	885-1124	HDOS	ENTERTAINMENT	20.00
INFO. SYSTEM AND TEL. & MAIL SYSTEM	885-1108-[37]	HDOS	DBMS	30.00
LOGBOOK	885-1107-[37]	HDOS	AMATEUR RADIO	30.00
MAPLE	885-8005	HDOS	COMMUNICATION	35.00
MAPLE	885-8012-[37]	CPM	COMMUNICATION	35.00
MICRONET CONNECTION	885-1122-[37]	HDOS	COMMUNICATION	20.00
MISCELLANEOUS UTILITIES	885-1089-[37]	HDOS	UTILITY	20.00
MORSE CODE TRANSCEIVER	885-8016	HDOS	AMATEUR RADIO	20.00
MORSE CODE TRANSCEIVER	885-8031-[37]	CPM	AMATEUR RADIO	20.00
PAGE EDITOR	885-1079-[37]	HDOS	UTILITY	25.00
PROGRAMS FOR PRINTERS	885-1082	HDOS	UTILITY	20.00
REMARK VOL 1 ISSUES 1-13	885-4001	N/A	1978 TO DECEMBER 1980	20.00
RUNOFF	885-1025	HDOS	TEXT PROCESSOR	35.00
SCICALC	885-8027	HDOS	UTILITY	20.00
SMALL BUSINESS PACKAGE	885-1071-[37]	HDOS	BUSINESS	75.00
SMALL-C COMPILER	885-1134	HDOS	LANGUAGE	30.00
SOFT SECTOR SUPPORT PACKAGE	885-1127-[37]	HDOS	UTILITY	20.00
STUDENT'S STATISTICS PACKAGE	885-8021	HDOS	EDUCATION	20.00
SUBMIT (Z80 ONLY)	885-8006	HDOS	UTILITY	20.00
TERM & HTOC	885-1207-[37]	CPM	COMMUNICATION & UTILITY	20.00
TINY BASIC COMPILER	885-1132-[37]	HDOS	LANGUAGE	25.00
TINY PASCAL	885-1086-[37]	HDOS	LANGUAGE	20.00
UDUMP	885-8004	HDOS	UTILITY	35.00
UTILITIES	885-1212-[37]	CPM	UTILITY	20.00
UTILITIES BY PS	885-1126	HDOS	UTILITY	20.00
VARIETY PACKAGE	885-1135-[37]	HDOS	UTILITY & GAMES	20.00
VOLUME I	885-1008	N/A	SOFTWARE LISTINGS	9.00
VOLUME II	885-1013	N/A	SOFTWARE LISTINGS	12.00
VOLUME III	885-1015	N/A	SOFTWARE LISTINGS	9.00
VOLUME IV	885-1037	N/A	SOFTWARE LISTINGS	12.00
WATZMAN ROM SOURCE & DOC	885-1221-[37]	CPM	H19 FIRMWARE	30.00
WATZMAN ROM	885-4600	N/A	H19 FIRMWARE	45.00
WHEW UTILITIES	885-1120-[37]	HDOS	UTILITY	20.00
XMET ROBOT X-ASSEMBLER	885-1229-[37]	CPM	UTILITY	20.00
Z80 ASSEMBLER	885-1078-[37]	HDOS	UTILITY	25.00
Z80 DEBUGGING TOOL (ALDT)	885-1116	HDOS	UTILITY	20.00
		H8 — H/Z-89/90 — H/Z-100 (Not PC)		
ADVENTURE	885-1222-[37]	CPM	GAME	10.00
BASIC-E	885-1215-[37]	CPM	LANGUAGE	20.00
CASSINO GAMES	885-1227-[37]	CPM	GAME	20.00
CHEAPCALC	885-1233-[37]	CPM	SPREADSHEET	20.00
CHECKOFF	885-8011-[37]	CPM	CHECKBOOK SOFTWARE	25.00
COPYDOS	885-1235-37	CPM	UTILITY	20.00
DISK DUMP & EDIT UTILITY	885-1225-[37]	CPM	UTILITY	30.00
DUNGEONS & DRAGONS	885-1209-[37]	CPM	GAMES	20.00
FAST ACTION GAMES	885-1228-[37]	CPM	GAME	20.00
FUN DISK I	885-1236-[37]	CPM	GAMES	20.00
FUN DISK II	885-1248-[37]	CPM	GAMES	35.00
GAMES DISK	885-1206-[37]	CPM	GAMES	20.00
GRADE	885-8036-[37]	CPM	GRADE BOOK	20.00
HRUN	885-1223-[37]	CPM	HDOS EMULATOR	40.00
HUG FILE MANAGER & UTILITIES	885-1246-[37]	CPM	UTILITY	20.00
HUG SOFTWARE CATALOG UPDATE #1	885-4501	VARIOUS	PRODUCTS 1983 THRU 1985	9.75
KEYMAP CPM-80	885-1230-[37]	CPM	UTILITY	20.00
MBASIC PAYROLL	885-1218-[37]	CPM	BUSINESS	60.00
MICRONET CONNECTION	885-1224-[37]	CPM	COMMUNICATION	16.00
NAVPROGSEVEN	885-1219-[37]	CPM	FLIGHT UTILITY	20.00
REMARK VOL 3 ISSUES 24-35	885-4003	N/A	1982	20.00
REMARK VOL 4 ISSUES 36-47	885-4004	N/A	1983	20.00
REMARK VOL 5 ISSUES 48-59	885-4005	N/A	1984	25.00
REMARK VOL 6 ISSUES 60-71	885-4006	N/A	1985	25.00
REMARK VOL 7 ISSUES 72-83	885-4007	N/A	1986	25.00

Make the no-hassle connection with your modem today! ***HUGMCP*** doesn't give you long menus to sift through like some modem packages do. With *HUGMCP, YOU'RE* always in control, not the software. Order ***HUG P/N 885-3033-37*** today, and see if it isn't the easiest-to-use modem software available. Joe Katz says it was so easy to use, he didn't even need to look at the manual. "It's the only modem software that I use, and I'm in charge of both HUG bulletin boards!" says Jim Buszkiewicz. *HUGMCP* runs on ANY Heath/Zenith computer that's capable of running MS-DOS!

HEPCAT is here! *HEPCAT* is here! *HEPCAT* is here! So what is *HEPCAT*, you may ask? Why it's just another Pat Swayne SUPER-UTILITY. *HEPCAT* is an acronym for *HUG Engineer's and Programmer's CAlculation Tool*. Just what we don't need, another memory resident calculator, right? Wrong! With *HEPCAT*, you can throw away the rest and use the best. *HEPCAT* only uses two partial lines on your screen, and best of all, does NOT cause existing programs to stop executing! That means, while your computer is grinding numbers internally, you can be grinding them externally. Order ***HUG P/N 885-3045-37***.

Can't remember how to use the MS-DOS 'COPY' command? Forget the exact command line format for 'ASGNPART'. Too far to go for the MS-DOS manuals on the shelf on the other side of the room? Why not just type 'HELP' on the keyboard? You say it comes back with "Bad command or file name"? It wouldn't if you had HUG's ***HELP*** program. With *HELP* installed on your hard disk, all you need to do is type 'HELP' for a complete list of MS-DOS commands and transients along with a brief explanation of how each command works, as well as the format for its use. *HELP*, ***HUG P/N 885-8040-37***, works on *ALL* Heath/Zenith computers that run MS-DOS!

PRODUCT NAME	PART NUMBER	OPERATING SYSTEM	DESCRIPTION	PRICE
SEA BATTLE	885-1211-[37]	CPM	GAME	20.00
UTILITIES BY PS	885-1226-[37]	CPM	UTILITY	20.00
UTILITIES	885-1237-[37]	CPM	UTILITY	20.00
X-REFERENCE UTILITIES FOR MBASIC	885-1231-[37]	CPM	UTILITY	20.00
ZTERM	885-3003-[37]	CPM	COMMUNICATION	20.00

H/Z-100 (Not PC) Only

ACCOUNTING SYSTEM	885-8048-37	MSDOS	BUSINESS	20.00
CALC	885-8043-37	MSDOS	UTILITY	20.00
CARDCAT	885-3021-37	MSDOS	BUSINESS	20.00
CHEAPCALC	885-3006-37	MSDOS	SPREADSHEET	20.00
CHECKBOOK MANAGER	885-3013-37	MSDOS	BUSINESS	20.00
CP/EMULATOR	885-3007-37	MSDOS	CPM EMULATOR	20.00
DBZ	885-8034-37	MSDOS	DBMS	25.00
ETCHDUMP	885-3005-37	MSDOS	UTILITY	20.00
EZPLOT	885-3023-37	MSDOS	PRINTER PLOTTING UTILITY	20.00
GAMES CONTEST PACKAGE	885-3017-37	MSDOS	GAMES	25.00
GAMES PACKAGE II	885-3044-37	MSDOS	GAMES	25.00
GRAPHICS	885-3031-37	MSDOS	ENTERTAINMENT	20.00
HELPSCREEN	885-3039-37	MSDOS	UTILITY	20.00
HUG BACKGROUND PRINT SPOOLER	885-1247-37	CPM	UTILITY	20.00
KEYMAC	885-3046-37	MSDOS	UTILITY	20.00
KEYMAP	885-3010-37	MSDOS	UTILITY	20.00
KEYMAP CPM-85	885-1245-37	CPM	UTILITY	20.00
MAPLE	885-8023-37	CPM	COMMUNICATION	35.00
MATHFLASH	885-8030-37	MSDOS	EDUCATION	20.00
ORBITS	885-8041-37	MSDOS	EDUCATION	25.00
POKER PARTY	885-8042-37	MSDOS	ENTERTAINMENT	20.00
SCICALC	885-8028-37	MSDOS	UTILITY	20.00
SKYVIEWS	885-3015-37	MSDOS	ASTRONOMY UTILITY	20.00
SMALL-C COMPILER	885-3026-37	MSDOS	LANGUAGE	30.00
SPELL5	885-3035-37	MSDOS	SPELLING CHECKER	20.00
SPREADSHEET CONTEST PACKAGE	885-3017-37	MSDOS	VARIOUS SPREADSHEETS	25.00
TREE-ID	885-3036-37	MSDOS	TREE INDENTIFIER	20.00
USEFUL PROGRAMS I	885-3022-37	MSDOS	UTILITIES	30.00
UTILITIES	885-3008-37	MSDOS	UTILITY	20.00
Z100 WORDSTAR CONNECTION	885-3047-37	MSDOS	UTILITY	20.00
ZBASIC DUNGEONS & DRAGONS	885-3009-37	MSDOS	GAME	20.00
ZBASIC GRAPHIC GAMES	885-3004-37	MSDOS	GAMES	20.00
ZBASIC GAMES	885-3011-37	MSDOS	GAMES	20.00
ZPC II	885-3037-37	MSDOS	PC EMULATOR	60.00
ZPC UPGRADE DISK	885-3042-37	MSDOS	UTILITY	20.00

H/Z-100 And PC Compatibles

ADVENTURE	885-3016-37	MSDOS	GAME	10.00
ASSEMBLY LANGUAGE UTILITIES	885-8046-37	MSDOS	UTILITY	20.00
DEBUG SUPPORT UTILITIES	885-3038-37	MSDOS	UTILITY	20.00
DPATH	885-8039-37	MSDOS	UTILITY	20.00
HADES	885-3040-37	MSDOS	UTILITY	40.00
HELP	885-8040-37	MSDOS	CAI	20.00
HEPCAT	885-3045-37	MSDOS	UTILITY	35.00
HUG BACKGROUND PRINT SPOOLER	885-3029-37	MSDOS	UTILITY	20.00
HUG EDITOR	885-3012-37	MSDOS	TEXT PROCESSOR	20.00
HUG MENU SYSTEM	885-3020-37	MSDOS	UTILITY	20.00
HUG SOFTWARE CATALOG UPDATE #1	885-4501	VARIOUS	PROD 1983 THRU 1985	9.75
HUGMCP	885-3033-37	MSDOS	COMMUNICATION	40.00
HUGPBBS SOURCE LISTING	885-3028-37	MSDOS	COMMUNICATION	60.00
HUGPBBS	885-3027-37	MSDOS	COMMUNICATION	40.00
ICT 8080 TO 8088 TRANSLATOR	885-3024-37	MSDOS	UTILITY	20.00
MATT	885-8045-37	MSDOS	MATRIX UTILITY	20.00
MISCELLANEOUS UTILITIES	885-3025-37	MSDOS	UTILITIES	20.00
REMARK VOL 5 ISSUES 48-59	885-4005	N/A	1984	25.00
REMARK VOL 6 ISSUES 60-71	885-4006	N/A	1985	25.00
REMARK VOL 7 ISSUES 72-83	885-4007	N/A	1986	25.00
REMARK VOL 8 ISSUES 84-95	884-4008	N/A	1987	25.00
SCREEN DUMP	885-3043-37	MSDOS	UTILITY	30.00
UTILITIES II	885-3014-37	MSDOS	UTILITY	20.00

PC Compatibles

ACCOUNTING SYSTEM	885-8049-37	MSDOS	BUSINESS	20.00
CARDCAT	885-6006-37	MSDOS	CATALOGING SYSTEM	20.00
CHEAPCALC	885-6004-37	MSDOS	SPREADSHEET	20.00
CP/EMULATOR II & ZEMULATOR	885-6002-37	MSDOS	CPM & Z100 EMULATORS	20.00
DUNGEONS & DRAGONS	885-6007-37	MSDOS	GAME	20.00
EZPLOT	885-6003-37	MSDOS	PRINTER PLOTTING UTILITY	20.00
GRADE	885-8037-37	MSDOS	GRADE BOOK	20.00
HAM HELP	885-6010-37	MSDOS	AMATEUR RADIO	20.00
KEYMAP	885-6001-37	MSDOS	UTILITY	20.00
LASERWRITER CONNECTION	885-8050-37	MSDOS	PRINTER UTILITY	40.00
PS's PC UTILITIES	885-6011-37	MSDOS	UTILITIES	20.00
SCREEN SAVER PLUS	885-6009-37	MSDOS	UTILITIES	20.00
SKYVIEWS	885-6005-37	MSDOS	ASTRONOMY UTILITY	20.00
TCSPELL	885-8044-37	MSDOS	SPELLING CHECKER	20.00

You've got a screen full of important technical data that would be nearly impossible to memorize, and you already have writer's cramps from the last screen full. With ***SCREENDUMP*** from HUG, you can reproduce a complete video screen on a dot matrix printer, including both text and graphics without having to exit the current program. *SCREENDUMP* supports most of the more popular dot matrix printers, including the newer 24-pin and laser jet models. The latest version of *SCREENDUMP* is ***HUG P/N 885-3043-37.***

"Thank Heaven for *HADES*!" That's what a lot of MS-DOS users are saying when ***HADES*** rescues a file that just got accidentally erased. Erased file recovery is only a small part of the capabilities of this program. *HADES* is *HUG's Absolute Disk Editing System*. Within the realms of MS-DOS, *HADES* allows you to directly edit any part of any disk. Directories, files, file attributes. FATS: nothing can hide from you when you use *HADES*. *HADES* works on *ANY* computer that can run MS-DOS version 2 or greater. Order ***HUG P/N 885-3040-37*** today!

Want to keep your H/Z-100? Want to run a lot of that good PC compatible software out there? Don't want to buy a PC compatible though? Then get ***ZPC II, HUG P/N 885-3037-37***, and the ***ZPC II upgrade disk, HUG P/N 885-3042-37.***

ORDERING INFORMATION

For VISA and MasterCard phone orders, telephone the Heath Users' Group directly at (616) 982-3838. Have the part number(s), descriptions, and quantity ready for quick processing. By mail, send your order, plus 10% postage and handling ($1.00 minimum charge, up to a maximum of $5.00) to: Heath Users' Group, P.O. Box 217, Benton Harbor, MI 49022-0217. VISA and MasterCard require minimum $10.00 order. **No C.O.D.s accepted.**

Questions regarding your subscription? Call Margaret Bacon at (616)982-3463.

Continued from Page 8

Beginner's Column

Dear HUG:

I guess I have always been a day late and a dollar short. I, too, would like to see a beginner's column, but someone beat me to the request.

I am one of the (un)fortunate ones who did not know too much about computers and when I plunked down my $2800, I was told by Heath sales people the Z-100 was the best buy at $1499. Much to my dismay four months later, the price had dropped to $999 and the Z-100 was on the way out. The salesman at the Heath store was gone, and since they always ask "Who's calling?", I feel I always get the "tell 'em I have a customer" answer. I might also add its an interzone call for me to either of the Heath stores in the Detroit area.

I have muddled through learning PeachTree's and PeachCalc by myself. I joined the DHUG group, who as of December, have cancelled all future meetings due to poor attendance. Of course, three years later, I still have found no one who uses PeachText (recommended by my long gone Heath salesman). Even PeachTree moved from Atlanta to Decatur, dropped the 800 number and reduced their office hours from 8 PM to 5 PM.

I subscribed to REMark last March and to this date, I have yet to read a single article I have been able to use. I did attempt to use "SCROLL:" from the July '87 issue, but as far as I got was to put the data on a disk. "Now what do I do?"

I am trying to justify another $19.95 to continue the subscription. I have installed a Scottie board in the Z-100 and purchased ZPC and its update disk, but the one single statement from REMark's Pat Swayne, "I've been through how to do patchen' enough so I ain't goin' to do it again?" So how do I do a patch with that attitude, what do I do with dBASE III+ when it don't print? Where do I go, what do I do?

So here I sit, Z-100, LX 80, modem with ACCESS, PeachText 5000, Lotus 123, Cobol, GW-BASIC, MS-BASIC, Multiplan, WordPerfect, PERKS/8, and no one to turn to. Like the person writing the letter to HUG I referred to at the beginning of this letter, I too am not stupid (I only have one degree), but I sure do feel dumb, so why did I go out last weekend and spend $39.94 + tax for two more "this 'ell show you how to do it" books.

Thanks for listening,

Howard Y. McBee
18675 Birchcrest
Detroit, MI 48221

Beginner's Column: An Outstanding Idea!

Dear HUG:

I just received the Tenth Anniversary issue of REMark, and in the "Buggin' HUG" section I read a letter from a fellow HUGgie titled, "Need a Beginner's Column." Before I continue, let me say that this is an outstanding idea!

I am an H/Z-148 owner and a veteran of such department store computers as the Texas Instruments TI-99/4A, Coleco Adam and Commodore 64. Although I use my computer quite a bit more than the writer of the letter, about 30 hours per week, I am still a beginner. I do not have a graduate degree, but I am an avid computer enthusiast who uses his computer for everything from games to word processing. I, too, am not stupid (I have half of my graduate degree), but the majority of the articles appearing in REMark are a little over my head.

The addition of a beginner's column would not only invite more articles for REMark, but would help many beginning HUGgies, such as myself, to better understand and learn more about Heath/Zenith computers. For example, I am having trouble grasping the basic purpose and uses of a spreadsheet and database. A beginner's column would allow other, more experienced HUGgies to write articles about topics such as these from a more basic standpoint than what is currently printed.

I guess what I'm saying is, "Yes, Mr. Editor, there are other takers for a beginner's column!" I just hope that other HUGgies take the time to write and express their desire for a beginner's column.

Continued on Page 52

Additional Floppies for Z-181

Mark Erbaugh
Microcomputer Enhancement
P.O. Box 282053
Columbus, OH 43228

I recently purchased a Z-181-92 as a second computer (My first is a Heathkit 151, which has given me excellent service for the past two years). This article describes my experience in adding an external floppy disk to the laptop. Installation should be similar for the model 181-93, but I have no experience with that.

An external 5.25" drive can be used to transfer software from that format to 3.5" disks or vice versa. If you already have another PC-compatible computer with 5.25" drives, you may not need an external floppy. A communication program and a null modem cable can be used to transfer files between the two machines. If you are using Zenith's MS-DOS 3.2 on the 5.25" machine, there is a program, ZCOM, that can be used. Although not mentioned in the program's documentation, the only cable connections required are pins 2, 3 and 7 with the leads for pins 2 and 3 crossed. However, this will not allow you to transfer copy-protected programs.

Heath offers an external 5.25" floppy drive for $399 plus a cable which is either $49 (for the 181-93) or $19 (for the 181-92). My local Heathkit store was selling units that they built up from parts for $200. I noticed that they were using the same floppy drive as I had installed in my 151 (Shugart SA-455) and an idea hatched. Some friends of mine had successfully replaced one of the 360K (48 tpi) drives in their IBM XT's with 96 tpi drives and could read and write 720K on a standard DSDD diskette. This was especially useful for backing up the hard disk. I decided to replace my drive B: on the 151 with a 96 tpi model and build the old 48 tpi drive into an external floppy for the laptop.

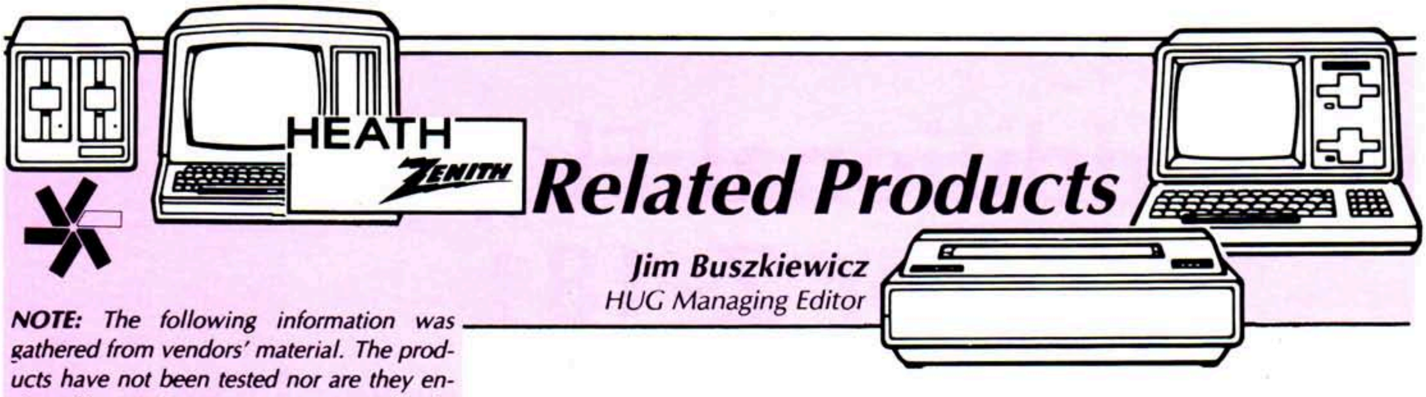
All that is needed to build an external floppy for the 181 is a disk drive, a connecting cable and a power supply for the drive. Most users will probably want a cabinet to house the power supply and drive. This article also assumes that you have Zenith's version of MS-DOS 3.2 or later.

The cable is available from many Heathkit dealers, although it is not a Heath or Zenith part. The cable I bought is from S.W.A.N. and is part MG-6055, "Z-181 External Drive Cable" and it cost $24.95. This is for a 181-92. Users with a 181-93 will need a different cable, which is also available from Heath dealers on special order. The cost is about $49. The cable I purchased had connectors for two drives. More on that below.

Many parts suppliers sell cases with built-in power supplies. I bought the one I used from Princeton Computer Products, Inc. (formerly Floppy Disk Services). It is their part 5SV, item number 5010, and I got it at a computer show special price of $55. The normal catalog price is $65. The unit is designed to hold two half-height drives vertically. I also bought a metal cover for the unused opening.

The drive, as mentioned earlier, I already had. Basically, any IBM PC-compatible 360K (or 48 tpi, it means the same thing) half-height drive should work. Suitable models include MPI-502, Tandon TM-50-2, Panasonic JU-445 (the old Shugart SA-455, taken over by Panasonic), or Teac 55-B. You should be able to find one of these new for around $100 - $150. These drives are also available from Princeton Computer Products.

First the disk drive logic board must be properly configured. Set the drive select to be the third drive. On some drives the first drive is marked drive 0, so you would set the drive select to be 2. On others the first drive is drive 1, so you would set the drive select to be 3. If you have a 181-93, you may have to configure the drive select for the first drive (0 or 1). On most drives, this is done by moving a jumper to a certain pair of pins as indicated on the logic board. These jumpers are usually marked with DS on the circuit board. Consult the data sheet for the drive or your dealer if you need additional information. Also make sure that line terminating resistors are installed on this drive as it will be the only drive on the cable. On some drives, the terminating resistors are in a DIP package, and may be the only DIP in a socket. Most drives come with these resistors installed so you shouldn't have to do anything. If there is an empty DIP socket on the logic board, this probably means the resistor pack has been removed. Again, refer to the dealer or the data sheet for your particular drive if you



Related Products

Jim Buszkiewicz
HUG Managing Editor

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

Information Conversion Services announces the availability of **ViceVersa**TM, a software utility that enables the Xerox Memorywriter to exchange documents bidirectionally with most popular PC word processing packages.

Conversions are performed by inserting a Memorywriter diskette into the PC or via a serial port using the Memorywriter's XMODEM communications feature. ViceVersa currently allows conversions with WordPerfect, MultiMate, DisplayWrite, Wordstar, Samna Word and other similar PC word processing packages.

ViceVersa features an easy-to-use menu-driven interface as well as a command-line interface for use in batch files. The conversion process preserves full document formatting features including: left and right margins, regular and decimal tabs, centering, underlining, boldface, overstrike, soft/hard carriage returns, subscripts and superscripts, indenting, required hyphens and page ends. The Memory writer's extended character set is preserved as closely as possible, allowing ViceVersa to convert foreign language documents.

ViceVersa carries an introductory price of $249.95. Future upgrades will include conversion capabilities between supported PC text processing packages plus conversions between the Memorywriter and other operating systems (UNIX, CTIX, etc.) as well as desktop publishing systems. Conversion solutions between varying disk sizes will also be available in the future.

For complete domestic and international dealer information contact: Information Conversion Services, Inc., 1625 South Fairview, Park Ridge, IL 60068. Phone (312) C-O-N-V-E-R-T.

Central Point Software has announced **PC Tools Deluxe**, an update to its PC TOOLS program. A DOS utility package, PC Tools Deluxe includes a DOS Shell, fast hard disk Backup and Restore, Undelete and Unformat, disk Optimization, safe disk Formatting, disk Caching, disk Verification, and a resident Word Processor with formatted page printing. PC Tools Deluxe includes both 5 1/4" and 3 1/2" diskettes and supports all IBM compatibles. An upgrade offer is being mailed to all registered owners of PC Tools allowing them to purchase PC Tools Deluxe for $15. More information can be obtained from Central Point Software, 9700 SW Capitol Hwy, Portland, Oregon 97219, (503) 244-5782.

The Hogware Company of St. Louis, MO, has introduced several new software utilities for the H/Z-100 computer system.

CGAGRAB is a screen capture program that will capture a graphic screen from a PC-compatible or PC-emulator board. CGAGRAB will capture both 640 x 200 (2color) and 320 x 200 (4-color) graphics so they can be loaded into ShowOff and enhanced with high resolution text, color and graphics.

CONVERT will take a high resolution ShowOff picture and convert it into the Macintosh format (576 x 720) for inclusion in documents in page layout programs (PageMaker, Ventura Publisher, Logitech Publisher), or for printing with PostScript devices (Apple LaserWriter, Linotronic typesetter).

These utilities are on the new ShowOff Art/Utility disk available from Hogware Co. Also included on the disk are ShowOff printer drivers for Epson and Hewlett-Packard laser printers; a patch for Z100 Windows to use the Logitech mouse; ShowMac, for displaying Macintosh pictures on the Z100; and some ShowOff clip art. This disk is available for $15.

ZIP brings to the H/Z-100 a method for creating computer graphics from video sources. ZIP supports the ImageWise Video Digitizer, accepting input from video camera or VCR. Computer images can be based on 3-D objects, photographs, or video tape. ZIP includes laser printer support with true halftone generation. ZIP Image Processing Software for the H/Z-100 is $49, and additional picture disks are $15.

For more information regarding these three products (CGAGRAB, CONVERT, and ZIP), contact the Hogware Co. 470 Belleview, St. Louis, MO 63119, (314) 962-7833.

Now available from **O'Neill Software**, is a product called **"The Text Collector"**. This software allows you to find, collect, examine, and analyze scattered blocks of text material. It searches an entire disk or specified groups of files for any combination of terms, and automatically saves all context blocks meeting the search criteria. Context blocks can be defined by the user or selected from a list which includes sentences, lines, paragraphs, records, and quotes. It also lets you collect material interactively as you browse through one or more files, by searching or tagging the material you want printed or saved.

Designed for programmers, writers, attorneys, and other business professionals, "The Text Collector" runs on IBM PC series or compatible computers with 12k or RAM and MS-DIS 2.0 or later. It is not copy protected and is available for $69 from O'Neill Software, Box 26111, San Francisco, CA 94126, (415) 398-2255.

need additional assistance.

Next, configure the computer. On the bottom of the 181 is a small rubber cover that can be pried off. Removing it reveals a set of five DIP switches. Switch 4 should be off and switch 5 on. This is the factory default setting and indicates two floppy drives (the internal ones). Change switch 4 to on and switch 5 off. This tells the computer that there are 3 drives.

Next, attach the cable from the power supply to the drive. It is keyed and can only be plugged in one way. Then connect the data cable. This cable can be plugged in two ways and only one of them is the correct way. Look on the disk drive circuit board near the connector. You should see numbers for some of the pins. Depending on which side is up, you should see a 1 indicating pin one or a 2 for pin two (odd pins are on one side and even pins on the other). The red stripe on the cable indicates the side of the connector with pin 1. Align the cable so that this side corresponds with the side of the logic board indicated as pin 1 or 2. The drive connector is notched for a key in the connector, but none of the cables that I have seen have the matching key installed. If you have a cable with two drive connectors, it does not matter which one you use as both have identical pins (see below). Use whichever one is most convenient (I prefer to have the unused connector concealed in the cabinet).

Some drive cables, designed for IBM computers, have a twist in the ribbon cable between the two drive connectors. That is, the cable is cut into three sets of conductors and the set in the middle is twisted, while the ends are not. This is done to allow all drives on the cable to use the save drive select setting. If your cable has this twist, you may have to use a different drive select setting.

Plug the other end of the cable into the 181, turn on the power to the external drive, and boot your system as normal. From the DOS prompt you should now be able to access drives A: and B: as before (the internal 720K floppies), but also drive C:. The drive should be able to read standard IBM PC type 360K floppies. You can use it as an additional drive on your system and can use DOS commands to copy files to and from it to your other drives.

You will not be able to use the DISKCOPY command to copy from the 5.25" drive to one of the 3.5" drives as DISKCOPY command requires drives of identical capacity. You should be able to make a copy of a 360K diskette by specifying DISKCOPY C: C:. If you wish to copy an entire disk's contents to a different format, you will have to first FORMAT the destination disk and then use the XCOPY command. Of course, when copying from a 720K to a 360K, make sure that the files you intend to copy will fit on the destination drive.

To copy an entire disk with XCOPY, enter the command XCOPY A:\*.* C: /s/e. This copies all the files in all the root directory and all of its subdirectories (the /s switch). It will also create the needed subdirectories on the destination drive, if necessary. The /e switch will also copy subdirectories if they are empty. Consult your DOS manual for more details on the operation of XCOPY.

I have also used the external drive with COPYIIPC to transfer copy-protected software to 3.5" format. This program copies track by track and makes the first 40 tracks of the destination 720K diskette look like a 360K disk to the computer. Note that this method will not copy any copy-protection that could not be copied by COPYIIPC on standard 5.25" drives. Fortunately, very little of my software is copy-protected.

When you use your 181 as a portable, and do not need the external drive, just disconnect it. You do not need to reset the switches on the bottom of the machine. If you try to access drive C:, the computer will try to access the disk and fail with the infamous "Abort, Retry, Ignore" message, but other than that, no harm will be done.

For those of you who are familiar with some of the internal workings of MS-DOS, you might try, as I did, to implement the external drive using the CONFIG.SYS command DEVICE=DRIVER.SYS/D:1/F:0 without changing the DIP switches. For some reason, this did not work on my system. Accessing drive C: resulted in the drive spinning, but the drive access LED did not light, and no data was read. Maybe someone can explain this.

You may have noticed that I have a cable with TWO drive connectors, a cabinet to house TWO drives, and if you read the 181 documentation, a laptop computer capable of supporting TWO external drives. I also have the curiosity to try TWO external drives, so guess what! Installing a second 360K diskette would allow me to do DISKCOPY C: D: (the fourth drive is D:), but I wanted to see if I could install a 720K (96 tpi) drive. It worked without a hitch!

The drive I had available is a CDC 9429, but any of the PC-compatible 80-track drives should be usable. I merely configured the drive for the fourth drive select position (either 3 or 4), removed the terminating resistors (only one drive on the cable should have them) and configured the 181 for 4 drives (both DIP switches 4 and 5 off). I used a Y adapter to allow the single power lead in the cabinet to connect to both drives.

To recognize the higher capacity of the drive, MS-DOS needs to be modified through the addition of the line DRIVPARM=/D:3 in the CONFIG.SYS file on the boot diskette. The /D parameter (the 3 in this case) tells DOS to reconfigure logical drive 3. "0" refers to drive A:, so 3 refers to drive D:. With no further parameters on the line, DOS defaults to a 720K drive, which is what we want. The DRIVPARM command is documented in Zenith's MS-DOS 3.2 manual, but may not be documented in by other vendors, including IBM and their PC-DOS 3.2. The DRIVPARM command was removed from PC-DOS 3.3.

With the 720K drive, I was able to read and write 720K on standard DSDD diskettes. This can be an alternative to using 3.5" diskettes. I buy DSDD diskettes for 29 cents, but the 3.5" ones cost 99 cents.

The 720K drive will not read or write 360K diskettes. It can read the directory, but if you try to access any of the files, you get the "Sector Not Found" error. The 360K disk can still be read in a standard 360K drive. Similarly, although DOS will allow you to format a disk for 360K by using the /4 option, that disk cannot be read (except for the directory)in a standard 360K drive. It can be read in the 720K drive, though. Friends of mine have been able to read 360K disks in their 720K drives on IBM XTs, but I have been unable to do this with either my laptop or desktop. I will have to assume that this is one of the few incompatibilities between Zenith and true blue.

As an interesting aside, the computer thinks the 5.25" 720K is the same as a 3.5"

720K diskette, so it is possible to do a DISKCOPY from one of the internal drives to the external 720K drive.

If one needed additional storage space, and didn't need the capability to read 360K disks, two 80-tracks could be installed, giving on online capacity of 2880K. I haven't tried it, but it may also be possible to use an external 1.2 megabyte drive, such as those found on an AT to further increase the storage capacity.

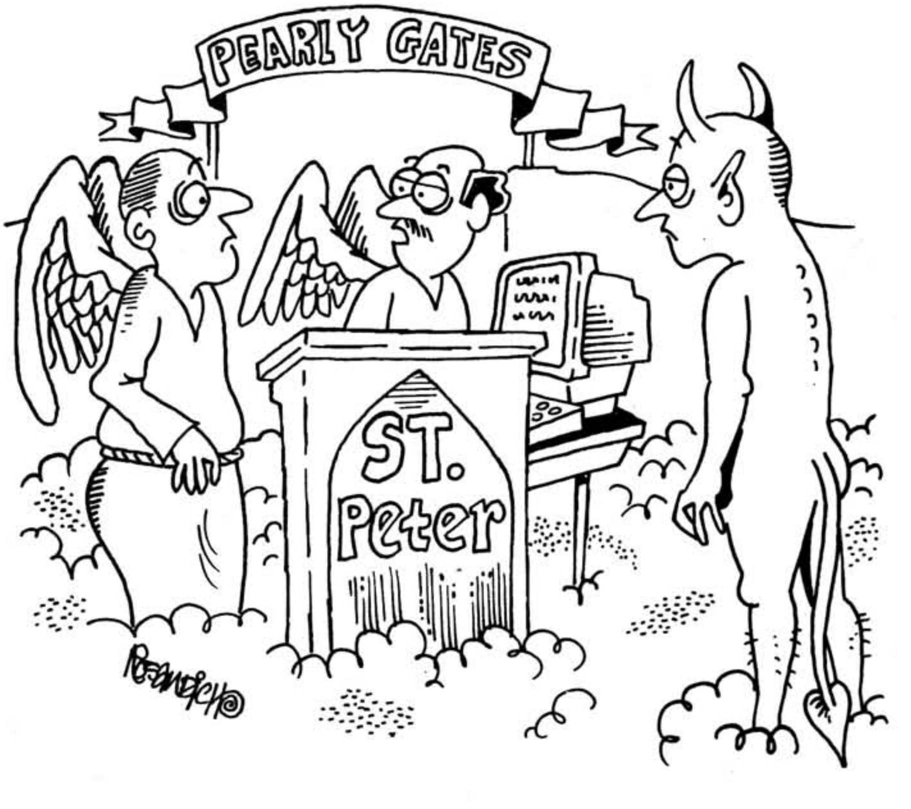
Many vendors are now offering replacement 3.5" drives for desktop machines, and it should be easy to install one of these into the external cabinet, if desired.

In summary, it is very easy to add an external 5.25", 360K drive to the 181. All the pieces can be easily obtained and put together without soldering. My total cost was less than half of Zenith's price for their drive, and I have the capability of adding a second drive. The only change required to the 181 is the setting of a couple of DIP switches, which can be accessed without opening the unit. No modification is required to DOS. It is almost as simple to add an external 720K drive, but this requires an addition to the CONFIG.SYS file.

Vendors Mentioned

Princeton Computer Products, Inc.
39 Everett Drive
Lawrenceville, NJ 08648
orders: (800) 223-0306
others: (609) 799-4440 (use for tech support)

S.W.A.N. Computer Cable & Connector Division
1423 Godwin
Duncanville, TX 75116
(214) 296-6380



"YES, I'D SAY THERE'S BEEN A COMPUTER ERROR!"

ZPC Update #21

Pat Swayne
HUG Software Engineer

This is the twenty first 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 edition of ZPC Update, I will present information and patches for Microsoft Quick C 1.0 and QuickBASIC 4.0, FoxBase Plus, and LogiCADD 3.0.

Quick C and QuickBASIC

A ZPC user reported that Quick C 1.0 and QuickBASIC 4.0 would both partially run under ZPC on a system equipped with a Scottie board. However, Quick C would not compile to memory and QuickBASIC would not compile to disk. Though it may sound incredible, the problems with both programs are caused by the same thing.

When ZPC is active, it attempts to protect the upper 64k of memory in a 768k system so that that upper memory can be used for emulated PC video memory. This protection works fine with most programs, but it hinders a few, including Quick C and the latest version of QuickBASIC. I have written a little program that disables the part of the protection that messes up these programs. If you type in and run the following BASIC program, it will create a program called FIXMEM.COM.

```
10 PRINT  "CREATING FIXMEM.COM
20 OPEN  "0" ,1, "FIXMEM.COM" :L=100
30 FOR I=1 TO 5 :C=0:FOR J=1 TO 10
40 READ B:C=C+B:PRINT #1,CHR$(B);:NEXT J:READ S
50 IF S<>C THEN PRINT  "TYPING ERROR IN LINE" ;L:STOP
60 L=L+10:NEXT I:CLOSE #1
100 DATA 138,22,93,0,184,17,53,205,33,191,936
110 DATA 0,32,185,0,16,176,128,187,252,74,1050
120 DATA 252,242,174,117,15,38,57,29,117,247,1288
130 DATA 128,250,79,116,7,38,198,69,3,67,955
140 DATA 205,32,38,198,69,3,101,205,32,0,883
```

To use FIXMEM, place it on your ZPC system disk, and enter

```
FIXMEM
```

at the system prompt, and press RETURN just before you want to run Quick C or QuickBASIC. When you are finished running them, you can restore normal operation if you enter

```
FIXMEM OFF
```

and press RETURN. For those interested in assembly language, the source code for FIXMEM is listed near the end of this article.

There is another problem involving QuickBASIC 4.0. If you compile a program so that it requires BRUN40.EXE, it will not run with FIXMEM active, and with it off, it will fill the screen with "garbage" characters when it first starts up, but it may run normally after that. Programs compiled to run alone, without BRUN40.EXE, will run without problems.

If you would like to run QuickBASIC 4.0 or Quick C 1.0 under ZPC without a Scottie Board, you can use these patches:

```
QUICKBASIC version 4.0
Insert the disk containing QB.EXE.
QB.EXE
1BFD2,CD,90
1E036,90,90,90,90,90,90,90,90,90,90,90
1E082,90,90,90,90,90,90,90,90,90,90,90
1E2B4,EB,13,90
1E383,90,90,90,90,90
1E396,90
1E3A8,90
1E54D,90
```

```
26B4B,CD,90
2B05E,EB,16
z
QUICK C version 1.0
Insert the disk containing QC.EXE
QC.EXE
F63,90,90,90,90,90,90,90,90,90,90,90
117B3,CD,90
137E3,90,90,90,90,90,90,90,90,90,90,90
1382F,90,90,90,90,90,90,90,90,90,90,90
13A61,EB,13,90
13B30,90,90,90,90,90
13B43,90
13B55,90
13CFE,90
z
```

FoxBase Plus

FoxBase Plus will run under ZPC without any patches or board support, but like Quick C and QuickBASIC, it has high memory problems. If the work it is doing requires too much memory, it will eat into the emulated video memory and may eventually crash. The only way I have found to prevent that from happening is to modify the program to run under the small memory version of ZPC (ZPCSM). The modification involves two things -- a patch, and a small memory resident program. I have worked out the patch portion of the modification for the multi-user version of FoxBase Plus release 2.0 only. Here are the PATCHER lines for the patch:

```
FOXBASE + Multi-user version 2.0
Insert the disk containing MFOXPLUS.EXE
MFOXPLUS.EXE
303E1,CD,A7,90
304B5,CD,A7,90
30585,CD,A7,90
z
```

If you have another version, rename the main EXE file to BIN (FOXPLUS.EXE to FOXPLUS.BIN or whatever), load the file into DEBUG, and search for the hex string 8A,E1,26,89,07. It should be found at two places. At each place, add 2 to the address where it is found and use the A command of DEBUG to change the code there to:

```
INT A7
NOP
```

Then search for the string 26,89,07,83,C3,02 and assemble the above code at the address where it is found. I found the strings in the fourth 64k segment of the program (it is quite large). To work with segments beyond the first segment of large programs using DEBUG, just add 1000 hex to the DS register whenever you want to move up by one segment. Be sure to reference the U and A commands to the DS register, as in

```
ADS:1234
```

The S (search) command always uses the DS register as its reference, so you do not to use the segment prefix with it.

The second part of the modification to run FoxBase Plus under ZPCSM is a memory resident program called RUNFOX.COM which can be created by running the following BASIC program:

```
10 PRINT "CREATING RUNFOX.COM
20 OPEN "O" ,1, "RUNFOX.COM" :L=100
30 FOR I=1 TO 8 :C=0:FOR J=1 TO 12
40 READ B:C=C+B:PRINT #1,CHR$(B);:NEXT J:READ S
50 IF S<>C THEN PRINT "TYPING ERROR IN LINE" ;L:STOP
60 L=L+10:NEXT I:CLOSE #1
100 DATA 235,44,144,80,83,81,82,85,209,235,80,139,1497
110 DATA 195,51,210,187,80,0,247,243,134,196,11,208,1762
120 DATA 183,0,180,2,205,16,88,138,220,185,1,0,1218
130 DATA 180,9,205,16,93,90,89,91,88,207,184,167,1419
140 DATA 37,186,3,1,205,33,186,66,1,180,9,205,1112
150 DATA 33,186,46,1,205,39,13,10,73,110,116,32,864
160 DATA 65,55,32,109,111,100,117,108,101,32,105,110,1045
170 DATA 115,116,97,108,108,101,100,46,13,10,36,0,850
```

RUNFOX should be loaded into memory right after ZPCSM. Like ZPCSM, it will remain in memory until you re-boot your computer. Since you have use ZPCSM instead of ZPC, you may want to make a separate system disk for running FOXBASE if you have a floppy only system. The source code for RUNFOX is at the end of this article.

LogiCADD 3.0

ZPC users have provided patches for to releases of LogiCADD version 3.0. (As you may know, LogiCADD is a release of Generic CADD sold by Logitech.) A patch for the release dated 4-24-87 was provided by Janet Hirsch of Hogware Company:

```
LOGICADD version 3.0 4-24-87
Insert the disk containing CADD.EXE.
CADD.EXE
1C83F,90
1C865,90
1C86A,90
1C89A,90
1C8F5,90
1C8FA,90
1CA61,90
1CB2B,90
1CB78,B0
1CC29,B0
1CC66,B0
1CCCF,B0
1CD20,0,0
1CD88,0,0
1CE03,0,0
z
```

Morrin Hazel, Jr. provided a patch for the release dated 8-31-87:

```
LOGICADD version 3.0 8-31-87
Insert the disk containing CADD.EXE.
CADD.EXE
1B42F,90
1B455,90
1B45A,90
1B48A,90
1B4E5,90
1B4EA,90
1B651,90
1B71B,90
1B768,B0
1B819,B0
1B856,B0
1B8BF,B0
1B910,0,0
1B978,0,0
1B9F3,0,0
z
```

The DOTPLOT program for each of the above releases takes the same patch:

```
LOGICADD DOTPLOT 3.0
Insert the disk containing DPLOT.EXE.
DPLOT.EXE
14BC8,B0
14C79,B0
14CB6,B0
14D1F,B0
14D70,0,0
14DD8,0,0
14E53,0,0
z
```

FIXMEM and RUNFOX Source Code

Here are the assembly source listings for the FIXMEM and RUNFOX programs.

```
;       FIXMEM -- THIS PROGRAM CAUSES ZPC TO NOT ALTER
;       THE DOS MEMORY ALLOCATION CALLS 48 AND 4A.
;       TO USE THIS PROGRAM, ENTER
;
;       FIXMEM TO INSTALL PATCH
;       FIXMEM OFF TO REMOVE PATCH
;
;       BY P. SWAYNE, HUG SOFTWARE ENGINEER 27-JAN-88

CODE    SEGMENT
        ASSUME  CS:CODE,DS:CODE,ES:CODE,SS:CODE
        ORG     5DH
ARG     LABEL   BYTE                        ;ARGUMENT IS HERE
        ORG     100H

START:  MOV     DL,ARG                      ;GET USER;S ARGUMENT
        MOV     AX,3511H
        INT     21H                         ;GET INT 11 VECTOR
        MOV     DI,2000H                    ;START SEARCH HERE
        MOV     CX,1000H                    ;SEARCH THIS MUCH
        MOV     AL,80H                      ;SEARCH FOR THIS
        MOV     BX,4AFCH                    ;AND THIS
        CLD
SLP:    REPNZ   SCASB                       ;LOOK FOR MEM CODE
        JNZ     NOTFND                      ;NOT FOUND
        CMP     ES:[DI],BX                  ;IS THIS THE PLACE?
        JNZ     SLP                         ;NO, KEEP LOOKING
        CMP     DL,'0'                      ;TURN PATCH OFF?
        JZ      POFF                        ;YES
        MOV     ES:BYTE PTR 3[DI],43H       ;ELSE, TURN IT ON
NOTFND: INT     20H                         ;AND EXIT
POFF:   MOV     ES:BYTE PTR 3[DI],65H       ;TURN PATCH OFF
        INT     20H                         ;AND EXIT

CODE    ENDS
        END     START

;       PROGRAM TO SUPPORT FOXBASE UNDER ZPCSM
;       BY P. SWAYNE, HUG SOFTWARE ENGINEER 29-JAN-88

CODE    SEGMENT
        ASSUME  CS:CODE,DS:CODE,ES:CODE,SS:CODE
        ORG     100H

START:  JMP     SETUP               ;SET UP INT A7

;       FOXBASE IS PATCHED TO PRODUCE INT A7 WHEN
;       A WRITE TO VIDEO MEMORY IS REQUIRED.  MEMORY
;       LOCATION IN BX, CHARACTER AND ATTRIBUTE IN AX.

INTA7:  PUSH    AX
        PUSH    BX
        PUSH    CX
        PUSH    DX
        PUSH    BP
        SHR     BX,1                ;DIVIDE LOCATION BY 2
        PUSH    AX
        MOV     AX,BX
        XOR     DX,DX               ;DX:AX = LOCATION
        MOV     BX,80
        DIV     BX                  ;AX = ROW, DX = COLUMN
        XCHG    AL,AH               ;AH = ROW
        OR      DX,AX               ;DX = CURSOR LOCATION
        MOV     BH,0
        MOV     AH,2
```

```
         INT    10H              ;SET CURSOR
         POP    AX               ;GET CHARACTER
         MOV    BL,AH            ;GET ATTRIBUTE
         MOV    CX,1
         MOV    AH,9
         INT    10H              ;WRITE CHARACTER
         POP    BP
         POP    DX
         POP    CX
         POP    BX
         POP    AX
         IRET

         SET UP PROGRAM

SETUP:   MOV    AX,25A7H
         MOV    DX,OFFSET INTA7
         INT    21H              ;SET INT A7
         MOV    DX,OFFSET MSG
         MOV    AH,9
         INT    21H
         MOV    DX,OFFSET SETUP
         INT    27H              ;EXIT, PROGRAM RESIDENT

MSG      DB     13,10,'Int A7 module installed.;,13,10,'$'

CODE     ENDS
         END    START
```

*

Local HUG Club Information Update

They say "all good things must come to an end . . ." and so three more clubs have notified us that they have disbanded. They are: **Blossomland HUG** — St. Joseph, Michigan; **Cleveland HUG** — Cleveland, Ohio; and **East Tennessee Central HUG** — Knoxville, Tennessee.

SUNHUG, Tucson, Arizona asks that you call for meeting time and place. New Vice President — Peter Collins (602) 888-6327.

New address and Contact Person for **CONNHUG**, c/o David Caranci, Meadow Street, Box 39, Litchfield, CT 06759. Phone: (203) 567-5188.

DENHUG, Denver, Colorado has merged with the Aurora DOS Users' Group. By the time you read this they should have a new name established for the club. They meet the second Wednesday each month in the Board Room of the Aurora North Branch Public Library from 7:00 pm to 8:45 pm. Chairman: Mike Reese, (303) 680-1751. BB 300/1200 baud (303) 331-0982. Congratulations and good luck with this merger!

New Contact Person/president for **SMUGH** St. Paul-Minneapolis, Minnesota is Tony Beltran. He may be reached at (612) 789-5501. New club address: 1515 S. 4th Street #E1011, Minneapolis, MN 55454.

New Contact Person for the **Dayton, Ohio HUG** is Keith Greer at (513) 429-1432. Also now meet in AFIT Conference Room, Bldg 640, Room 120.

New Contact Person for **St. Louis HUG** is Shirley Rubenstein. Phone: (314) 291-1850. They meet the 3rd Thursday each month at 7:00 pm at the HEC. New BB# (314) 946-2639.

SPOHUG has changed its name to **SMUG Spokane Microcomputer Users' Group**. New club address: P.O. Box 1753, Spokane, WA 99210. Ron Hodges is their Contact Person at (509) 326-4342. They meet the 3rd Wednesday each month at 7:00 pm at the West Central Community Center.

*

ENABLE Part 5

George P. Elwood
1670 N. Laddie Court
Beavercreek, OH 45432

A Tutorial Intermediate Word Processing

This article starts the second phase of ENABLE's modules, an intermediate look at these capabilities. This is the second of three articles on ENABLE's word processing module, and it will provide an increased awareness of the total word processing capability and the overall capability of ENABLE. Each of these articles will build on what you have learned in the previous articles.

These articles are based on ENABLE version 2.0. They are written toward the Z-100 user, although the information can be used by any users of ENABLE. Keystrokes are presented as Z-100 keystrokes with the PC equivalent in parenthesis. If the keystrokes are shown together, SHIFT/9 or ALT/U, the keys are pressed at the same time. If they are shown separated, F0 F4, they are pressed in the sequence shown, one after another. This same convention will be used throughout the articles, and this is just a review for those who have been following the entire series.

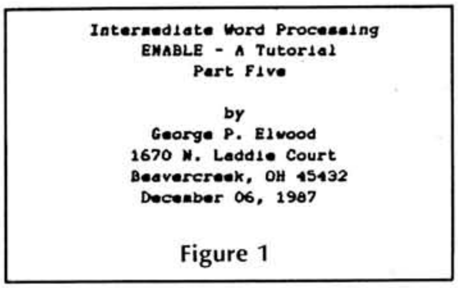
There are currently two books on the market that cover ENABLE. The books are "Using ENABLE" by Charles Spenzzano published by QUE, and "Taking Command of ENABLE" by Yvonne Johnson published by Key Publishing. Of these books, the Spenzzano book is the best. It provides a better understanding of ENABLE and highlights some of the more advanced features. The Software Group offers a series of books for instructors. These books include a student handbook and instructors guide. These books will provide a basic understanding of the program and the capability to instruct others on ENABLE.

If you have enough memory, ENABLE will permit you to move all of a module into memory so that further calls to the disk will not occur. To invoke this capability, change the setup to read ENABLE (,,,,a),vc,m. This will cause ENABLE to load all of the appropriate non-resident modules into memory when it is selected. This will speed operations throughout the program.

One thing I covered in the introduction to word processing needs to be addressed again, as several questions have been raised. If you terminate a document being printed in ENABLE in any way other than SHIFT/F0 F2 (CTRL/F2 for the PC), the print file will remain on the disk and the file name will remain in the file PRINTQ.TSG. When you return to ENABLE and start to print another document, the first document will be printed. You must remove the *.@NN file from the directory or remove the files from the file PRINTQ.TSG. The * indicates the name of the file and the NN represents the number it stands for in the print Q. The print Q feature is only in version two.

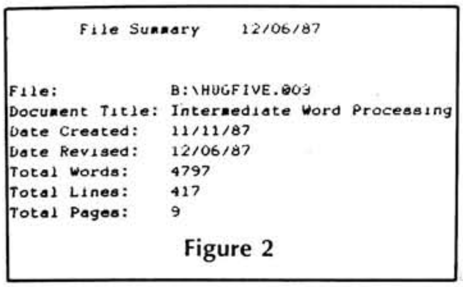
In the introduction to ENABLE's Word Processor, I mentioned in passing the Document title space that is available when you open a document. This space can be as long as you want and will not be printed unless you request it. If it is used, the first line will be displayed as the document title. The information placed in this section will be centered on the first page of the document. I have been using this space for information on documents to include the date it was printed. The document title area will use the system date function "%DATE" any place. I will type in the information I want for the first several lines, centering these lines, and then enter the date. The total lines will also be centered top to bottom on the page. You must select the title page to be printed from the first page of the print menu for it to be printed.

Another optional page that will be printed is the Summary page. This page prints the information displayed on the summary screen that is displayed using F10 (M)CM, (2) Files, (1) File Summary. The document title is taken from the document title entered, first line only. If selected from the first page of the Print Menu, this data will be printed as the last page of the document.



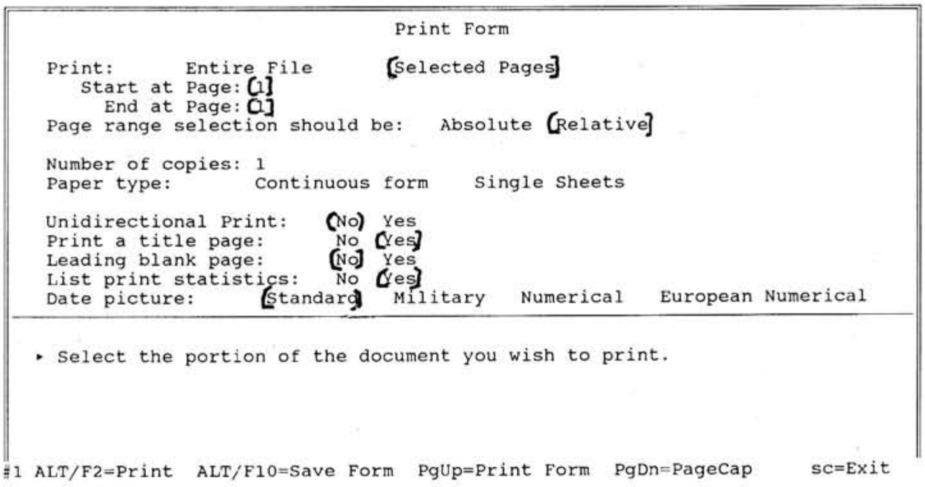
```
                 Intermediate Word Processing
                     ENABLE - A Tutorial
                          Part Five

                              by
                       George P. Elwood
                      1670 N. Laddie Court
                     Beavercreek, OH 45432
                       December 06, 1987
```

Figure 1



```
       File Summary        12/06/87


File:              B:\HUGFIVE.003
Document Title:    Intermediate Word Processing
Date Created:      11/11/87
Date Revised:      12/06/87
Total Words:       4797
Total Lines:       417
Total Pages:       9
```

Figure 2



```
                                  Print Form

Print:        Entire File        [✓ Selected Pages]
   Start at Page: [1]
     End at Page: [1]
Page range selection should be:    Absolute [✓ Relative]

Number of copies: 1
Paper type:         Continuous form    Single Sheets

Unidirectional Print:      [✓ No]  Yes
Print a title page:         No  [✓ Yes]
Leading blank page:        [✓ No]  Yes
List print statistics:      No  [✓ Yes]
Date picture:        [✓ Standard]  Military   Numerical   European Numerical

 ▸ Select the portion of the document you wish to print.




#1 ALT/F2=Print   ALT/F10=Save Form   PgUp=Print Form   PgDn=PageCap      sc=Exit
```

Figure 3

In the introduction to ENABLE's word processing, I touched on the ruler capability of ENABLE. To add a ruler in a document, use "F10 (L)ayout (2) - Default ruler". The same keystrokes are used for the PC version. The expert keystrokes are "F0 F6" for the Z-100 or "ALT/F6" for the PC. This will place the ruler on the line where the cursor was. It will be made a blank line. The default settings are displayed at this time. In addition to the (R)ight/(L)eft margins and (T)ab setting on the line, you can also set numeric places. To do this, type in "N" where you want these displayed.

How many times have you made an outline, made changes and had to go back and change most of the numbers/letters. Or even miscounted the sections and ended up with too many or duplicated sections. ENABLE provides a capability to build outline formats using the rulers. By inserting a ruler and placing the below characters on the line, an outline format can be quickly created. The formats available are "I" for upper case Roman numerals, "A" upper case letters, "1" Arabic numbers, "i" lower case Roman numerals, and "a" lower case letters. To use this capability, you would place the above characters on the ruler line as you would like. You can mix these characters without problems. The line might look like this:

```
Intermediate Word Processing

I     Introduction
        A. Basic functions
II    Books
        A. Spenzzano
        B. Johnson
III   Setup Enhancements
        A. "M"
        B. Printing
           1. Document Title
              a.. "%DATE"
           2. Summary page

--L----II--AA--11--aa------------R---------------

Intermediate Word Processing

#      Introduction
         # Basic functions
#      Books
         # Spenzzano
         # Johnson
#      Setup Enhancements
         # "M"
         # Printing
              # Document Title
                    # "%DATE"
            # Summary page
```

Figure 4
Outline ruler and basic outline the way it appears on screen and printed.

You would then start your outline, tabbing to the location and placing a "#" in that first spot. Each sub area would be tabbed to the location and another "#" would be placed there. In the draft form the "#" would show but when printed or displayed in the final form, the requested character would be displayed. If you make changes, the characters will change without you having to go in and make the changes manually. In fact, you should not make changes to the lettering/numbering in the final mode as errors will result.

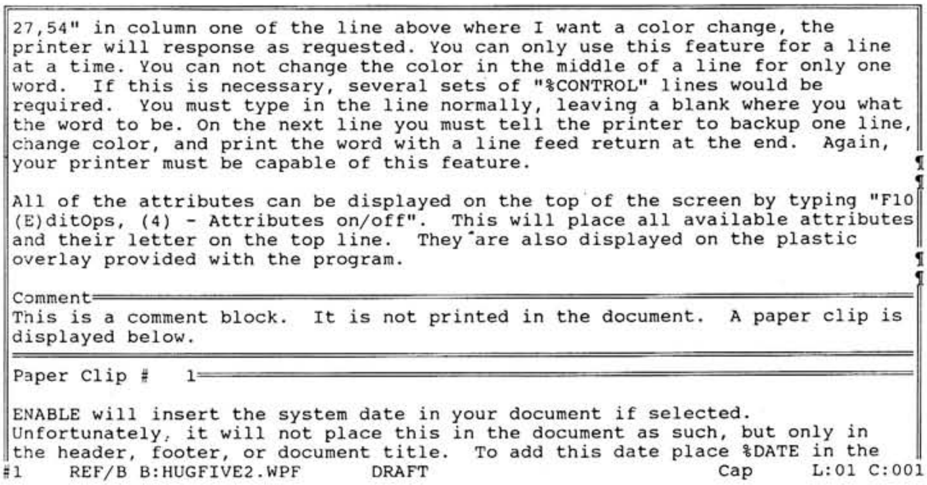
If you are like me, you transpose letters because your fingers get ahead of you. ENABLE will permit you to make the transpose correct with out rekeying the letters. To use this feature, move the cursor after the two letters and press SHIFT/ F0 T (CTRL T for the PC). ENABLE will transpose the previous two letters.

Having spent 20 years in the U.S. Air Force, I have had to write many messages. At times the message would come back from the typist in regular format, upper and lower case letters. The OCR requires that all message be in upper case letters. ENABLE has a feature that will permit the conversion of letters to all upper case or all lower case. To accomplish this, select the attribute "F0 F" (ALT/F for the PC) for all upper case or "F0 K"(ALT/K for the PC). This feature is displayed on the bottom of the screen, note the "F" is displayed. Then move the cursor to the start of the paragraph and press "SHIFT/F0 P" (CTRL/P for the PC) and all characters in the paragraph will be CONVERTED TO UPPER CASE. This capability would have saved much time if it had been available. This is only available in version two, so those of you who have the DoD version 1.15 will have to wait.

Like most word processors, ENABLE will permit you to have automatic reformatting of paragraphs. When selected, the option, "REF", is displayed on the bottom line on the left side between the window number and file name. With this selected, deleting words will cause the remainder of the word to move up if space permits. If you select a block of text that includes paragraph breaks and blank lines, using the paragraph reformat will remove all of these features. To select this feature, type "F10 (E)ditOps, (d) Reformat Paragraph". You must have a section of text blocked before using this option. If you don't have anything blocked, ENABLE will BEEP and indicate this on the bottom line.

Many times when you are writing a document you would like to make a note to yourself about a section of text. With ENABLE you can insert non-printing comments any place in the text. The comment space can be inserted using the menus, "F10 (E)ditOps (f) Insert Comment" or with the expert commands "F9 ICHR N C" (F9 INS N C for the PC). An area will be provided for your comment. The area will expand to meet your needs and will only be displayed on the screen. These display comments can be turned off by using the expert command "F9 O N C". The comments are still there but are not displayed. The expert command is a toggle and can be used to turn the comments back on. To delete the comments, place the cursor on the line with the word "COMMENT" and press SHIFT/DEL LINE or "F0 F3" (ALT/F3 for the PC). This deletes the complete entry.

If you work with large documents, marking text with paper clips is a means to move around. In hard copy documents, paper clips indicate areas that need attention for one reason or another. ENABLE has electronic paper clips which makes moving around a large document easy. WordStar has a similar feature which permits moving to sections of text. Like comments, paper clips are displayed but are not printed. You can add as many as you want and can number them so they can be found easily. Like comments, paper clips can be added from both the menu or using expert commands. Using the menu, "F10 (E)ditOps (f) Insert Paper



```
27,54" in column one of the line above where I want a color change, the
printer will response as requested. You can only use this feature for a line
at a time. You can not change the color in the middle of a line for only one
word.  If this is necessary, several sets of "%CONTROL" lines would be
required.  You must type in the line normally, leaving a blank where you what
the word to be. On the next line you must tell the printer to backup one line,
change color, and print the word with a line feed return at the end.  Again,
your printer must be capable of this feature.                                  ¶
                                                                               ¶
All of the attributes can be displayed on the top of the screen by typing "F10
(E)ditOps, (4) - Attributes on/off".  This will place all available attributes
and their letter on the top line.  They are also displayed on the plastic
overlay provided with the program.                                             ¶
                                                                               ¶
Comment══════════════════════════════════════════════════════════════════════
This is a comment block.  It is not printed in the document.  A paper clip is
displayed below.
═════════════════════════════════════════════════════════════════════════════
Paper Clip #   1═════════════════════════════════════════════════════════════

ENABLE will insert the system date in your document if selected.
Unfortunately, it will not place this in the document as such, but only in
the header, footer, or document title.  To add this date place %DATE in the
#1    REF/B B:HUGFIVE2.WPF       DRAFT                           Cap     L:01 C:001
```

Figure 5 - Comment line within text.

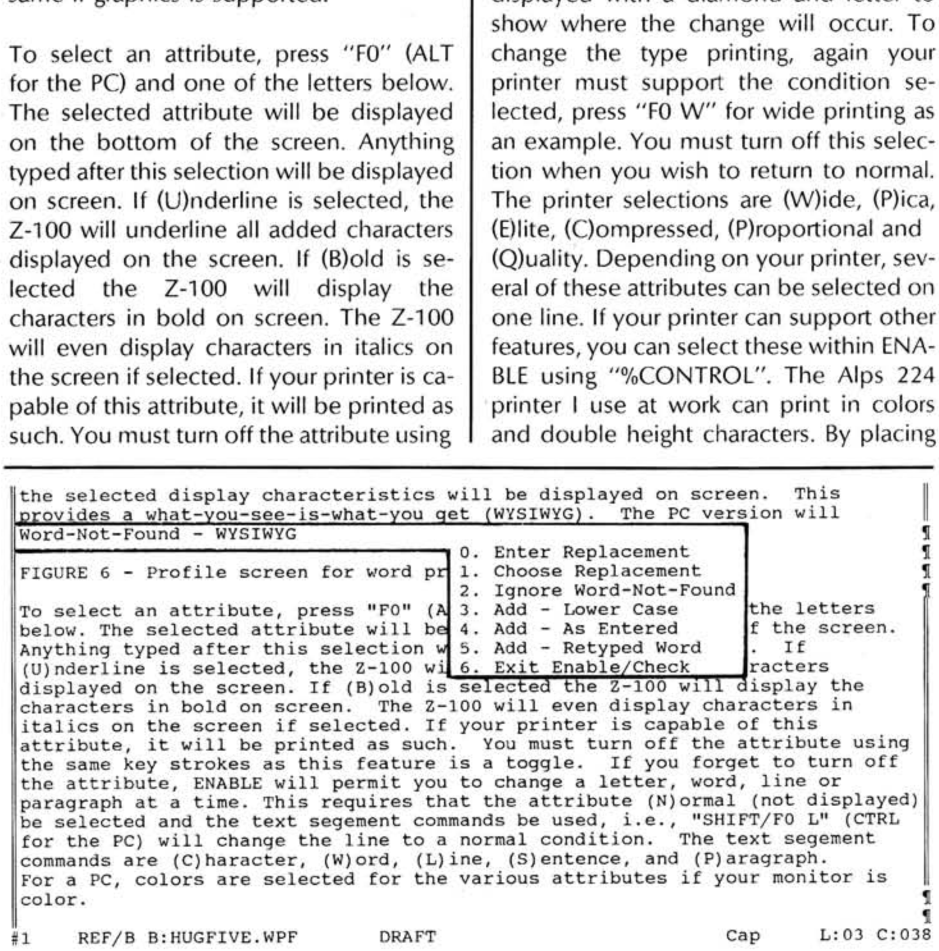
clips" is selected. You are then prompted for a number to be associated with the clip. You then place the cursor above where you want the clip and press <RETURN>. Using the expert commands press "F9 ICHR M C" and ENABLE will prompt for the number to be associated with the clip. To move to any paper clip press "F2 n M C". The F2 key is used with movement to special points in the document and are listed under cursor control commands. The "M C" keys are used with all paper clip operations using the expert commands.

I have mentioned attributes in the word processor several times. These attributes consist of underline, subscript, superscript, bold, italics and combinations of these characteristics. When you developed the profile in the first article, the display was selected. If you want, you can return to the profile and change these characteristics to meet your needs. With the Z-100, the selected display characteristics will be displayed on screen. This provides a what-you-see-is-what-you-get (WYSIWYG). The PC version will display the same if graphics is supported.

To select an attribute, press "F0" (ALT for the PC) and one of the letters below. The selected attribute will be displayed on the bottom of the screen. Anything typed after this selection will be displayed on screen. If (U)nderline is selected, the Z-100 will underline all added characters displayed on the screen. If (B)old is selected the Z-100 will display the characters in bold on screen. The Z-100 will even display characters in italics on the screen if selected. If your printer is capable of this attribute, it will be printed as such. You must turn off the attribute using the same key strokes, as this feature is a toggle. If you forget to turn off the attribute, ENABLE will permit you to change a letter, word, line or paragraph at a time. This requires that the attribute (N)ormal (not displayed) be selected and the text segment commands be used, i.e., "SHIFT/F0 L" (CTRL for the PC) will change the line to a normal condition. The text segment commands are (C)haracter, (W)ord, (L)ine, (S)entence, and (P)aragraph. For a PC, colors are selected for the various attributes if your monitor is color.

Attribute	**Letter**
Underline	u
Bold	b
Italics	i
Subscript	v
Superscript	a
Normal	n
Upper Case	k
Lower Case	f
Double-strike	d
Marked text	m

Part of the attributes relate directly to printer applications. These attributes are displayed with a diamond and letter to show where the change will occur. To change the type printing, again your printer must support the condition selected, press "F0 W" for wide printing as an example. You must turn off this selection when you wish to return to normal. The printer selections are (W)ide, (P)ica, (E)lite, (C)ompressed, (P)roportional and (Q)uality. Depending on your printer, several of these attributes can be selected on one line. If your printer can support other features, you can select these within ENABLE using "%CONTROL". The Alps 224 printer I use at work can print in colors and double height characters. By placing "%CONTROL 27,54" in column one of the line above where I want a color change, the printer will respond as requested. You can only use this feature for a line at a time. You can not change the color in the middle of a line for only one word. If this is necessary, several sets of "%CONTROL" lines would be required. You must type in the line normally, leaving a blank where you want the word to be. On the next line you must tell the printer to backup one line, change color, and print the word with a line feed return at the end. Again, your printer must be capable of this feature.



```
the selected display characteristics will be displayed on screen.  This
provides a what-you-see-is-what-you get (WYSIWYG).  The PC version will
Word-Not-Found - WYSIWYG                                                ¶
                                 0. Enter Replacement                   ¶
FIGURE 6 - Profile screen for word pr 1. Choose Replacement             ¶
                                 2. Ignore Word-Not-Found               ¶
To select an attribute, press "F0" (A 3. Add - Lower Case    the letters
below. The selected attribute will be 4. Add - As Entered    f the screen.
Anything typed after this selection w 5. Add - Retyped Word  .  If
(U)nderline is selected, the Z-100 wi 6. Exit Enable/Check   racters
displayed on the screen. If (B)old is selected the Z-100 will display the
characters in bold on screen.  The Z-100 will even display characters in
italics on the screen if selected. If your printer is capable of this
attribute, it will be printed as such.  You must turn off the attribute using
the same key strokes as this feature is a toggle.  If you forget to turn off
the attribute, ENABLE will permit you to change a letter, word, line or
paragraph at a time. This requires that the attribute (N)ormal (not displayed)
be selected and the text segement commands be used, i.e., "SHIFT/F0 L" (CTRL
for the PC) will change the line to a normal condition.  The text segement
commands are (C)haracter, (W)ord, (L)ine, (S)entence, and (P)aragraph.
For a PC, colors are selected for the various attributes if your monitor is
color.                                                                  ¶
                                                                        ¶
#1    REF/B B:HUGFIVE.WPF        DRAFT                    Cap    L:03 C:038
```

FIGURE 6 - Profile screen for word processor display.

All of the attributes can be displayed on the top of the screen by typing "F10 (E)ditOps, (4) - Attributes on/off". This will place all available attributes and their letter on the top line. They are also displayed on the plastic overlay provided with the program.

ENABLE will insert the system date in your document if selected. Unfortunately, it will not place this in the document as such, but only in the header, footer, or document title. To add this date, place %DATE in the location you want the date printed. I use this feature in the document title as a way to know the most current hard copy of the document. The printed date is in the format selected on the second print screen menu. The date picture can be standard, December 1, 1987; military, 1 December 1987; numerical, 12/1/87; or European, 1/12/87. Using the numerical selection also prompts for the separators, SLASHES, HYPHENS, or DOTS.

Like other word processors, ENABLE has block commands. Using these commands, you can copy, move or delete parts of the document. To mark a section of text for the block operations, place the cursor at the start of the block and press the "F7" key. Move the cursor to the end of the block of text and press the "F7" key again. The selected block will be highlighted.

To mark a block in column form, move the cursor under the first character in the upper left corner and press "SHIFT/F7". Move the cursor to the lower right corner and again press "SHIFT/F7". The area between the two points should be highlighted and can be used in any of the block commands.

After establishing a block using one of the two methods listed above, you can move, delete, or copy it. The command to delete the block is "F9 D CHR B" or "F9 DEL B" for the PC. This is a quick method for deleting large areas within a document. If you delete the text and then decide that you really didn't want it removed, it can be restored using the UNDO commands. These are "F10 (D)elete (0) Undo" or expert command "F9 U". To move the block, place the cursor in the new location for the text and press "F0 F8" (ALT /F8 for the PC). You can place the cursor any place in the text, even in the middle of a line. You may have to do some minor editing if the block you moved had a <RETURN> hard paragraph break in it. To make a copy of the block, place the cursor where you want the copy to be placed and press "F8". You can copy the block as many times as you wish using this command.

A slightly different block is the selection of individual words or groups of words for block operations. By pressing "F0 m" (ALT/m for the PC), the attribute (m)ark is displayed. Then you can mark the words you wish to use in the block operation by marking them as explained above in the discussion on attributes.

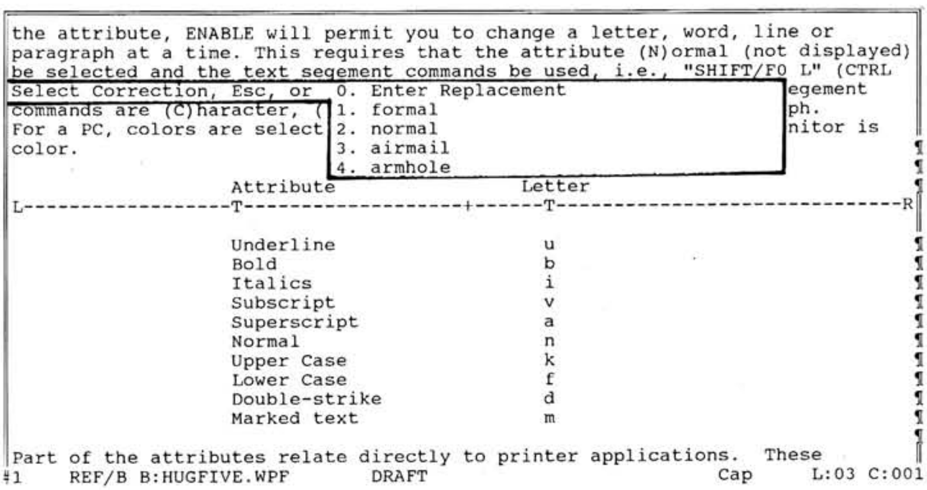
A spelling checker is included within the ENABLE package. ENABLE/CHECK can also be used to check the spelling of documents saved in ASCII, Volkwriter, EasyWriter, WordStar, MultiMate, PeachText, and DCA formats. The spelling checker within ENABLE is a fair product, although I like CorrectStar in WordStar better. It handles words better, checking for possessive and provides breaks. ENABLE/CHECK uses the 80,000 word Proximity/Merriam Webster Linguibase as its base. ENABLE's dictionary will check the text and compare it against words listed. When it finds a word that is not in the dictionary, it will stop there and offer options. It will check punctuation and capitalization for proper nouns.

ENABLE/CHECK is entered from the top line menu by pressing "F10 (2)"-Spell. You can then select the default menu or change the options. The default selection will check for double words and scan from the beginning of the document. (C)hange Options will permit the selection of the double word check and scan from the top or from the current position. This is helpful when you want to check part of the document.

When ENABLE finds an error, a word or character that is not in the dictionary, it will stop and display the "Word-Not-Found" in a window just below the line with the indicated word. This permits you to see the word in context. You have several selections from this menu with a <RETURN> giving you a choice of replacement words. The choices are (0) Enter Replacement, (1) Choose Replacement, (2) Ignore Word-Not-found, (3) Add -Lower Case, (4) Add - As Entered, (5) Add - Retyped Word, and (6) Exit ENABLE/CHECK. If you type "0", ENABLE will provide a highlighted area to enter the word. The "1" choice will open another window with possible replacement words from the dictionary. Normally the number "1" choice is the correct word. You can select any of the words displayed and it will replace the word-not-found. You can also enter a replacement from this screen. If the word is correct and you do not wish to entered it in the dictionary, select "2". Option "3" will permit you to add the displayed word to your personnel dictionary in lower case. Option "4" will enter the word as it is typed in the document. Option "6" will cause you to leave the spell checker. When you leave, a summary screen will be displayed with words checked, number of duplicate words, words not found, number of corrections made and number of words added to the dictionary. Pressing "ESC" will return you to the document.

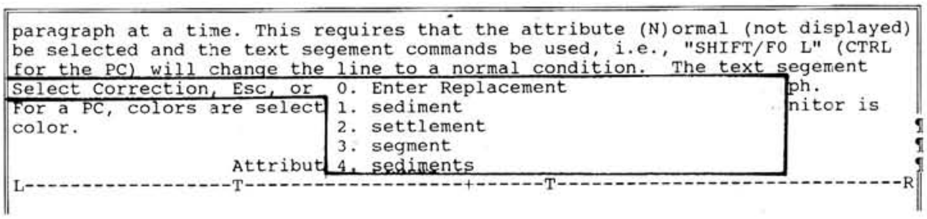
Words added to the dictionary are placed in a separate file called "USERDICT.TSC" You can view and change the words in this file using the word processor. To access the user dictionary to view or correct words, select (U)se System, (W)ord Processing, and (D)ictionary. You can now enter or correct words in the dictionary as any other word processing document. The maximum number of letters and/or character in any enter is 30. When you are done, save the work like any word processing document.

ENABLE/CHECK also checks for duplicate word entries. These errors are hard to find during normal proofreading if you read fast. ENABLE is case sensitive during this check so "Can can" will not be highlighted. When duplicate words appear in the text, ENABLE will stop the display on the word and ask if you want to delete the duplicate word, ignore the error, or exit the spell checker.



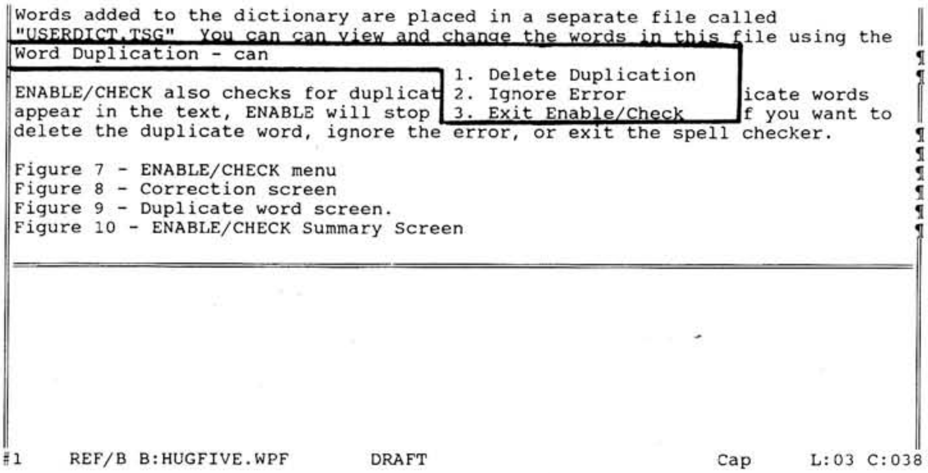
```
the attribute, ENABLE will permit you to change a letter, word, line or
paragraph at a time. This requires that the attribute (N)ormal (not displayed)
be selected and the text segement commands be used, i.e., "SHIFT/F0 L" (CTRL
Select Correction, Esc, or  0. Enter Replacement                    egement
commands are (C)haracter, ( 1. formal                               ph.
For a PC, colors are select 2. normal                               nitor is
color.                      3. airmail                                        ¶
                            4. armhole                                        ¶
                   Attribute                Letter                            ¶
L------------------T-------------------+------T-------------------------------R

                   Underline                  u                               ¶
                   Bold                       b                               ¶
                   Italics                    i                               ¶
                   Subscript                  v                               ¶
                   Superscript                a                               ¶
                   Normal                     n                               ¶
                   Upper Case                 k                               ¶
                   Lower Case                 f                               ¶
                   Double-strike              d                               ¶
                   Marked text                m                               ¶
                                                                              ¶
Part of the attributes relate directly to printer applications.  These
#1     REF/B B:HUGFIVE.WPF        DRAFT                          Cap     L:03 C:001
```

Figure 7 - ENABLE/CHECK menu



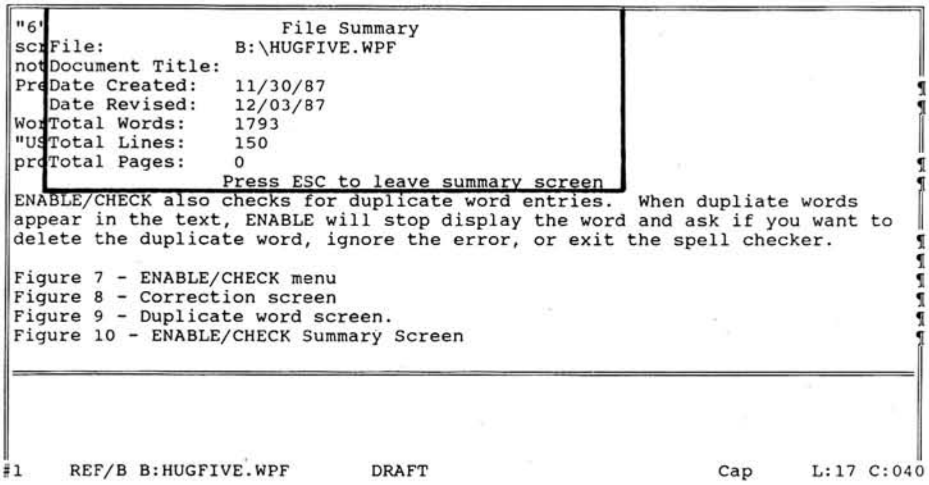
```
paragraph at a time. This requires that the attribute (N)ormal (not displayed)
be selected and the text segement commands be used, i.e., "SHIFT/F0 L" (CTRL
for the PC) will change the line to a normal condition.  The text segement
Select Correction, Esc, or  0. Enter Replacement                    ph.
For a PC, colors are select 1. sediment                             nitor is
color.                      2. settlement                                     ¶
                            3. segment                                        ¶
                   Attribut 4. sediments                                      ¶
L------------------T-------------------+------T-------------------------------R
```

Figure 8 - Correction screen



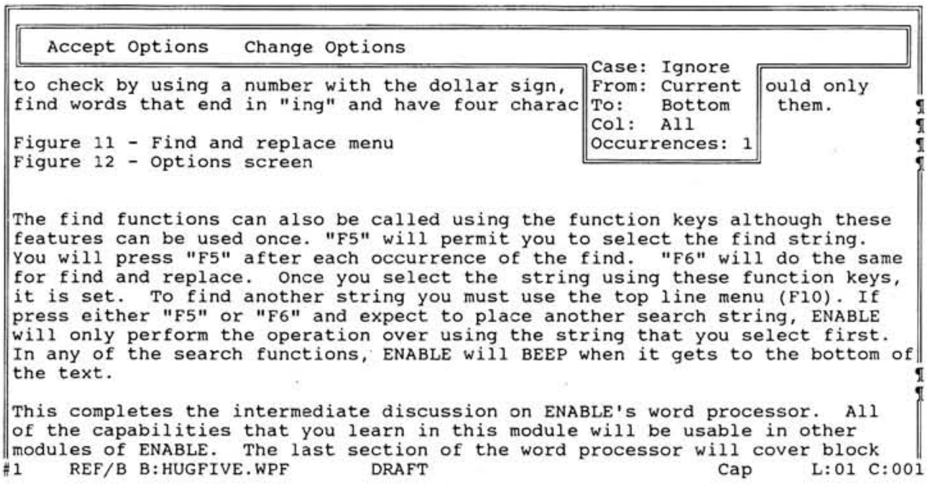
```
Words added to the dictionary are placed in a separate file called
"USERDICT.TSG"  You can can view and change the words in this file using the
Word Duplication - can
                                         1. Delete Duplication
ENABLE/CHECK also checks for duplicat 2. Ignore Error           icate words
appear in the text, ENABLE will stop  3. Exit Enable/Check      f you want to
delete the duplicate word, ignore the error, or exit the spell checker.

Figure 7 - ENABLE/CHECK menu
Figure 8 - Correction screen
Figure 9 - Duplicate word screen.
Figure 10 - ENABLE/CHECK Summary Screen




#1     REF/B B:HUGFIVE.WPF        DRAFT                        Cap      L:03 C:038
```

Figure 9 - Duplicate word screen.



```
"6'                      File Summary
scrFile:              B:\HUGFIVE.WPF
notDocument Title:
PreDate Created:      11/30/87
   Date Revised:      12/03/87
WorTotal Words:       1793
"USTotal Lines:       150
prdTotal Pages:       0
                         Press ESC to leave summary screen
ENABLE/CHECK also checks for duplicate word entries.  When dupliate words
appear in the text, ENABLE will stop display the word and ask if you want to
delete the duplicate word, ignore the error, or exit the spell checker.

Figure 7 - ENABLE/CHECK menu
Figure 8 - Correction screen
Figure 9 - Duplicate word screen.
Figure 10 - ENABLE/CHECK Summary Screen



#1     REF/B B:HUGFIVE.WPF        DRAFT                        Cap      L:17 C:040
```

Figure 10 - ENABLE/CHECK Summary Screen



```
  Accept Options   Change Options
                                           Case: Ignore
to check by using a number with the dollar sign,  From: Current  ould only
find words that end in "ing" and have four charac To:   Bottom    them.
                                           Col:  All
Figure 11 - Find and replace menu          Occurrences: 1
Figure 12 - Options screen


The find functions can also be called using the function keys although these
features can be used once. "F5" will permit you to select the find string.
You will press "F5" after each occurrence of the find.  "F6" will do the same
for find and replace.  Once you select the  string using these function keys,
it is set.  To find another string you must use the top line menu (F10). If
press either "F5" or "F6" and expect to place another search string, ENABLE
will only perform the operation over using the string that you select first.
In any of the search functions, ENABLE will BEEP when it gets to the bottom of
the text.

This completes the intermediate discussion on ENABLE's word processor.  All
of the capabilities that you learn in this module will be usable in other
modules of ENABLE.  The last section of the word processor will cover block
#1     REF/B B:HUGFIVE.WPF        DRAFT                        Cap      L:01 C:001
```

Figure 11 - Find and replace menu

ENABLE permits Finding and Replacing of characters within a document. The two options are to only find and display, to find and replace, or to find and mark. To select find, press "F10 (F)ind (1) Find only". You are then prompted for the characters you are looking for. This can be one letter/character to several groups of letters/characters, or a string. You are then prompted for acceptance of the default values or to change them. If you select change, you can make the search case sensitive, set where the search will begin (top, current,or bottom), and to set if only certain columns are to be checked.

The find and replace can be configured to do this throughout the document or only the first occurrence. You select this option in the same manner as the find only. You are prompted for the replacement string after the search string. Again you can change the default values for the search and replace. One additional option is the number of occurrence you want to change. If you select "1", ENABLE will stop after every change. Find and replace works great when you are using a document and want to change one name throughout the document.

The last find option is find and mark. This selection permits you to indicate a string you wish to find and mark. You are prompted for the string and given a chance to change the default values for the operation. If you place "all" in the number of occurrences, all occurrences of the string will be highlighted within the document.

If you are not sure of the string you are looking for, ENABLE permits you to place wild card characters in the strings. The two wild card characters are "$" and "?". The "?" would replace one letter in the string, i.e., "th?n" would find than or then. The dollar sign is like the search character in dBase. If you are not sure of the spelling of the search string, you can use as little as one letter. "a" would find any string with an "a" in it. ENABLE permits you to select the number of character positions you would like to check by using a number with the dollar sign, i.e., "4$ing" would only find words that end in "ing" and have four characters in front of them.

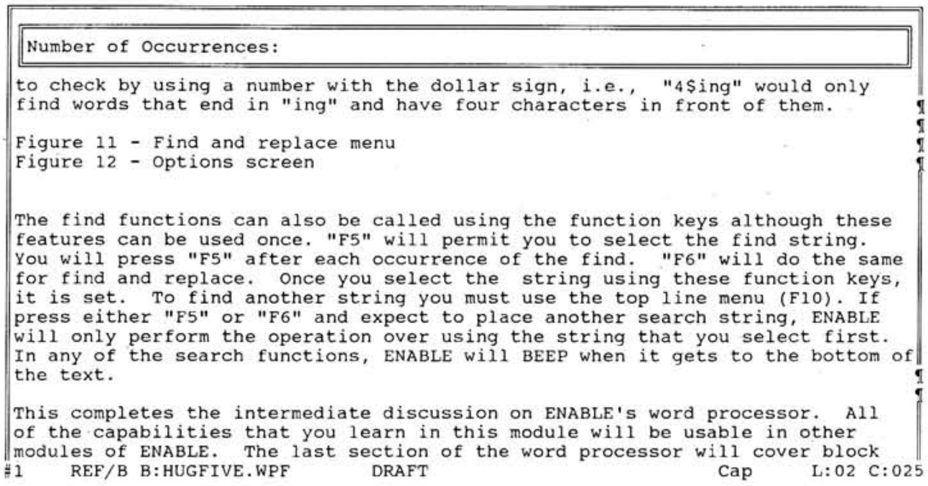
The find functions can also be called using the function keys although these features can be used once. "F5" will permit you to select the find string. You will press "F5" after each occurrence of the find. "F6" will do the same for find and replace. Once you select the string using these function keys, it is set. To find another string, you must use the top line menu (F10). If you press either "F5" or "F6" and expect to place another search string, ENABLE will only perform the operation over using the string that you selected first. In any of the search functions, ENABLE will BEEP when it gets to the bottom of the text.

This completes the intermediate discussion on ENABLE's word processor. All of the capabilities that you learn in this module will be usable in other modules of ENABLE. The last section of the word processor will cover block graphics, windows, and mail merge. The next article in this series will cover intermediate spreadsheet.

Using ENABLE
Que Corporation
P.O. Box 50507
Indianapolis, IN 46250
$17.95

Taking Command of ENABLE
Key Publications
Corporate Computer Training Center
812 Lydon Ln.
Louisville, KY

*



```
Number of Occurrences:

to check by using a number with the dollar sign, i.e.,  "4$ing" would only
find words that end in "ing" and have four characters in front of them.

Figure 11 - Find and replace menu
Figure 12 - Options screen


The find functions can also be called using the function keys although these
features can be used once. "F5" will permit you to select the find string.
You will press "F5" after each occurrence of the find.  "F6" will do the same
for find and replace.  Once you select the  string using these function keys,
it is set.  To find another string you must use the top line menu (F10). If
press either "F5" or "F6" and expect to place another search string, ENABLE
will only perform the operation over using the string that you select first.
In any of the search functions, ENABLE will BEEP when it gets to the bottom of
the text.

This completes the intermediate discussion on ENABLE's word processor.  All
of the capabilities that you learn in this module will be usable in other
modules of ENABLE.  The last section of the word processor will cover block
#1    REF/B B:HUGFIVE.WPF        DRAFT                         Cap     L:02 C:025
```

Figure 12 - Options screen

Expanded HUG Discount List
Featuring Products Eligible For Discount To HUG Members Only

CB-5063-50	10%
HS-248-T	10%
HS-248-TX	10%
HS-386A	10%
HSM-100	20%
HSM-100-3	20%
HV-2000	10%
HWD-20	10%
HWD-20-AT	10%
PM-160	20%
PM-160-3	20%
PMK-121	10%
PMK-130	10%
SK-201	10%
SK-202	10%
SK-203	10%
SK-204	10%
SK-205	10%
SK-209	10%
SK-210	10%
SK-211	10%
TM-140	10%
TM-150	10%
TM-158	10%
TM-159	10%
TM-170	10%
TM-200	10%
TM-240	10%
TM-380	10%
Z-205-4	10%
Z-207-7	10%
Z-304	10%
Z-315	10%
Z-316-8	10%
Z-405A	10%
Z-405-1	10%
Z-415	10%
Z-416-2	10%
Z-416-C	10%
Z-417	10%
Z-445	10%
Z-449	10%
Z-505	10%
Z-515	10%
Z-516	10%
Z-525	10%
ZA-170-1	10%
ZA-170-3	10%
ZA-170-4	10%
ZA-180-20	20%
ZA-180-21	10%
ZA-180-35	10%
ZA-180-40	10%
ZA-180-45	10%
ZA-181-4	10%
ZA-181-7	10%
ZA-181-8	10%
ZA-181-9	10%
ZA-181-17	10%
ZA-181-19	10%
ZA-181-20	10%
ZB-315-1	10%
ZBF-2337-BK	20%
ZBF-2339-BK	20%
ZBF-2526-EK	20%
ZBF-2527-EK	20%
ZBF-3339-EK	20%
ZBF-3340-EK	20%
ZCM-1390	20%
ZCM-1400-1	10%
ZCM-1490	20%
ZD-12	10%
ZD-14	10%
ZD-200	10%
ZD-372	10%
ZD-400	10%
ZD-800	10%
ZDF-1211-DY	20%
ZDF-1217-DY	20%
ZDF-2225-BK	20%
ZDF-2237-BK	20%
ZDF-2255-BK	20%
ZF-171-42	20%
ZFL-181-93	20%
ZKB-2	20%
ZMM-149A	20%
ZMM-149P	20%
ZMM-1470G	20%
ZVM-135	20%
ZVM-1200-1	10%
ZVM-1240	20%
ZVM-1300-1	10%
ZVM-1330	20%
ZVM-1380C	20%
ZWL-183-93	20%
All software included in the software chart shown on pages 89, 90, 91 of catalog #210	20%

Continued from Page 36

You could call the column: "Rookie Headquarters".

Sincerely,

T. R. Menneke
Bitburg AB
West Germany

For A Beginner's Column

Dear HUG:

I am very much for a Beginner's Column. As an owner of a Z-100, which I purchased used, recently, I got excited with it and had it upgraded to 768K RAM and purchased WordPerfect. My many years of experience has been only in the use of computers in a business environment, that is, I served in the role of a den mother to programs written by others. I never owned a computer and as a beginner, I am frustrated with reading REMark. It's too deep for me.

I want to expand with the use of a mouse, learn what graphics can be done with my equipment and experiment with BASIC program writing.

I would like to read what other beginners have done with the Z-100. Possibly, you could print the names and addresses of those people who would like to correspond and give assistance. There may be some Z-100 users around the corner I could get to know. Another problem I have is knowing what to specify in purchasing a mouse, an RGB monitor, etc. Perhaps, software could be exchanged and sold on an economical basis what with living on Social Security.

I believe the local HUG group is too sophisticated and ahead of me with their technical knowledge and interests. I live on a Technical Island with beaucoup tech companies.

I applaud the suggestion of a Beginner's Column!

Very truly yours,

Arthur R. Siegel
1 Ash Place #3B
Great Neck, NY 11021

Another For The Beginner's Column

Dear HUG:

Three cheers for the Z-151 user who wants a beginner's column! (2/88)

I have been a HUG member for over two years, at least, and I keep wondering why I keep subscribing as everything seems way over my head.

I have built a '151 and a '148 for mostly personal use, and do not have time to take courses, so a "beginner's" column would be wonderfully welcome. I'm sure there are many, many others like myself.

Here I sit with Microsoft Word and cannot figure out how to bring the spellcheck portion up to check my spelling on the letters and papers I type! I read and reread the manuals to no avail.

Yes, we beginners need a column.

Cordially,

Marshall Ginsburg
1328 Buttermilk Lane
Reston, VA 22090

One More For A Beginner's Column

Dear HUG:

Add my vote for a beginner's column, please. It would make the journal much more meaningful for me and I suspect many others.

Sincerely,

Sharon L. Woodcock
2404 Vinewood Boulevard
Ann Arbor, MI 48104

"Plain Vanilla Floppy"

Dear HUG:

In the January 1988 issue of REMark, columnist William Adney gave some good advice on troubleshooting software. The first thing he suggested was to return your machine to a "plain vanilla" configuration by renaming your AUTOEXEC.BAT and CONFIG.SYS files, and then rebooting to remove TSR's and device drivers that might conflict with the new software. This is the best first step, but here is an easier way to do it. You don't have to rewrite your files each time if you prepare a bootable floppy (label it "Plain Vanilla Booter") using the FORMAT /S command and then writing a new AUTOEXEC.BAT file to do some of the things you have to do (like call up the date and time for your onboard clock if you have one), and maybe a minimum CONFIG.SYS file setting the buffers and files. Then, when you have a problem with a new piece of software you boot from this vanilla floppy. Thus, you leave your original files alone and do not take the chance of creating a major problem for yourself. You also save time and know that everything is done right. This "Vanilla floppy" is a great $1.00 investment. Even if you have a hard disk that is the default booter, you can still boot from a floppy. Each Zenith machine does this differently, but your Zenith documentation will tell you how.

Yours truly,

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

Thanks For Running T. E. Thompson's Articles

Dear HUG:

Had to drop you a line and thank you for running the articles by T. E. Thompson. They have excited me more than any articles since the early days of HUG. I think this is a very interesting field for home computerists to get into and I hope you will print many more of his articles.

Thanks again and Best Regards,

R. C. Olson
24450 Kirby Street #146
Hemet, CA 92343

A Do-It-Yourself Printer Driver For AutoCAD

Patrick Swayne
HUG Software Engineer

In my article "Desktop Publishing with AutoCAD," I stated that AutoCAD versions 2.5 and above support user supplied printer drivers. In this article, I will present information on how to write a printer driver in assembly language, along with the source code for an actual driver designed for use with the MPI 150 printer.

Beginning with version 2.18 (approximately) of AutoCAD, AutoDesk (the company that produces it) invented something called the AutoDesk Device Interface (ADI) so that peripheral devices not supported by AutoCAD's built-in drivers could be supported by user-supplied external drivers. At first only digitizers were supported by ADI drivers, but later (with versions 2.5 and above), support for ADI drivers for all of the peripheral devices AutoCAD can use, including the screen, digitizer, plotter and printer, was added. Information on ADI drivers is provided free of charge by AutoDESK to anyone who has a modem and a CompuServe membership. All you have to do is log onto the AutoDesk SIG on CompuServe (type GO ADESK), and download a file called ADIKIT.ARC. This file is an archived (compressed) library of documentation files and sample ADI drivers that can be studied by anyone who wants to write a driver for an unsupported device. The ARC utility, available for downloading from just about any computer bulletin board, can be used to retrieve the files in ADIKIT.ARC.

The only thing wrong (at least to me) with the sample drivers in the ADI Kit is that they are written in the C programming language, with some subroutines in assembly language. I prefer to write just about everything I do entirely in assembler, so I studied the documentation in the Kit to learn the specifications required for a driver. Then I wrote my driver in assembly language from "scratch", without referring to the sample driver written in C at all. The result is a fast, compact driver that works well. The assembly code for my driver is listed at the end of this article.

The AutoCAD ADI printer driver can pass information to the user- supplied driver in three ways. It can write the information to a disk file in two different formats, which the user's driver can then process to make the plot. Or it can transfer information directly via a software interrupt. My driver uses the interrupt method and processes the information from AutoCAD directly.

Information is sent to the driver from AutoCAD via the processor's registers. The AX register will contain either a function code or a byte count. The high bit is used to flag a function code (which means that the maximum byte count possible is 32767), and the code itself is in the low byte (AL register). There are only 3 codes: Begin Plot, End Plot, and Abort Plot. If the code is Begin Plot, the BX, CX, and DX registers will contain a File Level (actually a version number), the horizontal resolution of the printer in dots, and the vertical resolution. My driver ignores all of this information. It does not care about the version number, and its resolution is fixed at the maximum resolution that an MPI printer can support. However, it does initialize the printer when it receives the Begin Plot code. It returns a 1 in the AX register if initialization is successful or 0 if not, and it returns a 0 in the BX register to indicate that the printer does not support color.

If the AX register contains a byte count, then the BX register will be an offset address of the data bytes, and the CX register will be the segment address. For a monochrome printer, each bit within each data byte represents a dot to be placed on the paper, and the entire group of data bytes for each interrupt represents a "scan line" of dots across the paper. As you probably know, the printhead on a dot matrix printer has 9 or more wires (or pins) arranged in a vertical line, so it is capable of producing several scan lines with each pass of the printhead across the paper. When a typical dot matrix printer is in the graphic mode, the bits within each byte send to it activate one of the printhead wires, so that up to 8 scan lines are printed simultaneously with each pass of the printhead. Since AutoCAD sends the information a single scan line at a time, the driver must perform a conversion of bits representing horizontal dots on a single scan line to bits representing vertical dots of several scan lines. My driver does this by saving scan lines sent by AutoCAD in a buffer until the number of lines saved equals the maximum number of printhead wires the printer can "fire" in a single pass in the graphic mode. Then it converts the information from horizontal bits to vertical bits, and sends it to the printer.

The code that handles translation from horizontal to vertical bits is below the comment "DATA CONVERSION SECTION" in the source listing. You can study the code to see how it works. If you are planning to write your own driver, you should be aware that the MPI printer that my driver is written for requires the graphic data to be in ASCII format, and that it only fires a maximum of 6 printhead wires in the graphic mode. That is why only 6 lines from AutoCAD are saved and processed at a time, and why the converted byte is OR'ed with 40H before being stored.

Some graphic printers, including most in the Epson line, can move the printhead a fraction of the width of one printhead wire vertically. An Epson MX-80 with Graftrax Plus, for example, can move the printhead 1/3 the width of a wire. The wires are 1/72 of an inch wide, so the normal vertical graphic resolution is 72 dots per inch, but by using fractional vertical movements, it is possible to have an effective vertical resolution of 216 dots per inch. A driver written to support this resolution could store 24 lines at a time from AutoCAD, and then form vertical bytes from every third horizontal line. It would do this three times, starting one line down each time.

If my driver receives an End Plot or Abort Plot code from AutoCAD, it converts and prints any scan lines that have been stored even though the number of lines sent since the last conversion may be less than 6. The driver does nothing else at this point, but if you are designing a driver, you may want to have it issue a form feed to the printer to eject the completed drawing.

Configuring the Driver

To use my MPI 150 driver, you must load it into memory before you run AutoCAD by entering the the name of the program at the system prompt. I called the driver ADIMPI.COM, and you can use that name if you wish. After you load the driver, and before you can use it for the first time, you must configure AutoCAD. Select the Configure printer plotter option from the configuration menu, and select "ADI printer plotter" from the list of available devices. (You will need to insert your second driver disk in a drive, unless all of your drivers are on your hard disk.) Answer the questions asked as follows:

Specify plot size in millimeters <N> (press Return)
Maximum horizontal (X) plot size in inches? <11.0>

Type 8 to the above question if you have narrow paper, or 13.5 if you have wide paper.

Printer dots per inch in the horizontal (X) direction <100.0> 85
Maximum vertical (Y) plot size in inches? <8.5>

Answer 10 to the above question if you want to confine your drawing to a single sheet of paper, or any larger number if you want to allow it to "spill over" onto other sheets.

Printer dots per inch in the vertical (Y) direction? <100.0> 72
Select output format: 2
Hexadecimal interrupt code (INT 0XXh) <7B>: (press Return)
Does the printer/plotter support color? <N> (press Return)

The rest of the configuration is as for supported printers, and you can consult your AutoCAD installation manual for more information. When you are asked

Rotate plot 90 degrees? <N>

type Y if you are using narrow paper, and you have a drawing that is wider than it is high and you want it to fill the page as much as possible. You may also want to answer Y if you are using wide paper and you allowing the drawing to spill over more than one sheet of paper. When you plot a drawing, you will have the opportunity to change some of the above settings.

Below is the source listing of my MPI 150 driver for AutoCAD. It should also work with MPI 99G and Sprinter printers. It can be assembled in a test mode by setting TEST EQU TRUE. It does not load itself into memory in the test mode, but prints a small pennant shape on the paper and then exits.

```
          PAGE    ,132
          TITLE ADI INT DRIVER FOR MPI 150 PRINTER
;         THIS PROGRAM IS A MEMORY RESIDENT UTILITY THAT
;         PROVIDES AN ADI DRIVER INTERRUPT PROCESSOR FOR
;         AUTOCAD VERSION 2.5 OR HIGHER TO ALLOW PLOTTING
;         ON AN MPI 150 OR MPI SPRINTER PRINTER.
;
;         BY PATRICK SWAYNE, HUG SOFTWARE ENGINEER 09-NOV-87
TRUE      EQU     0FFFFH            ;DEFINE TRUE
FALSE     EQU     NOT TRUE          ;AND FALSE
TEST      EQU     FALSE             ;ASSEMBLE TEST VERSION IF TRUE

CODE      SEGMENT
          ASSUME CS:CODE,DS:CODE,ES:CODE,SS:CODE
          ORG     100H

START:    JMP     SETUP             ;SET UP PROGRAM

;         DATA AREA

NEFLG     DB      0                 ;NOT EMPTY FLAG

;         INPUT LINE BUFFERS.  6 LINES ARE INPUT AT A TIME
;         BECAUSE THE MPI PRINTER DRIVES 6 PINS DURING GRAPHIC
;         OUTPUT.

LINE1     DB      146 DUP (0)       ;ROOM FOR FULL WIDTH PAGE
LINE2     DB      146 DUP (0)       ; PLUS A COUNT BYTE
LINE3     DB      146 DUP (0)
LINE4     DB      146 DUP (0)
LINE5     DB      146 DUP (0)
LINE6     DB      146 DUP (0)

LINPTR    DW      0                 ;LINE POINTER
;         OUTPUT LINE BUFFER.  THE LINE IS 8 TIMES THE LENGTH
;         OF ONE LINE ABOVE, BECAUSE OF 8 DOTS PER BYTE.

GRCODE    DB      1BH,'W'           ;PRINT GRAPHICS COMMAND
OUTLINE   DB      '6'               ;END OF LINE MARKER
          DB      1151 DUP (0)
          DB      0                 ;SPACER

;         ADI INTERRUPT PROCESSOR

ADIINT:   PUSH    DX                ;WE'LL USE THIS
          STI
          TEST    AH,80H            ;CHECK FOR CONTROL
          JZ      DATALIN           ;IT;S A DATA LINE
          CMP     AL,1              ;START PLOT?
```

```
        JZ      STPLOT          ;YES
        PUSH    SI
        PUSH    DI
        PUSH    DS
        PUSH    ES
        PUSH    CS
        POP     ES              ;PUT ES HERE
        PUSH    CS
        POP     DS              ;PUT DS HERE
        JMP     PRTLIN          ;PRINT THE LAST LINE
STPLOT:MOV      AH,2
        MOV     DX,0
        INT     17H             ;TEST PRINTER PORT
        AND     AH,90H
        CMP     AH,90H          ;IS PRINTER READY?
        JNZ     PRNBAD          ;NO
PRCH1:  MOV     AX,1BH          ;AH = 0, AL = ESC
        INT     17H             ;SEND ESCAPE TO INIT PRINTER
        TEST    AH,80H
        JZ      PRNBAD          ;PRINTER CONKED OUT
PRCH2:  MOV     AX,5FH
        INT     17H             ;SET 85 DOTS PER INCH
        TEST    AH,80H
        JZ      PRNBAD
        MOV     BX,0            ;INDICATE MONOCHROME
PRNOK:  MOV     AX,1            ;RETURN INIT OK
        POP     DX
        IRET
PRNBAD:XOR      AX,AX           ;RETURN INIT NO GOOD
        POP     DX
        IRET

;       A DATA LINE HAS BEEN SENT.  STORE IT, AND IF 6 LINES
;       HAVE BEEN SENT, PRINT THEM.

DATALIN:PUSH    SI              ;SAVE SOME REGISTERS
        PUSH    DI
        PUSH    DS
        PUSH    ES
        PUSH    CS
        POP     ES'             ;PUT ES HERE
        OR      AX,AX           ;EMPTY LINE?
        JZ      EMLIN           ;IF SO, SKIP IT
        MOV     CS:NEFLG,1      ;MARK LINE NOT EMPTY
        CMP     AX,145          ;TEST LINE SIZE
        JBE     DATAL1          ;OK
        MOV     AX,145          ;ELSE, TRUNCATE
DATAL1: MOV     DS,CX           ;DATA SEGMENT TO DS
        MOV     SI,BX           ;DATA OFFSET TO SI
        MOV     CX,AX           ;COUNT TO CX
        MOV     AX,ES:LINPTR    ;GET LINE POINTER
        MOV     BX,146          ;AND SIZE OF LINE
        MUL     BX              ;GET LINE OFFSET
        ADD     AX,OFFSET LINE1;AX = DESTINATION ADDRESS
        MOV     DI,AX           ;RESULT TO DI
        CLD
        MOV     AL,CL
        STOSB                   ;STORE COUNT
        REP     MOVSB           ;MOVE THE DATA
```

```
EMLIN:  PUSH    CS
        POP     DS              ;PUT DS HERE
        MOV     AX,LINPTR       ;GET LINE POINTER
        INC     AL              ;ADD 1
        MOV     LINPTR,AX       ;UPDATE POINTER
        CMP     AL,6            ;6 LINES DONE?
        JZ      PRTLIN          ;TIME TO PRINT LINE
LINDN:  POP     ES
        POP     DS
        POP     DI
        POP     SI
        JMP     PRNOK           ;EXIT, INDICATING OK
PRTLIN:MOV      LINPTR,0        ;RESET LINE POINTER
        CMP     NEFLG,0         ;IS LINE EMPTY?
        JZ      DOPRT           ;IF SO, SKIP CONVERSION

;       FIND OUT WHICH LINE IS LONGEST, AND USE LENGTH FOR
;       CONVERSION COUNT

        MOV     SI,OFFSET LINE1 ;POINT TO LINE 1
        MOV     AL,[SI]         ;GET COUNT THERE
        MOV     CX,5            ;SET COUNTER
FNDSIZ:ADD      SI,146          ;MOVE TO NEXT LINE
        CMP     AL,[SI]         ;COMPARE SIZES
        JAE     SIZOK           ;CURRENT SIZE IS OK
        MOV     AL,[SI]         ;ELSE, GET BIGGER SIZE
SIZOK:  LOOP    FNDSIZ
        MOV     DH,AL           ;SIZE TO DH

;       DATA CONVERSION SECTION.  WHAT WE HAVE TO DO IS CONVERT
;       INFORMATION IN SCREEN (SCAN LINE) FORMAT (HORIZONTAL BYTES
;       OF PIXELS) INTO PRINT HEAD FORMAT (VERTICAL BYTES OF DOTS).

        MOV     DI,OFFSET OUTLINE ;POINT TO OUTPUT LINE
        MOV     SI,OFFSET LINE1+1 ;POINT TO LINES SAVED (LESS COUNT)
MLOOP:  MOV     DL,8            ;8 DOTS PER BYTE
BLOOP:  MOV     AL,0            ;CLEAR OUTPUT BYTE
        MOV     CX,6            ;6 LINES TO PROCESS
SLOOP:  SHL     BYTE PTR [SI],1 ;SHIFT BYTE IN A LINE
        RCL     AL,1            ;PUT RESULT IN OUTPUT BYTE
        ADD     SI,146          ;MOVE TO NEXT LINE
        LOOP    SLOOP           ;UNTIL 6 LINES DONE
        OR      AL,40H          ;MAKE RESULT ASCII
        STOSB                   ;STORE IT
        SUB     SI,146*6        ;MOVE BACK TO LINE 1
        DEC     DL              ;COUNT THIS ROW OF DOTS
        JNZ     BLOOP           ;LOOP UNTIL INPUT BYTE PROCESSED
        INC     SI              ;MOVE TO NEXT BYTE
        DEC     DH              ;COUNT THIS BYTE
        JNZ     MLOOP           ;UNTIL ALL BYTES PROCESSED

;       STORE ASCII "6" AT THE END OF THE LINE, TO SIGNAL
;       THE END OF GRAPHICS PRINTING.

        MOV     AL,'6'
        STOSB

;       NOW, PRINT THE CONVERTED LINE, WHICH REPRESENTS 6 LINES OF
;       DATA FROM AUTOCAD.
```

```
DOPRT: MOV    NEFLG.0          ;CLEAR NOT EMPTY FLAG
       MOV    SI.OFFSET GRCODE ;POINT TO OUTPUT LINE + GRAPHIC CODE
       MOV    DX,0             ;SET PRINTER PORT
PRLOOP:LODSB                   ;GET A BYTE
       MOV    AH,0
       INT    17H              ;PRINT IT
       TEST   AH,80H           ;TEST PRINTER
       JZ     PRTBAD           ;IT'S BAD
       CMP    AL,'6'           ;DONE?
       JNZ    PRLOOP           ;IF NOT, LOOP
       MOV    DI.OFFSET LINE1  ;POINT TO DATA
       MOV    CX,(146*6)/2
       MOV    AX,0
       REP    STOSW            ;CLEAR EVERYTHING
       MOV    DI.OFFSET OUTLINE
       MOV    CX,1152/2
       REP    STOSW
       MOV    BYTE PTR OUTLINE,'6' ;ALLOW NULL LINE
       JMP    LINDN            ;LINE DONE, EXIT
PRTBAD:POP    ES
       POP    DS
       POP    DI
       POP    SI
       JMP    PRNBAD           ;EXIT, INDICATING BAD
ENDRES:                        ;END OF RESIDENT CODE

;      INSTALL THIS DRIVER IN MEMORY

SETUP: MOV    AX,257BH         ;SET INT. 7BH
       MOV    DX,OFFSET ADIINT ;HANDLER IS HERE
       INT    21H
       MOV    DX,OFFSET SIGNON
       MOV    AH,9
       INT    21H              ;PRINT SIGN-ON
       MOV    DX,OFFSET ENDRES ;POINT TO END OF RESIDENT CODE
       IF     NOT TEST
       INT    27H              ;EXIT, CODE RESIDENT
       ELSE
       MOV    AX,8001H
       INT    7BH              ;INITIALIZE PRINTER
       OR     AL,AL            ;TEST PRINTER
       JNZ    GOON             ;GO ON WITH TEST
       MOV    DX,OFFSET SAYBAD
       MOV    AH,9
       INT    21H
       INT    20H
GOON:  MOV    DL,6             ;SET A COUNTER
       MOV    AX,6             ;GET A BYTE COUNT
       MOV    BX,OFFSET GDATA  ;POINT TO GRAPHIC DATA
TSTLOOP:MOV   CX,DS            ;USE THIS DS
       PUSH   AX               ;SAVE COUNT
       PUSH   BX               ;AND POINTER
       INT    7BH              ;CALL DRIVER
       POP    BX
       POP    AX
       DEC    AX               ;MAKE NEXT LINE SHORTER
       DEC    DL
       JNZ    TSTLOOP
       MOV    AX,8002H
       INT    7BH              ;SIGNAL END
       INT    20H
SAYBAD DB     13,10,'Printer reported bad.$'
GDATA  DB     0FFH,0FFH,0FFH,0FFH,0FFH,0FFH
       ENDIF

SIGNON DB      13,10,'ADI Driver for MPI 150 printer, v 1.0',13,10,'$'

CODE   ENDS
       END    START
```

*

RAM Disk Usage

Edward W. Snyder
4045 Forest Ridge Boulevard
Dayton, OH 45424-4834

If you're anything like me, you've been itching to finally be able to use a RAM Disk on your computer. That's all well and good. But how can you use the RAM Disk to save disk space or prolong the life of your disks? I'll mention two things below that seem to me to save disk space and to reduce the disk accesses during bootup. The examples below were written for a Zenith Z-121, but the concept will work for most any computer.

I like to have a few small utility .COM and .BAT files in the RAM Disk, since they save me time, effort, and typing. If I kept each file on the booth disk (hard and/or floppy), each file would take up from 1K to 4K bytes (1K = 1024 bytes), even if it were only 7 bytes long. That's a lot of wasted space if you have ten to fifteen or more of those little files! I read in one of the computer magazines where someone had created a file with a redirected input file to DEBUG, and decided that that could be the way to go. I built a DEBUG input file to build all the small files, I would normally load in the RAM Disk (see Example Listing 1, BOOT.INP). For each file, I described just what it was for (note that DEBUG will label all the comments as errors, but it doesn't stop the execution and makes it more readable), named the file (including the RAM Disk letter), cleared a little space (no special reason, except it kept the field clear if I needed to debug it), entered in the file length and the necessary data, then wrote it to the RAM Disk. The necessary data can be found by DEBUGging the .COM file of interest, DEBUG assembling a file (as given in some computer magazines), or by entering the text (surrounded by apostrophes and followed by a Carriage Return and Line Feed) and counting the characters.

So now instead of those ten files used in the example taking up between 10K and 40K bytes on your disk, the one file only takes up between 2K and 4K bytes. If I didn't want to have so many comments in the file, it would take up even less.

Then I started wondering how else I could accomplish the same end but without using DEBUG. ARC came to mind immediately. With this you could save small, as well as larger files and extract them at boot time. The same ten files as above were reduced to just 1K bytes on a floppy (up to 4K on a hard). But like everything else, there is a trade-off to consider. DEBUG takes up just 13K bytes (on floppy), while the ARC version I have takes up 32K bytes, two and a half times more space (see DIR of files, Listing 2).

The other use I found for the RAM Disk concerns running .BAT files from it. It has always bugged me how slow the REM and ECHO statements were in .BAT files, and I like to use them to let the user know what's going on while the system is being set up. An answer turned out to be very simple: Let the AUTOEXEC.BAT file do nothing much more than copy a .BAT file to the RAM Disk, then run from the RAM Disk! This resulted in fewer disk accesses and going through set-up much faster (see examples, Listings 3 and 4, AUTOEXEC.BAT and BOOT.BAT). Try it yourself and see if you like it, too.

Listing 1

```
* DEBUG INPUT FILE FOR WRITING SEVERAL RAMDISK FILES

*****  SEND <BEL> (07H) TO TERMINAL
NI:BEL.COM
F 0 100 00
RCS
000B
E100 BA 09 01 B4 09 CD 21 CD 20 07 24
W

*****  SEND <BEL> (07H) TO TERMINAL AND PRINTER
NI:BELL.COM
F 0 100 00
RCS
0011
E100 BA 0F 01 B4 09 CD 21 B2 07 B4 05 CD 21 CD 20 07 24
W

*****  BATCH FILE FOR SORTED DIRECTORY SHOWING DIRECTORY NAMES:
*        SDIR %1/R
NI:D.BAT
F 0 100 00
RCX
000A
E100 'SDIR %1/R' 1A
W

*****  SEND <FF> (0CH) TO PRINTER
NI:FF.COM
F 0 100 00
RCX
000C
E100 B4 05 B2 0D CD 21 B2 0C CD 21 CD 20
W

*****  TURN CONSOLE KEYCLICK OFF
NI:KCOFF.COM
F 0 100 00
RCX
000D
E100 BA 09 01 B4 09 CD 21 CD 20 1B 78 32 24
W

*****  TURN CONSOLE KEYCLICK ON
NI:KCON.COM
F 0 100 00
RCX
000D
E100 BA 09 01 B4 09 CD 21 CD 20 1B 79 32 24
W

*****  BATCH FILE TO "DUMP" TO PRINTER:
*        TYPE I:LPD > PRN [CR,LF]
*        TYPE %1 > PRN [CR,LF]
*        TYPE I:LPR TO PRN [CNTL Z]
NI:LP.BAT
F 0 100 00
RCX
0032
E100 'TYPE I:LPD > PRN' 0D 0A 'TYPE %1 > PRN' 0D 0A 'TYPE I:LPR > PRN' 1A
W

*****  DEC LN03 PRINTER "DUMP" SETUP SEQUENCE:
*        SOFT RESET, 12 CPI, LM6, RM80, TM1, BM1
NI:LPD
F 0 100 00
RCX
0029
E100 1B 5B 21 70 1B 5B 3F 32 30 20 4A 1B 5B 31 34 6D
E110 1B 5B 30 77 1B 5B 30 7A 1B 5B 39 3B 39 30 73 1B
E120 5B 32 3B 36 35 72 0D 0A 1A
W

*****  DEC LN03 PRINTER SOFT RESET:
NI:LPR
F 0 100 00
RCX
000F
E100 1B 5B 21 70 1B 50 30 3B 31 3B 30 79 1B 5C 1A
W

*****  BATCH FILE TO PRINT CHKDSK AND SDIR DATA:
*        TYPE I:LPD > PRN [CR,LF]
*        CHKDSK %1*.* > PRN [CR,LF]
*        SDIR %1/A/R/W > PRN [CR,LF]
*        TYPE I:LPR > PRN [CNTL Z]
*        BEL
NI:SDL.BAT
F 0 100 00
RCX
0051
E100 'TYPE I:LPD > PRN' 0D 0A 'CHKDSK %1*.* > PRN' 0D 0A
E126 'SDIR %1/A/R/W > PRN' 0D 0A 'TYPE I:LPR > PRN' 0D 0A 'BEL' 1A
W

*****  DEBUG INPUT FILE COMPLETED
Q
```

Listing 2
DIR Of Files Used

```
 Volume in drive B is SUBMISSION
 Directory of  B:

ARTICLE  DOC    7552   7-14-87   8:42a
DEBUG    COM   15786   4-01-86   9:10a
BOOT     INP    1874   7-09-87  12:52p
ARC      EXE   32429   2-13-86  10:23a
BOOT     ARC     531   7-14-87   8:20a
DIR      DOC     878   7-14-87   8:27a
AUTOEXEC BAT      79   7-14-87   8:26a
BOOT     BAT    1452   7-14-87   8:27a
BEL      COM      11   7-14-87   8:15a
BELL     COM      17   7-14-87   8:15a
D        BAT      10   7-14-87   8:15a
FF       COM      12   7-14-87   8:15a
KCOFF    COM      13   7-14-87   8:15a
KCON     COM      13   7-14-87   8:15a
LP       BAT      50   7-14-87   8:15a
LPD               41   7-14-87   8:15a
LPR               15   7-14-87   8:15a
SDL      BAT      81   7-14-87   8:15a
      18 File(s)    290816 bytes free
```

Listing 3
AUTOEXEC>BAT File Example

```
ECHO OFF
CLS
RAMLIM SIZE=316 START=452
COPY BOOT.BAT I: > NUL
I:
BOOT
```

Listing 4
BOOT.BAT File Example

```
ECHO OFF
CLS
ECHO
ECHO       ***  BE SURE TO ENTER TODAYS DATE AND CURRENT TIME!!  ***
ECHO
DATE
ECHO
TIME
VERIFY ON
A:
RESNET
I:
A:PSC
PATH=I:\;A:\
CLS
ECHO
ECHO                              Copying key files to the RAM-Disk  . . .
ECHO
ECHO      COMMAND.COM     BEL.COM         DBASE.COM       PT.EXE
ECHO      ALTCHAR.SYS     BSE.EXE         DBASEOVR.COM    PT.PGM
ECHO      CHKDSK.COM      FF.COM                        PTMENU.PGM
ECHO      DEBUG.COM       LABEL.COM                     PTHELP.DAT
ECHO      SDIR.COM        LOCATE.COM                     EDIT.PGM
ECHO                    LP.BAT (LPD/LPR)                PRINT.PGM
ECHO                    PAGE.EXE
ECHO                    SDL.BAT
A:LABEL I:RAM DISK
DEBUG < A:BOOT.INP > NUL
COPY A:ALTCHAR.SYS  I: > NUL
COPY A:CHKDSK.COM   I: > NUL
COPY A:DEBUG.COM    I: > NUL
COPY A:LABEL.COM    I: > NUL
COPY A:LOCATE.COM   I: > NUL
COPY A:PAGE.EXE     I: > NUL
COPY A:SDIR.COM     I: > NUL
COPY A:BSE.EXE      I: > NUL
COPY A:EDIT.PGM     I: > NUL
COPY A:PT.EXE       I: > NUL
COPY A:PT.PGM       I: > NUL
COPY A:PTHELP.DAT   I: > NUL
COPY A:PTMENU.PGM   i: > NUL
COPY A:PRINT.PGM    I: > NUL
COPY A:DBASE.COM    I: > NUL
COPY A:DBASEOVR.COM I: > NUL
SDIR/A
```

*

RAM Technology Products

All RAM Technology products include Hardware, Installation Instructions, and One Year Warranty. No soldering is required.

SMARTWATCH: Real-time no-slot clock/calendar module with 10 year lithium battery for the Z-100, all Z-100 PC series computers, PC/XT compatibles and most other MS-DOS computers. Includes software with source code. **$39.95**
Spacer Kit For Z-100 Computers. **$2.00**

Z-100 1.2 MB External Floppy System: Utilize your Z-100's built-in 8" controller for high capacity storage and back-up. System includes drive(s), cables, external power supply and cabinet.
One 5 1/4" 1.2 MB High Density Floppy Disk Drive system. **$249.00**
Two 5 1/4" 1.2 MB High Density Floppy Disk Drive system. **$439.00**

Z-MAX 100: 768K Memory Upgrade for Z-100's with new motherboards; #181-4918 or greater, simple plug-in installation. **$59.95**

Z-MAX 148: 704K Memory Upgrade for the H/Z138/148 computer. **$19.95**

Z-MAX 150: 640/704K Memory Upgrade for H/Z 150/160 and Eazy 150 series computers. Installs on your existing memory board. **$24.95**

Z-MEG 150: 1.2 MB. Memory Upgrade for H/Z 150/160 and Eazy 150 series computers. Up to 704K of addressable memory and 512K RAM DISK can be installed on your existing memory board. Please specify 640K EGA Compatible or 704K version when ordering. RAM-DSK software included. **$49.95**

Z-MEG 171: 1 MB. Memory Upgrade for the ZFL-171. Up to 640K of addressable memory and a 384K RAM DISK can be installed on your existing memory board. Software included. **$89.95**

Z-181 External 5 1/4" Disk Drive: 360K DS/DD 48 TPI Disk Drive mounted in half height enclosure. Interface Cable included. Please specify the model number of your computer: ZFL-181-92, ZFL-181-93 or ZWL-183-92. .... **$249.00**

FREE SHIPPING ON ORDERS OVER $100.00

HARD DISK DRIVES & CONTROLLERS

All Hard Disk Drives come with Installation Instructions, hardware, cables and a One Year Warranty. All controllers capable of handling two drives unless otherwise specified.

PC Compatible Hard Disk Systems

ST-125: 20 MB.. 28 ms., Half Height Hard Disk with Controller.	**$359.00**
ST-225: 20 MB., 65 ms., Half Height Hard Disk with Controller.	**$299.00**
ST-238R: 30 MB., 65 ms., Half Height Hard Disk with RLL Controller.	**$339.00**
ST-251: 40 MB., 40 ms., Half Height Hard Disk with Controller.	**$539.00**
ST-251R: 44 MB., 40 ms., Half Height Hard Disk with RLL Controller.	**$569.00**
ST-157R: 47 MB., 28 ms., Half Height Hard Disk with RLL Controller.	**$599.00**
ST-277R: 65 MB., 40 ms., Half Height Hard Disk with RLL Controller.	**$619.00**

Bare and AT Compatible Hard Disk Drives

ST-125: 20 MB.. 28 ms., Half Height.	**$299.00**
ST-225: 20 MB., 65 ms., Half Height.	**$269.00**
ST-238R: 30 MB., 65 ms., Half Height.	**$299.00**
ST-251: 40 MB., 40 ms., Half Height.	**$479.00**
ST-251R: 44 MB., 40 ms., Half Height.	**$669.00**
ST-157: 47 MB., 28 ms., Half Height.	**$499.00**
ST-277R: 65 MB., 40 ms., Half Height.	**$699.00**
ST-4038: 30 MB., 40 ms., Full Height.	**$529.00**
ST-4051: 40 MB., 40 ms., Full Height.	**$619.00**
ST-4053: 45 MB., 28 ms., Full Height.	**$639.00**
ST-4096: 80 MB., 28 ms., Full Height.	**$1019.00**
ID-40: 40 MB., 25 ms., Full Height.	**$779.00**
ID-45: 44 MB., 25 ms., Half Height.	**$799.00**
ID-100R: 103 MB., 25 ms., Full Height W/Controller.	**$1225.00**
ID-230R: 230 MB., 25 ms., Full Height W/Controller.	**$2550.00**

Hard Disk Controllers: We carry Adaptec, DTC, OMTI, Seagate and Western Digital MFM, SCSI and RLL Controllers for PC's , PC/AT's and compatibles in stock. Call for additional information and pricing.

RAM Technology Hard Cards

RAMDISK-20: 20 MB. Hard Card.	**$389.00**
RAMDISK-30: 30 MB. Hard Card.	**$439.00**
RAMDISK-40: 47 MB. Hard Card.	**$599.00**

RAM Technology

427-3 Amherst Street ▪ Suite 265
Nashua, New Hampshire 03063

603 889-0633

Hours Of Operation: Monday - Friday 9:00 A.M. - 5:00 P.M.

Z100 UPGRADE ACCESSORIES

RAM Technology carries a complete selection of upgrades and accessories for your Z-100 computer. Call for pricing and availability.

C.D.R. ZS-100: 7.67 Mhz. Speed Upgrade, no soldering required.	**$44.95**
CDR 317 Controller Kit: Interface And Controller.	**$575.00**
CDR 317-IIB SCSI Interface: For use with SCSI Hard Disk Drives.	**$335.00**

SMARTWATCH: See description under RAM Technology products. **$41.95**

Z-100 1.2 MB. External Floppy Disk Drive Sub-systems: See description under RAM Technology Products. .... From **$249.00**

Z-MAX 100: See description under RAM Technology Products. **$59.95**

FBE RESEARCH ZMF-100A: 768K Memory Upgrade for old motherboards; #181-4917 or lower; no soldering required. **$59.95**

UCI EASYRAM: 2 MB. Memory Expansion Board.	**$320.00**
UCI EASYDRIVE: RAM Disk Softwarefor UCI EASYRAM.	**$45.00**
UCI EASYIO/2: Provides 2 PC Compatible Serial Ports.	**$125.00**
UCI EASY PC: PC emulator board.	**$480.00**
UCI EASYWIN: Hard Disk Controller Interface.	**$245.00**
UCI EASYWIN: Hard Disk Interface And Controller.	**$335.00**
UCI EASY 87: Math Co-Processor Board.	**$85.00**

Z-100 Hard Disk Drive Sub-Systems

Includes Installation Instructions, Hard Disk Controller, Cables, Hardware, Software and One Year Warranty.

20 MB. Internal Hard Disk Drive System With CDR Controller:	**$725.00**
40 MB. Internal Hard Disk Drive System With CDR Controller:	**$925.00**
65 MB. Internal Hard Disk Drive System With CDR Controller:	**$1075.00**

SERVICE ▪ SUPPORT ▪ QUALITY ▪ VALUE

DYNAMIC RAM MEMORY & PROCESSORS

Prices current 12/21/87 and subject to change. Call for current prices.

4164-150:	64K 150 ns. DRAM.	**$3.35**
41256-150:	256K 150 ns. DRAM.	**$9.65**
41256-120:	256K 120 ns. DRAM.	**$10.65**
41256-100:	256K 100 ns. DRAM.	**$11.00**
41256-80:	256K 80 ns. DRAM.	**$11.50**
8088-2:	8 MHz. Processor.	**$9.95**
8087-3:	5 MHz. Math Co-Processor.	**$129.00**
8087-2:	8 MHz. Math Co-Processor.	**$179.00**
8087-1:	10 MHz. Math Co-Processor.	**$249.00**
80287-6:	6 MHz. Math Co-Processor.	**$199.00**
80287-8:	8 MHz. Math Co-Processor.	**$290.00**
80287-10:	10 MHz. Math Co-Processor.	**$350.00**
80387-16:	16 MHz. Math Co-Processor.	**$619.00**
80387-20:	20 Mhz. Math Co-Processor.	**$929.00**

NEC V20: 8 MHz. 8088 replacement Microprocessor. Increases CPU efficiency 10-30%. Includes Installation Instructions, One Year Warranty. **$14.95**

NO SURCHARGE ON VISA AND MASTERCARD

MISCELLANEOUS UPGRADES

Just because you don't see it doesn't mean we don't carry it. Call for pricing and availability on the accessory of your dreams!

BOC-1000-PC: PC/XT 1 MB. memory expansion board, LIM/EMS compatible, expandable to 2 MB. with daughter board, 0K installed. **$159.00**

BOC-2000-AT: PC/AT 2 MB. memory expansion board, LIM/EMS compatible, expandable to 4 MB. with daughter board, 0K installed. **$179.00**

MICRON MEMORY EXPANSION BOARDS: 2 to 16 MB. memory expansion boards for PC's and PC/AT's. Call for futher information.

MICROSPEED FAST 88: Speed upgrade for PC's and Compatibles. Increases CPU speed up to 8 MHz. Speed is switchable via software. **$99.00**

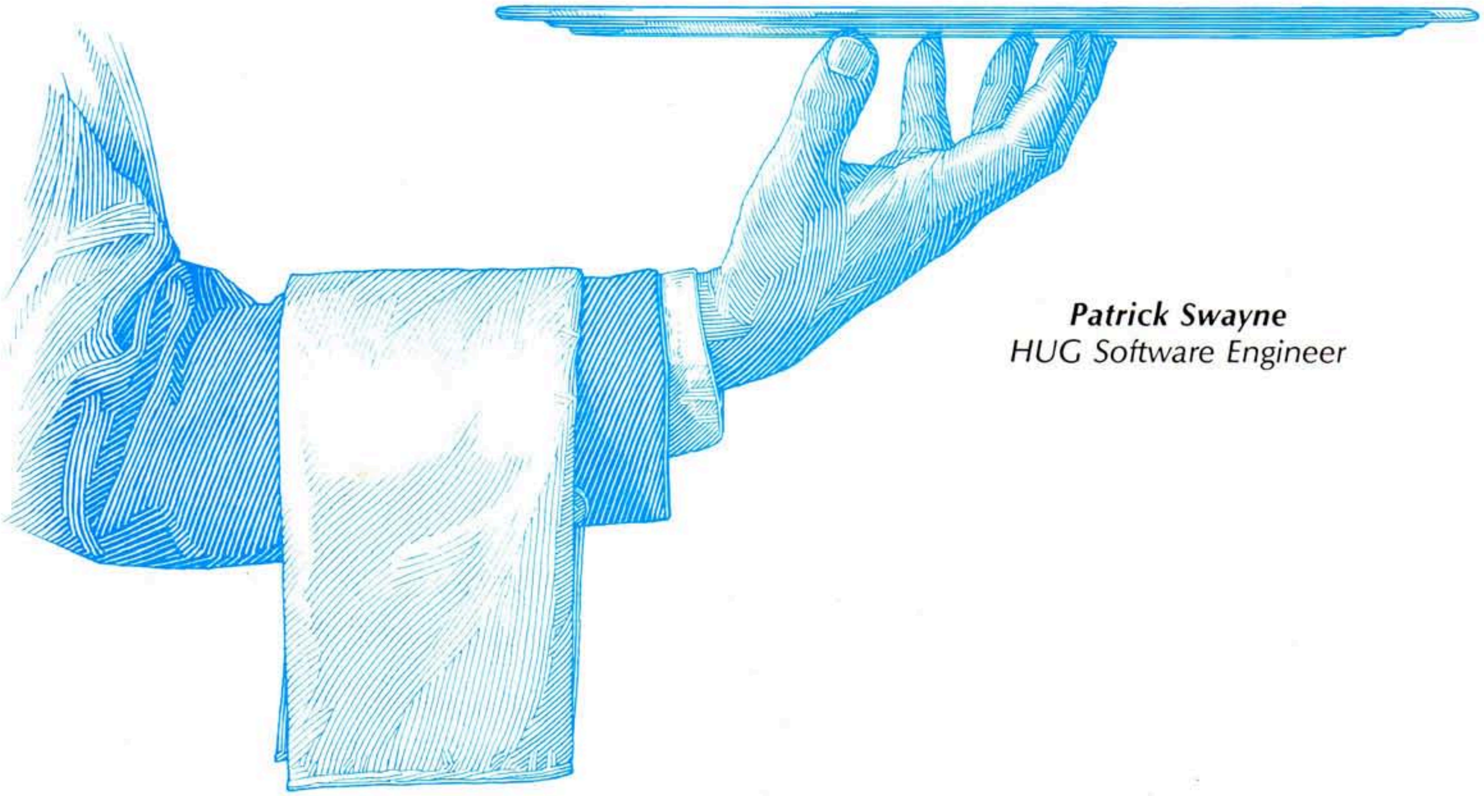
PTZ-148 DAUGHTERBOARD: Expansion board for Z-148's provides 2 expansion slots and clock/calendar. **$119.00**

3 1/2" Disk Drive Upgrades: For PC's and Compatibles including the Z-100, Z-100 PC, Z-200 PC and Z-300 PC series computers with standard 5 1/4" enclosures.
FDD-3530: 720K, for all systems, requires MS-DOS 3.2. .... **$139.00**
FDD-3550: 1.4 MB., for Z-200, Z-300 systems, requires MS-DOS 3.21. .... **$169.00**

The Fine Print

Prices and specifications subject to change without notice. Personal checks held 10 working days; money orders accepted as cash. COD orders accepted; cash or cashier's check only. No surcharge for VISA or MASTERCARD orders. Free shipping on orders over $100.00 via UPS in Continental U.S.A, all other orders are charged our cost of shipping (min. $3.00). Purchase orders accepted from qualified businesses, government agencies and educational institutions.

Simple But Elegant Menu System



Patrick Swayne
HUG Software Engineer

In the early days of personal computing, nearly everyone who owned a computer was dedicated not only to doing something with the computer, but to learning the computer itself. Today, however, many users are interested only in doing a particular job with the computer, and as a result there are many users who are unfamiliar with even the most rudimentary operating procedures. To make use of the computer easier, various menu systems have been devised. Some menu systems are general systems that try to satisfy everyone with a single structure, while others are configurable to a specific need. The configurable type of menu system is best for a new user, if an experienced user can set it up for him/her. The trouble is that many configurable menu systems are difficult to set up.

Batch Menu Systems

In the October 1987 issue of REMark, an article by Earl Zimmerman explained how you can make a menu system using "batch" files. The system presented there makes it easy to construct custom menus, and it works, but it is not efficient because all of those little batch files each take up 4k or more of disk space on a hard disk system, even if they only contain a few bytes of data. (For an explanation of disk space usage, see "On The Leading Edge" in the February 1988 issue.) It would be nice if a menu system could be devised using a single batch file to run the whole system.

In this article, I will present a way to do just that -- construct a menu system with a single batch file. The system is just as easy to configure as Mr. Zimmerman's menu system, and it is also powerful enough to do any job that more complex menu systems do. From a menu you construct, you can run programs, look at disk directories, change directories, run sub-menus from a master menu, and do generally anything that can be done by typing a command at the system prompt.

Using the Errorlevel Variable

What makes my menu system possible is a somewhat poorly documented variable called "errorlevel" available for use in batch program "if" statements that returns the exit code of the last program that was run. The exit code is a code that a program can pass on to the "shell" (usually COMMAND.COM, the MS-DOS command processor) under which it was run. The "errorlevel" variable lets a batch program test the exit code and make a decision based on its value.

Usually there is not much you can do with the "errorlevel" test, but I have written a little program called OPTION.COM that waits for the user to type a single character. When the character is typed, it is checked to see if it is within a specified range, and if it is, it is passed to the system as an exit code. To use OPTION, you just include a line in your batch file like this:

```
OPTION x-y
```

where x is the first character in the allowable range, and y is the last character in the range. The range can be a group of letters or numbers. For example, if you have OPTION 1-5, the acceptable characters are 1, 2, 3, 4, and 5. Similarly, if you have OPTION A-D, the allowable characters are A, B, C, and D, or a, b, c, and d (OPTION ignores case). If the user types a character that is outside of the specified range while OPTION is waiting for input, it will just beep and wait for another character.

You can also use OPTION to select from only two characters that are not adjacent, by placing the larger character first. For example, if you have OPTION Y-N, you are allowed to type only Y or N. (Y is considered greater than N, because it is farther down in the alphabet.)

The OPTION program turns the "errorlevel" variable into a useful tool that can be used to build a menu system. The source code for OPTION is listed at the end of this article, but you can also create it if you type in and run this BASIC program:

```
10 PRINT "CREATING OPTION.COM
20 OPEN "O" ,1, "OPTION.COM" :L=100
30 FOR I=1 TO 8 :C=0:FOR J=1 TO 11
40 READ B:C=C+B:PRINT #1,CHR$(B);:NEXT J:READ S
50 IF S<>C THEN PRINT " TYPING ERROR IN LINE" ;L:STOP
60 L=L+10:NEXT I:CLOSE #1:SYSTEM
100 DATA 190,93,0,128,60,32,117,2,205,32,177,1036
110 DATA 0,252,172,138,216,172,60,45,117,243,172,1587
120 DATA 138,248,58,216,114,2,177,1,81,180,8,1223
130 DATA 205,33,89,60,97,114,6,60,123,115,2,904
140 DATA 36,95,10,201,117,10,58,195,114,14,58,908
150 DATA 199,119,10,235,16,58,195,116,12,58,199,1217
160 DATA 116,8,178,7,180,2,205,33,235,210,138,1312
170 DATA 208,180,2,205,33,138,194,180,76,205,33,1454
```

Here is a very simple menu that illustrates the use of the OPTION program. Type the following lines into a file called TESTMENU.BAT. You can use a word processor that can create ASCII files, a text editor such as EDLIN, or the COPY CON procedure explained in your MS-DOS manual in the chapter on Batch Processing.

```
echo off
:menu
echo This is a test menu.
echo List files on drive A: .... 1
echo List files on drive B: .... 2
echo Exit from menu ............ 3
echo Please select 1, 2 or 3:
option 1-3
if errorlevel 51 goto exit
if errorlevel 50 goto option2
if errorlevel 49 goto option1
:option1
dir a:
goto wait
:option2
dir b:
:wait
pause
goto menu
:exit
```

Place the file TESTMENU.BAT and the OPTION.COM program in your root directory or in a directory that your PATH description points to. To run the test menu, type TESTMENU at the system prompt, and press Return.

There are two things you should be aware of about the test menu. The first is that the arguments used in the "if errorlevel" tests are the ASCII values of the numbers 3, 2, and 1. To get the ASCII values of other numbers and letters, refer to Table 1 in this article. The second thing you should be aware of is that the tests for the numbers are done in descending order. That is because the "if errorlevel" test does not return a true indication for the specified number only, but for that number and any higher number. So if you put the tests in ascending order, then any number you typed within the specified range would cause the item in the menu corresponding to the lowest number to be selected.

Menu System Improvements

As you can see, if you typed in the test menu and ran it, the menu system works quite well, but so far it is not very "elegant". It would look better if the list of items in the menu were spaced somewhat and listed in the center of the screen. We can take care of that by putting a CLS statement right after the label :menu, by adding a few tabs before each line in the menu list, and by adding a few blank lines in the list. It is really not possible to add a blank line using the ECHO command, but you can say ECHO, which will just put a period at the left edge of the screen, with the rest of the line blank. Better yet, you can put the menu list in a box of asterisks so that blank lines in the list will still have asterisks in them. I will show you what I mean later.

Another problem with the appearance of the menu is that the cursor is not at the end of the prompt (the "Please select" line), but on the next line when the menu is run. To solve that problem, I wrote another little program that can be used in menu batch files. This program is called LOCATE.COM, and you can make it by typing in and running the following BASIC program:

```
10 PRINT "CREATING LOCATE.COM
20 OPEN "O",1,"LOCATE.COM":L=100
30 FOR I=1 TO 8 :C=0:FOR J=1 TO 12
40 READ B:C=C+B:PRINT #1,CHR$(B);:NEXT J:READ S
50 IF S<>C THEN PRINT "TYPING ERROR IN LINE";L:STOP
60 L=L+10:NEXT I:CLOSE #1:SYSTEM
100 DATA 161,93,0,60,32,117,2,205,32,232,55,0,989
110 DATA 138,240,161,109,0,60,32,116,242,232,43,0,1373
120 DATA 138,208,184,64,0,142,216,128,62,0,0,233,1375
130 DATA 14,31,116,8,180,2,183,0,205,16,205,32,992
140 DATA 134,214,129,194,32,32,137,22,89,1,186,87,1257
150 DATA 1,180,9,205,33,205,32,128,252,32,117,3,1197
160 DATA 44,48,195,45,48,48,138,236,177,10,246,225,1460
170 DATA 2,197,195,27,89,0,0,36,0,0,0,0,546
```

To use LOCATE, you just include a line in your batch file like this:

```
LOCATE r,c
```

where r is the row the cursor is to be on, and c is the column. This program considers the upper left corner of the screen to be 0,0 instead of 1,1 as BASIC does, so you should take that into consideration when you count rows and columns. LOCATE runs on any PC-compatible computer or on a Z-100 series computer. The source code is listed at the end of this article.

Here is a sample menu that uses all of the techniques I have discussed so far:

```
echo off
:menu
cls
echo [3 tabs]******************************
echo [3 tabs]* Super Simple Menu System   *
echo [3 tabs]*         Test Menu          *
echo [3 tabs]*                            *
echo [3 tabs]* Show directory of A: ... 1 *
echo [3 tabs]*                            *
echo [3 tabs]* Show directory of B: ... 2 *
echo [3 tabs]*                            *
echo [3 tabs]* Check disk in A: ....... 3 *
echo [3 tabs]*                            *
echo [3 tabs]* Check disk in B: ....... 4 *
echo [3 tabs]*                            *
echo [3 tabs]* Exit from menu system .. 5 *
echo [3 tabs]*                            *
echo [3 tabs]* Enter your selection:      *
echo [3 tabs]******************************
locate 14,48
option 1-5
cls
if errorlevel 53 goto exit
if errorlevel 52 goto opt4
if errorlevel 51 goto opt3
if errorlevel 50 goto opt2
if errorlevel 49 goto opt1
:opt1
dir a:
goto wait
:opt2
dir b:
goto wait
:opt3
chkdsk a:
goto wait
:opt4
chkdsk b:
:wait
pause
goto menu
:exit
echo Type MENU2 to return to the menu system
```

Type the above lines into a file called MENU2.BAT When you type them, press the Tab key 3 times whenever you see [3 tabs] in the listing. Be sure to put one space after the word echo even when you are typing the lines with tabs. To run the new menu, put MENU2.BAT, OPTION.COM, and LOCATE.COM in your root directory, or in a directory pointed to by your PATH description. Then type MENU2 at the system prompt, and press Return.

Adding Color

When you run MENU2, you will notice that it looks much better on the screen than the original test menu. However, there is something else that can be done to improve screen appearance. You can add color to the menu. One way to add color is to include the appropriate escape sequences that produce color changes. On a PC-compatible computer, you would have to load ANSI.SYS and use the ANSI escape sequences for color changes. On a Z-100, you could just use the built-in Z-100 escape sequences.

Another way to add color is to use a little program I have written that allows you to change the foreground and background colors on the entire screen with one command. This program is called COLOR.COM, and you can make it if you type in and run this BASIC program:

```
10 PRINT "CREATING COLOR.COM
20 OPEN "O",1,"COLOR.COM":L=100
30 FOR I=1 TO 11 :C=0:FOR J=1 TO 12
40 READ B:C=C+B:PRINT #1,CHR$(B);:NEXT J:READ S
50 IF S<>C THEN PRINT "TYPING ERROR IN LINE";L:STOP
60 L=L+10:NEXT I:CLOSE #1:SYSTEM
100 DATA 161,93,0,60,32,117,2,205,32,232,83,0,1017
110 DATA 138,248,161,109,0,60,32,116,242,232,71,0,1409
120 DATA 138,216,184,64,0,142,216,128,62,0,0,233,1383
130 DATA 14,31,116,28,177,4,210,227,10,251,185,0,1253
140 DATA 0,186,79,24,184,0,6,205,16,186,0,0,886
150 DATA 183,0,180,2,205,16,205,32,139,195,37,7,1201
160 DATA 7,187,113,1,215,134,196,215,5,48,48,163,1332
170 DATA 123,1,186,121,1,180,9,205,33,205,32,128,1224
180 DATA 252,32,117,3,44,48,195,128,236,48,176,10,1289
190 DATA 2,196,36,15,195,0,1,4,5,2,3,6,465
200 DATA 7,27,109,0,0,27,69,13,36,0,0,0,288
```

Table 1
Table of ASCII Codes for 0-9 and A-Z (in decimal).

0	48	A	65	K	75	U	85
1	49	B	66	L	76	V	86
2	50	C	67	M	77	W	87
3	51	D	68	N	78	X	88
4	52	E	69	O	79	Y	89
5	53	F	70	P	80	Z	90
6	54	G	71	Q	81		
7	55	H	72	R	82		
8	56	I	73	S	83		
9	57	J	74	T	84		

To use COLOR, you just include a line in your batch file like this:

```
COLOR f,b
```

where f is the foreground color expressed as a number, and b is the background color. The color numbers are:

0	black	8	dark gray
1	blue	9	light blue
2	green	10	light green
3	cyan	11	light cyan
4	red	12	light red
5	magenta	13	light magenta
6	brown	14	yelow
7	light gray	15	white

This program works on both PC-compatible computers and Z-100 computers, but on a Z-100 you can only use color numbers 0-7, and 6 and 7 are yellow and white. On a PC-compatible, you can use colors 0-15 for the foreground, and colors 0-7 for the background. You can run this program by itself at the command prompt as well as in batch programs.

If you replace the first "cls" statement in MENU2 with

```
color 7,1
```

you will get a menu with light gray (white) characters on a blue background. You should NOT replace the second "cls" with a "color" command, because the "if errorlevel" tests will see the exit code from the COLOR program and not from the OPTION program. If you want to cause a color change when a menu item is selected, you should put the "color" statement in the section of the batch file that is executed for that item. For example, I could cause the directory of drive A: to be listed in green characters on a black background if I put a "color 2,0" statement right after the label :opt1 in MENU2.BAT.

Graphic Lines

There is one more thing that you may be able to do to improve the appearance of your menu. If you have a word processing program that can be used to draw graphic boxes on the screen, you may be able to use it to put a box around your menu list. The word processor must be able to output graphic lines in such a way that they will be able to be shown on the screen. Since the word processor is designed to put its final output on a printer, you may have to use a round-about method to produce a file which is plain ASCII except for some graphic lines. I only have experience with one word processor that can draw graphic lines: WordStar 4.0. The procedure for putting graphic boxes in your menu with WordStar 4.0 is as follows. Call the file MENU.TXT or something like that (do not use .BAT). Add these dot commands to the beginning of your file: .PO 0, .MT 0, .MB 0. Then print the file, and press Return to all of the Print questions until you get to "Name of printer?". Answer SIMPLE >MENU.BAT to this question, where MENU.BAT is the name of your menu. Load up MENU.BAT in the non-document mode, and remove the escape sequence at the beginning of the file. (The escape sequence will appear as "^[" followed by some characters at the beginning of your first line.) Also, remove the extra blank lines that will probably be at the end of the file by marking them as a block and using ^KY. Note: When you load in the MENU.BAT file produced by printing MENU.TXT, the graphic lines will no longer appear as lines on the screen, but as various text characters. That is because they are no longer in WordStar's internal format, but are the actual graphic characters. If you TYPE the file, the lines will appear as lines.

A Faster Menu

You can make the menu display itself on the screen faster by placing all of the screen text in a separate file. Instead of having a lot of "echo" statements in the batch file, you would have a single "type" command, such as

```
type menu.txt
```

The file menu.txt would contain the text to be displayed on the screen.

Menu Construction Hints

You have probably noticed that each item in MENU2 goes to a label called :wait after it executes its task. At :wait, there is a "pause" command, and then a "goto menu". Any time you run a program or command that leaves information on the screen to be read after control is returned to DOS, you should provide a "pause" command so that the user has a chance to read the information before the menu reappears. If you run something that is going to clear the screen or otherwise leave no information, you can go directly to the menu after it exits.

You can design sub-menus that can be loaded as options from a main menu just by using the name of the submenu as a command. If the submenus are in subdirectories, add a CD or CHDIR command before the name of the submenu. Just be sure that the main menu and the programs used by the menu system, OPTION, LOCATE, COLOR, etc., are in your root directory, or in a directory pointed to by your PATH description.

If you will be using menus on a floppy system, you may want to have an option to change disks as one of the menu items. To do that, you will need an additional batch file called CHANGE.BAT, which should contain these lines:

```
cls
echo Insert new disk in drive.
pause
menu
```

The menu file on each disk should be called MENU.BAT, or some other name that is the same on all disks, and the last line in CHANGE.BAT should be that name minus the .BAT extension.

Note: Batch processing under MS-DOS version 2.xx is so slow that a floppy-based menu system is not practical under that version. You should use MS-DOS version 3.xx for a floppy-based menu system, and it is best for hard disk use also. This system will not work at all with versions of MS-DOS (including Z-DOS) prior to version 2.xx.

To have your menu system start up automatically when you boot up your system, all you have to do is add the name of your menu (minus the .BAT) as the last line in an AUTOEXEC.BAT file on your system disk.

Although I have shown an option to exit from the menu system as one of the items in my sample menus, such an option is not needed. You can exit just by pressing Control-Break (PC-compatible only) or Control-C at the menu prompt, and then answer Y to the "Terminate batch job (Y/N)?" question. If you should get this question if you used Control-Break or Control-C to exit from a program, and you do not want to exit from the menu system, answer N.

Program Source Listings

Here are the source listings to the OPTION.COM, LOCATE.COM, and COLOR.COM programs.

```
        PAGE    ,132
;       THIS PROGRAM WAITS FOR THE USER TO INPUT
;       A KEY WITHIN A SPECIFIED RANGE, AND PASSES
;       THE ASCII VALUE OF THAT KEY (CAPITALIZED)
;       AS AN EXIT CODE.

CODE    SEGMENT
        ASSUME  CS:CODE,DS:CODE,ES:CODE,SS:CODE
        ORG     100H

START:  MOV     SI,OFFSET 5DH  ;POINT TO ARGUMENT
        CMP     BYTE PTR [SI],' '  ;ANY ARGUMENT?
        JNZ     GOTARG         ;YES
NOARG:  INT     20H            ;ELSE, EXIT
GOTARG:MOV      CL,0           ;CLEAR RANGE FLAG
        CLD
        LODSB                  ;GET BEGINNING OF RANGE
        MOV     BL,AL          ;SAVE IN BL
        LODSB                  ;GET ANOTHER CHARACTER
        CMP     AL,'-'         ;SHOULD BE THIS
        JNZ     NOARG
        LODSB                  ;GET END OF RANGE
        MOV     BH,AL          ;SAVE IN BH
        CMP     BL,AL          ;CHECK RANGE
        JB      GETCHR         ;IT'S A LEGAL RANGE
        MOV     CL,1           ;ELSE, MARK NOT A RANGE
GETCHR:PUSH      CX            ;SAVE RANGE FLAG
        MOV     AH,8
        INT     21H            ;GET USER'S PROMPT
        POP     CX
        CMP     AL,'a'         ;CHECK FOR UPPER CASE
        JB      NOTUC          ;IT'S NOT
        CMP     AL,'z'+1
        JNB     NOTUC
        AND     AL,5FH          ;ELSE, FORCE TO UPPER CASE
NOTUC:  OR      CL,CL           ;CHECKING A RANGE
        JNZ     NCR             ;NO
        CMP     AL,BL           ;ELSE, CHECK BOTTOM
        JB      BADRSP          ;BAD RESPONSE
        CMP     AL,BH           ;CHECK TOP
        JA      BADRSP
        JMP     SHORT EXIT      ;RESPONSE OK, EXIT
NCR:    CMP     AL,BL           ;IS AL = TO FIRST CHARACTER?
        JZ      EXIT            ;YES, EXIT
        CMP     AL,BH           ;OR TO SECOND CHAR?
        JZ      EXIT            ;YES, EXIT
BADRSP:MOV      DL,7
        MOV     AH,2
        INT     21H             ;BAD RESPONSE, RING BELL
        JMP     GETCHR          ;GET ANOTHER CHARACTER
EXIT:   MOV     DL,AL           ;CHARACTER TO DL
        MOV     AH,2
        INT     21H             ;ECHO CHARACTER
        MOV     AL,DL           ;GET CHARACTER
        MOV     AH,4CH
        INT     21H             ;EXIT WITH CHARACTER

CODE    ENDS
        END START
```

```
        PAGE    ,132
;       THIS PROGRAM LOCATES THE CURSOR
;       TO A USER-SUPPLIED LOCATION.
;
;       TO USE THIS PROGRAM, ENTER
;
;         LOCATE R,C
;
;       WHERE R IS A ROW NUMBER, AND C IS A
;       COLUMN NUMBER.
;
;       BY P. SWAYNE, HUG SOFTWARE ENGINEER  11-FEB-88

BIOS     SEGMENT AT 40H
         ORG 0
JMPTBL  LABEL BYTE ;            DEFIND Z-100 JUMP TABLE
BIOS     ENDS
```

```
CODE    SEGMENT
        ASSUME CS:CODE,DS:CODE,ES:CODE,SS:CODE
        ORG 5DH
ARG1    LABEL WORD              ; DEFINE ARGUMENT 1
        ORG 6DH
ARG2    LABEL WORD              ;DEFINE ARGUMENT 2
        ORG 100H

START:  MOV     AX,ARG1         ;GET ARGUMENT 1
        CMP     AL,' '          ;ANYTHING THERE?
        JNZ     GOTARG          ;YES
NOARG:  INT     20H             ;ELSE, EXIT
GOTARG:CALL     CONBIN          ;CONVERT TO BINARY
        MOV     DH,AL           ;ROW TO DH
        MOV     AX,ARG2         ;GET ARGUMENT 2
        CMP     AL,' '          ;ANYTHING THERE
        JZ      NOARG           ;IF NOT, EXIT
        CALL    CONBIN          ;CONVERT TO BINARY
        MOV     DL,AL           ;COLUMN TO DL
        MOV     AX,40H
        MOV     DS,AX           ;POINT TO BIOS RAM
        ASSUME  DS:BIOS
        CMP     JMPTBL,0E9H     ;CHECK FOR JUMP TABLE
        PUSH    CS
        POP     DS              ;FIX DS
        ASSUME  DS:CODE
        JZ      GOTZ            ;IT;S A Z-100
        MOV     AH,2            ;CURSOR POS. FUNCTION
        MOV     BH,0            ;PAGE 0
        INT     10H             ;POSITION CURSOR
        INT     20H             ;AND EXIT
GOTZ:   XCHG    DL,DH           ;EXCHANGE ROW, COLUMN
        ADD     DX,' '          ;ADD ASCII OFFSET
        MOV     CPOS,DX         ;STORE POSITION IN STRING
        MOV     DX,OFFSET POSSTR ;POINT TO POSITION STRING
        MOV     AH,9
        INT     21H             ;POSITION CURSOR
        INT     20H             ;AND EXIT

CONBIN: CMP     AH,' '          ;SINGLE DIGIT?
        JNZ     GOT2D           ;NO
        SUB     AL,'0'          ;ELSE, REMOVE ASCII
        RET                     ;AND RETURN
GOT2D:  SUB     AX,'00'         ;REMOVE ASCII
        MOV     CH,AH           ;SAVE LOW DIGIT
        MOV     CL,10
        MUL     CL              ;MULTIPLY HIGH DIGIT BY 10
        ADD     AL,CH           ;ADD LOW DIGIT
        RET

POSSTR  DB      27,'Y'
CPOS    DW      0
        DB      '$'
CODE    ENDS
        END     START

        PAGE    ,132
;       THIS PROGRAM SETS THE SCREEN COLOR
;       TO USER-SUPPLIED VALUES.
;
;       TO USE THIS PROGRAM, ENTER
;
;         COLOR F,B
;
;       WHERE F IS THE FOREGROUND COLOR, AND
;       B IS THE BACKGROUND COLOR.
;
;       BY P. SWAYNE, HUG SOFTWARE ENGINEER  11-FEB-88

BIOS    SEGMENT AT 40H
        ORG     0
JMPTBL  LABEL   BYTE            ;DEFIND Z-100 JUMP TABLE
BIOS    ENDS
```

```
CODE    SEGMENT
        ASSUME  CS:CODE,DS:CODE,ES:CODE,SS:CODE
        ORG     5DH
ARG1    LABEL   WORD            ;DEFINE ARGUMENT 1
        ORG     6DH
ARG2    LABEL   WORD            ;DEFINE ARGUMENT 2
        ORG     100H

START:  MOV     AX,ARG1         ;GET ARGUMENT 1
        CMP     AL,' '          ;ANYTHING THERE?
        JNZ     GOTARG          ;YES
NOARG:  INT     20H             ;ELSE, EXIT
GOTARG:CALL     CONBIN          ;CONVERT TO BINARY
        MOV     BH,AL           ;FOREGROUND TO BH
        MOV     AX,ARG2         ;GET ARGUMENT 2
        CMP     AL,' '          ;ANYTHING THERE
        JZ      NOARG           ;IF NOT, EXIT
        CALL    CONBIN          ;CONVERT TO BINARY
        MOV     BL,AL           ;BACKGROUND TO BL
        MOV     AX,40H
        MOV     DS,AX           ;POINT TO BIOS RAM
        ASSUME  DS:BIOS
        CMP     JMPTBL,0E9H     ;CHECK FOR JUMP TABLE
        PUSH    CS
        POP     DS              ;FIX DS
        ASSUME  DS:CODE
        JZ      GOTZ            ;IT'S A Z-100
        MOV     CL,4
        SHL     BL,CL           ;SHIFT BACKGROUND UP
        OR      BH,BL           ;ADD RESULT TO FOREGROUND
        MOV     CX,0            ;UPPER CORNER
        MOV     DX,24*256+79    ;LOWER CORNER
        MOV     AX,600H         ;CLEAR ENTIRE PAGE
        INT     10H             ;CLEAR PAGE WITH NEW COLOR
        MOV     DX,0
        MOV     BH,0
        MOV     AH,2
        INT     10H             ;HOME CURSOR
        INT     20H             ;AND EXIT
GOTZ:   MOV     AX,BX           ;COLORS TO AX
        AND     AX,707H         ;ENSURE THEY ARE IN RANGE
        MOV     BX,OFFSET COLMAP
        XLAT                    ;MAP BACKGROUND COLORS
        XCHG    AL,AH           ;SWAP BGND, FGND
        XLAT                    ;MAP FOREGROUND COLORS
        ADD     AX,'00'         ;ADD ASCII OFFSET
        MOV     COLOR,AX        ;STORE COLOR IN STRING
        MOV     DX,OFFSET COLSTR ;POINT TO COLOR STRING
        MOV     AH,9
        INT     21H             ;SET COLOR
        INT     20H             ;AND EXIT

CONBIN: CMP     AH,' '          ;SINGLE DIGIT?
        JNZ     GOT2D           ;NO
        SUB     AL,'0'          ;ELSE, REMOVE ASCII
        RET                     ;AND RETURN
GOT2D:  SUB     AH,'0'          ;FIX LOW DIGIT
        MOV     AL,10           ;ASSUME NOT > 19
        ADD     AL,AH           ;ADD LOW DIGIT
        AND     AL,0FH          ;ENSURE <= 15
        RET

COLMAP DB       0,1,4,5,2,3,6,7 ;Z-100 TO PC COLOR MAP
COLSTR DB       27;'m'
COLOR   DW      0
        DB      27,'E',13,'$'
CODE    ENDS
        END     START
```

*

Announcement!

HUG MEMBERS ONLY!!

The HUG-386 and HUG-386-C Upgrade Kits will be available shortly. Wheelin' Dealin' Jim has managed a super-fantastic deal on these two products for Heath Users' Group members who originally purchased an H-241 or H-248; ***one-thousand two-hundred dollars*** off the regular purchase price! That's right! If you originally purchased an H-241 or H-248, and you're a HUG member, you can get $1200.00 off the regular retail price of either of these two upgrade kits!

Now, how do you determine which upgrade kit to buy? If you have any version of the H-241 or H-248-S series of computer, then you will need to purchase the HUG-386-C Upgrade kit, which contains a new disk controller board. If you have a standard H-248, you will need the HUG-386.

Here are the three ways you can order your upgrade:

Write-In Orders

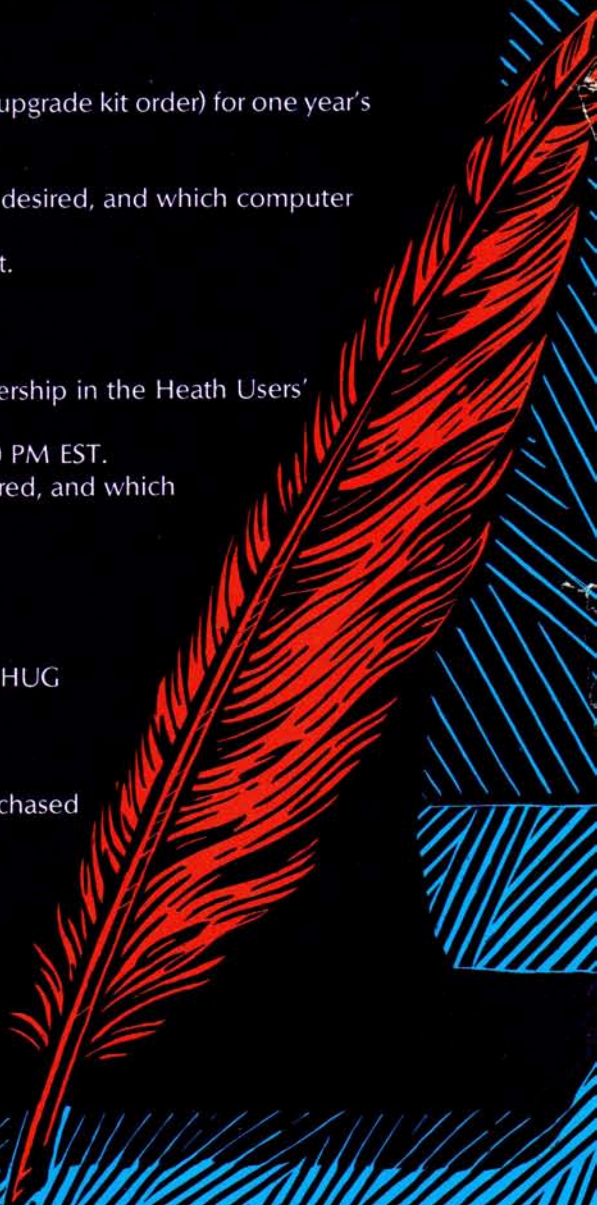
- Non-HUG members ***can*** order by including payment (with the upgrade kit order) for one year's membership in the Heath Users' Group.
- All orders should be submitted to the Heath Users' Group.
- Each order must indicate the model number of the upgrade kit desired, and which computer kit it was purchased for.
- Each order must have the persons HUG ID number written on it.

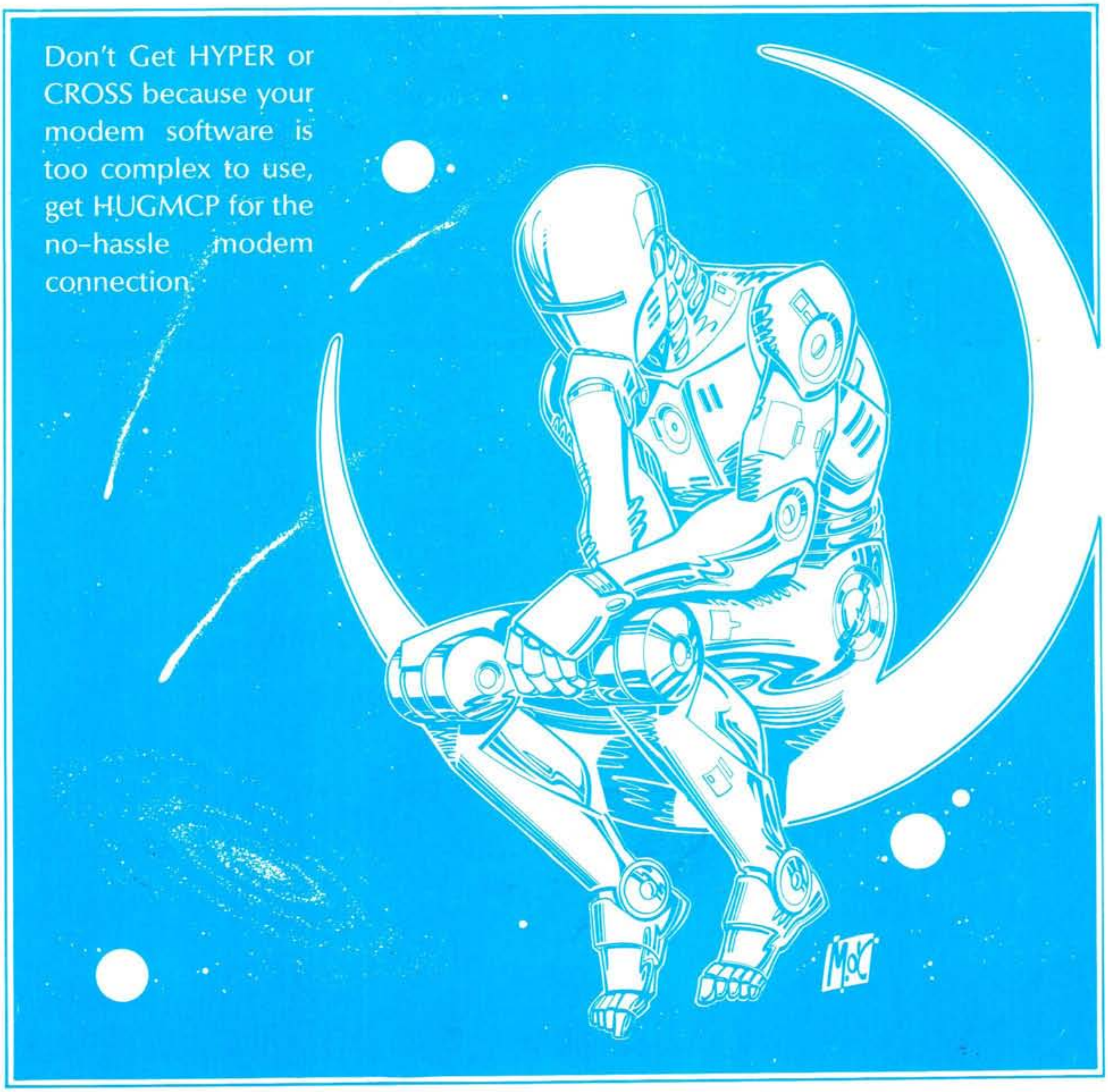
Phone-In Orders

- Non-HUG members ***can*** order by first ordering a one year's membership in the Heath Users' Group.
- All orders must be phoned in to (616) 982-3838 from 8 AM to 4:30 PM EST.
- Each order must indicate the model number of the upgrade kit desired, and which computer kit it was purchased for.
- The person ordering must supply his/her current HUG ID number.

Heath/Zenith Computer Store Sales

- Non-HUG members ***can*** purchase an upgrade kit by first purchasing a HUG membership from the store.
- Orders for the upgrade kit can be taken in the normal fashion.
- Each order must have the buyer's HUG ID number on it.
- Each order should indicate which computer kit the upgrade was purchased for.




Don't Get HYPER or CROSS because your modem software is too complex to use, get HUGMCP for the no-hassle modem connection.

Heath/Zenith Users' Group

P.O. Box 217
Benton Harbor, MI 49022-0217

POSTMASTER: If undeliverable, please do not return.