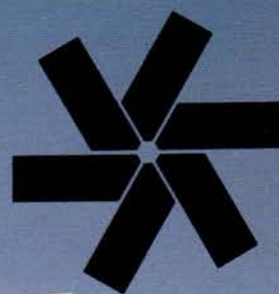


The Official **ZENITH** /Heath Computer Users Magazine

# REMark®



March 1989

**FIXING YOUR FLOPPY  
WITH HADES**

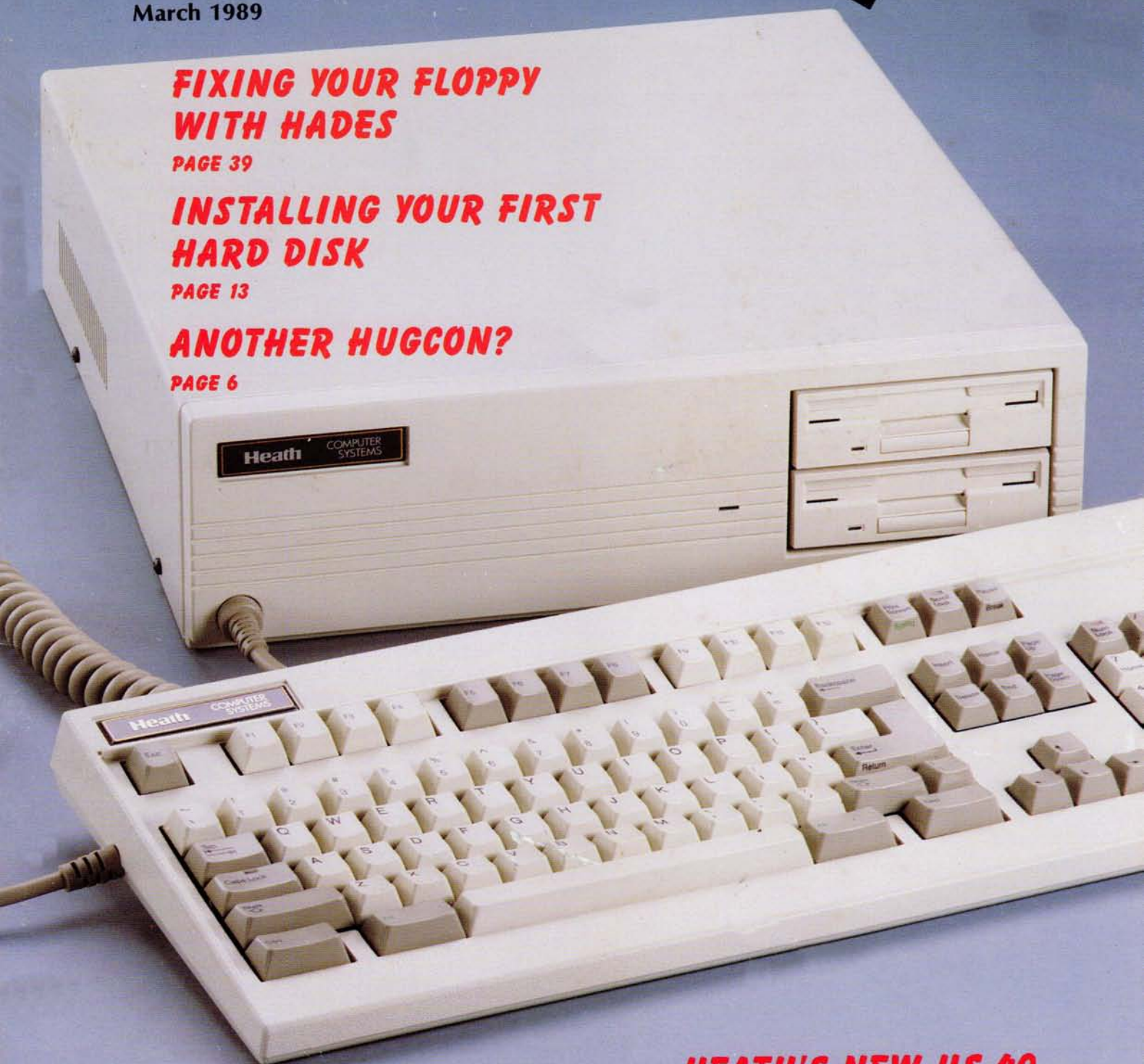
**PAGE 39**

**INSTALLING YOUR FIRST  
HARD DISK**

**PAGE 13**


**ANOTHER HUGCON?**

**PAGE 6**



**HEATH'S NEW HS-40**





## "How Can You Take Advantage of Me"

"... If you don't call? I have everything you could possibly want! My software selection continues to grow, and remains my most popular feature. I'm fast, but, if you don't have the time to download, these software disks can now be purchased for a small copying charge! My message base has also become quite popular. Through it, HUGgies are exchanging more information than ever before. Finally, there's my legendary Bargain Centre. It alone, will make you come back for more! Did you know how inexpensive I am? Why pay \$14 per hour connect time to someone else when your phone company charges less than \$12 per hour (less on weekends) from anywhere within the continental U.S.!

So, go ahead and take advantage of me.

Just set your modem for 300, 1200, or 2400 baud (8N1) and call (616) 982-3956.

You needn't type anything, I'll know you're there!"

MOC



# REMark®

Volume 10, Issue 3 • March 1989

**On the Cover:** Occupying roughly the same footprint as Zenith's largest (ZCM-1490) color monitor, it's hard to believe that this small package houses a full blown, 8 MHz (no-wait state), 80286 PC-AT computer system. This jewel comes configured with two 3-1/2" 1.4 MB floppies, and other standard features include: deluxe 101-key keyboard; not one, but two serial ports; a parallel port; and the new HVB-550 video card. This new video card will run CGA, EGA, MDA, Hercules, and true VGA! Additional options include: An external 5-1/4" floppy drive interface, a 20 MB or 40 MB Winchester hard disk, a 2 MB memory upgrade, and an 80287 Numeric Co-Processor. This powerful system is available in kit form as model number HS-40 and construction is a breeze (only 2 solder connections)! Configured with a single floppy and 40 MB Winchester, the wired Zenith version (model ZCF-2326-EY) is also available.

Reader Service No.	Page No.
101 American Cryptronics .....	64
168 BEA-SOFT Computers .....	20
104 FBE Research Company, Inc. ....	16
184 First Capitol Computer .....	25
191 First Capitol Computer .....	29
192 First Capitol Computer .....	40
105 First Capitol Computer .....	48
107 Paul F. Herman .....	60
108 Hogware .....	29
137 Jay Gold Software .....	25
136 Lindley Systems .....	20
117 Payload Computer .....	26
193 Quikdata, Inc. ....	40
121 Scottie Systems .....	60

## PC Compatibles

All models include the following series of computers: H/Z-130, 140, 150, 160, 170, 180, H/Z-200, and 300.

## PC Compatible

<b>POWERING UP</b> William M. Adney .....	9
<b>Softening the Hard Disk — Part 1</b> John A. Negus .....	13
<b>MS-DOS, OS/2, and the Numlock Key</b> Robert Metz .....	35
<b>SuperSporting and XyWriting Through the NBA</b> Glen Nelson .....	37
<b>Greatly Enhanced WordStar 2000 Plus V3.0</b> Richard L. Mueller .....	45
<b>Recovering Data Files</b> Raymond H. Murphy .....	55

## H/Z-100 and PC Compatible

<b>Is This Disk Formatted?</b> Pat Swayne .....	7
<b>ENABLE — Part 15</b> George P. Elwood .....	17
<b>On the Leading Edge</b> William M. Adney .....	21
<b>Menu Screens</b> Michael R. Gage .....	33
<b>Using HADES to Fix Damaged Floppies</b> Pat Swayne .....	39
<b>An ANSI Graphics Header File for C</b> Stephen A. Jacob .....	41

## H/Z-100 Only (Not PC)

<b>Z-100s, Batch Files, Virtual Disks and PeachTree Software</b> Rodney E. Cavin .....	57
--	----

## General

<b>Didactic Demonstration of Outlining with WordPerfect 5.0</b> William N. Campbell .....	27
<b>The Most Important Peripheral: Printers</b> William N. Campbell .....	49

## Resources

<b>HUG Price List</b> .....	2
<b>HUG New Products</b> .....	4
<b>Buggin' HUG</b> .....	5
<b>Bug Zapping</b> .....	30
<b>Classified Ads</b> .....	47
<b>Glitches</b> .....	47
<b>Heath/Zenith Related Products</b> .....	61

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) 982-3463

Contributing Editor ..... William M. Adney

Advertising ... Rupley's Advertising Service  
Dept. REM, 240 Ward Avenue  
P.O. Box 348  
St. Joseph, MI 49085-0348  
(616) 983-4550

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 Center 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/Zenith equipment. As such, little or no evaluation of the programs or products advertised in REMark. The Software Catalog, or other HUG publications is performed by Heath Company, in general, and HUG, in particular. The prospective user is hereby put on notice that the programs may contain faults, the consequence of which Heath Company, in general, and HUG, in particular, cannot be held responsible. The prospective user is, by virtue of obtaining and using these programs, assuming full risk for all consequences.

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

Copyright © 1989, Heath/Zenith Users' Group

# HUG

PRODUCT NAME	PART NUMBER	OPERATING		PRICE
		SYSTEM	DESCRIPTION	
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
ASCIRITY	885-1238-[37]	CPM	AMATEUR RADIO	20.00
AUTOFIRE (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	30.00
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
MAGBASE	885-1249-[37]	CPM	MAGAZINE DATABASE	25.00
MAPLE	885-8005	HDOS	COMMUNICATION	35.00
MAPLE	885-8012-[37]	CPM	COMMUNICATION	35.00
MICRONET CONNECTION	885-1122-[37]	HDOS	COMMUNICATION	16.00
MISCELLANEOUS UTILITIES	885-1089-[37]	HDOS	UTILITY	20.00
MORSE CODE TRANSCIVER	885-8016	HDOS	AMATEUR RADIO	20.00
MORSE CODE TRANSCIVER	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
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
COPYDCS	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
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



# Price List

PRODUCT NAME	PART NUMBER	OPERATING SYSTEM	DESCRIPTION	PRICE
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 II	885-3049-37	MSDOS	PRINTER PLOTTING UTILITY	25.00
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	885-3018-37	MSDOS	VARIOUS SPREADSHEETS	25.00
TREE-ID	885-3036-37	MSDOS	TREE IDENTIFIER	20.00
USEFUL PROGRAMS I	885-3022-37	MSDOS	UTILITIES	30.00
UTILITIES	885-3008-37	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	MSDOS	GAME	10.00
ASSEMBLY LANGUAGE UTILITIES	885-8046	MSDOS	UTILITY	20.00
BOTH SIDES PRINTER UTILITY	885-3048	MSDOS	UTILITY	20.00
CXREF	885-3051	MSDOS	UTILITY	17.00
DEBUG SUPPORT UTILITIES	885-3038	MSDOS	UTILITY	20.00
DPATH	885-8039	MSDOS	UTILITY	20.00
HADES	885-3040	MSDOS	UTILITY	40.00
HELP	885-8040	MSDOS	CAI	25.00
HEPCAT	885-3045	MSDOS	UTILITY	35.00
HUG BACKGROUND PRINT SPOOLER	885-3029	MSDOS	UTILITY	20.00
HUG EDITOR	885-3012	MSDOS	TEXT PROCESSOR	20.00
HUG MENU SYSTEM	885-3020	MSDOS	UTILITY	20.00
HUG SOFTWARE CATALOG UPDATE #1	885-4501	VARIOUS	PROD 1983 THRU 1985	9.75
HUGMCP	885-3033	MSDOS	COMMUNICATION	40.00
HUGPBBS SOURCE LISTING	885-3028	MSDOS	COMMUNICATION	60.00
HUGPBBS	885-3027	MSDOS	COMMUNICATION	40.00
ICT 8080 TO 8088 TRANSLATOR	885-3024	MSDOS	UTILITY	20.00
MAGBASE	885-3050	VARIOUS	MAGAZINE DATABASE	25.00
MATT	885-8045	MSDOS	MATRIX UTILITY	20.00
MISCELLANEOUS UTILITIES	885-3025	MSDOS	UTILITIES	20.00
PS's PC & Z100 UTILITIES	885-3052	MSDOS	UTILITY	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	885-4008	N/A	1987	25.00
SCREEN DUMP	885-3043	MSDOS	UTILITY	30.00
UTILITIES II	885-3014	MSDOS	UTILITY	20.00
Z100 WORDSTAR CONNECTION	885-3047	MSDOS	UTILITY	20.00

## PC Compatibles

ACCOUNTING SYSTEM	885-8049	MSDOS	BUSINESS	20.00
CARDCAT	885-6006	MSDOS	CATALOGING SYSTEM	20.00
CHEAPCALC	885-6004	MSDOS	SPREADSHEET	20.00
CP/EMULATOR II & ZEMULATOR	885-6002	MSDOS	CPM & Z100 EMULATORS	20.00
DUNGEONS & DRAGONS	885-6007	MSDOS	GAME	20.00
EZPLOT II	885-6013	MSDOS	PRINTER PLOTTING UTILITY	25.00
GRADE	885-8037	MSDOS	GRADE BOOK	20.00
HAM HELP	885-6010	MSDOS	AMATEUR RADIO	20.00
KEYMAP	885-6001	MSDOS	UTILITY	20.00
PS's PC UTILITIES	885-6011	MSDOS	UTILITIES	20.00
SCREEN SAVER PLUS	885-6009	MSDOS	UTILITIES	20.00
SKYVIEWS	885-6005	MSDOS	ASTRONOMY UTILITY	20.00
TCSPELL	885-8044	MSDOS	SPELLING CHECKER	20.00
ULTRA RTTY	885-6012	MSDOS	AMATEUR RADIO	20.00

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.

Magazines everywhere, and no way to reference the wealth of information they hold? Not anymore! Now there's **MAGBASE**; a database designed specifically for referencing magazine articles. Don't let those one-hundred-and-some back issues of REMark, or C Users Journal, or Veterinary Medicine, (or any magazine) gather dust, use **MAGBASE**, and find that article you read two years ago! **MAGBASE** is available for **MSDOS HUG P/N 885-3050** or **CP/M (P/N 885-1249-[27])**.

**LAPTOP OWNERS** . . . don't feel left out! All of HUG's MSDOS software is available on 3-1/2" micro-floppies too! When ordering, just add a "-80" to the 7-digit HUG part number. For the standard 5-1/4" floppy, just add a "-37".

**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!**

## ORDERING INFORMATION

For VISA and MasterCard phone orders, telephone the Heath Users' Group directly at (616) 982-3463. 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.





# HUG NEW PRODUCTS



- 10 - Very Good
- 9 - Good
- 8 - Average

## TABLE C Product Rating

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

When ordering any version of MSDOS software, you must specify what type of media you want the software supplied on. If you want 5-1/4" floppies, add a "-37" to the 7-digit part number. If you want 3-1/2" micro-floppies, add a "-80" to the 7-digit part number.

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.**

## HUG P/N 885-8040 HELP Update ..... \$25.00

**Introduction:** The author of HELP for MS-DOS, has updated it to Version 3.30b. This new version contains many enhancements, and now includes complete coverage of Zenith's new MS-DOS Version 3.3 Plus. Laptop computer information has also been included. The H/Z-100 (not PC Compatible computer) portion of HELP remains unchanged. Original owners of HELP can obtain an updated disk by returning their original distribution disk to the Heath Users' Group, along with \$7 (made out to HUG). REMEMBER, Laptop owners can obtain this software on a 3-1/2" disk just by adding a "-80" to the basic HUG model number (885-8040-80).

The files on Disk A: are for the PC Compatible version of HELP. The following is a list of those files.

**HELP.EXE** — Executable version of the PC Compatible HELP program.

**HELP.DAT** — Random Access HELP data base file.

**HELPS.DAT** — Sequential HELP data base file.

**HELP.DOC** — Documentation file for HELP.

**HELPCNVT.EXE** — Executable version of PC Compatible conversion program.

**README.DOC** — HUG Documentation file for this disk.

The files on Disk B: are for the H/Z-100 (not PC) version of HELP. The following is a list of those files:

**HELPCNVT.BAS** — BASIC source code for data base conversion program.

**HELPS.DAT** — Sequential HELP data base file.

**HELP.DAT** — Random access HELP data base file.

**HELP.DOC** — Documentation file for HELP.

**README.DOC** — HUG Documentation file for this disk.

**HELP.BAS** — BASIC source code for Z-100 HELP program.

**HELP.EXE** — Executable version of Z-100 HELP program.

**HELPCNVT.EXE** — Executable version of Z-100 conversion program.

The sorted directory utility programs are provided strictly as a convenience to the user and are not required to use the HELP programs.

**Program Author:** John F. Stetson

**Program Content:** The HELP program are designed to be easy to use and efficient in operation. The HELP data base is a BASIC random-access file for high speed access, even on floppy-based systems. Over 100 entries are presented using a full-screen HELP menu and may be easily selected using the keyboard arrow keys. The commands in the HELP data base are divided into the following functional categories:

**Reference Commands** — General reference information for MS-DOS concepts and capabilities.

**Resident Commands** — Information for commands which are resident in memory.

**Resident Command Aliases** — Information for command aliases for the resident commands.

**Resident Batch Processing Commands** — Information for commands which are used in batch processing.

**Transient Utilities** — Information for commands which are part of the Zenith MS-DOS Programmer's Utility Pack.

\* Once a command is selected, the following information is displayed:

**Command** — General information about the command: alias names; Z-100 vs. Z-100 PC; MS-DOS V2 vs. MS-DOS V3; etc.

**Function** — Brief description of the function or purpose of the command.

**Syntax** — Complete, detailed command line syntax of the command including all file names, option switches, etc.

Continued on Page 42



# BUGGIN' HUG

## Solution for Z-100's Inability to Boot

Dear HUG:

In a recent article in the REMark magazine, one of your readers described a problem with the Z-100's inability to boot from any disk other than A drive. I believe that I have a partial solution to the boot problem. My personal computer is a dual floppy Z-100 which has been upgraded with a 2 megabyte UCI ram board which can be configured as 1 to 8 ram drives designated as drives I-P. In my situation, I must initially boot from A drive, but then I usually transfer my working programs to I drive and my data files to J drive on the ram disk. However, the operating system insisted on having Command.com on A drive even though I installed a copy of Command.com on I drive. Two DOS Commands can be used to overcome this problem. The set command can be used to define an alternate program can be used in place of Command.com. Second, the map command can be used to redefine physical to logical drive assignment. The standard drive definitions are as follows:

A = 0 , B = 1 , ... Z = 25

Therefore, the following sequence can be used to force the operating system to operate entirely from the desired disk without going back to A drive to access command.com.

```
COPY A:COMMAND.COM = X:COMMAND.COM
COPY A:MAP.COM = X:MAP.COM
SET COMSPEC = X:COMMAND.COM
MAP A:N Y:0
X:
PROGRAMNAME
MAP A:0
A:
```

Where X is the letter designation of the desired operating disk (I, in my case) and N is the number designation of drive X, number 8, in my case, for drive I. Y is another unused drive letter. If unit 0 is not reassigned to an unused drive, it will be inaccessible. That solved my problem using DOS 2.2.

The only problem with this procedure is that FORMAT will not work once MAP is installed unless you reboot. For some reason, reassigning the drives back to the default configuration does not satisfy the FORMAT program. There is no way to uninstall the Map program under DOS 2.2.

Also according to the Z-100 users manual, it is possible to override the standard boot-up procedure from a drive by manually booting as follows:

Turn on the machine with no disk in A drive (assuming A is a floppy drive).

Wait for the hand prompt.

Type B (for boot)

Then, one of the function keys F1, F2, F3 to select drive type.

F1 = a or b : 5-1/4" floppy drives

F2 = c or d : 8 inch floppy drives

F3 = e, f, g, h : hard drive partitions

Finally, type:

0 for a, 1 for b

0 for c, 1 for d

0, 1, 2, 3 for e, f, g, h

I hope this information may be of some use to your readers.

Sincerely,

Ronald L. Nave  
318 Whitmer Road  
Horsham, PA 19044

## CHUGCON88

Dear Jim:

It's with great sorrow and distress that I left CHUGCON88. I had high expectations of another superb conference and swap meet, only to find a smaller version of the usual DC Computer Symposium/Exposition — and a completely changed attitude. After I returned to North Carolina nearly empty-handed, I had the opportunity to read the August '88 edition of >CHUG, the monthly magazine of the Capital Heath/Zenith Users' Group. What I read caused chills up and down my spine. I quote:

"This is the seventh Exposition we have held and the first where we have gone from a specialized User Group to a path cutting across the entire PC spectrum. Visitors who have been at past CHUGCONs are seeing a different array of vendors and exhibitors and may notice that some of the old standbys are not here."

"There has been, out of necessity, a change of philosophy at CHUG brought on by changes in the Heath/Zenith community, as well as changes at Zenith Data Systems. No longer are the major Heath/Zenith machines different from the rest of the industry. As much as we felt the H/Z-100 was a very powerful and versatile computer, it is out of production and spares for the existing machines are the units out in the field. The old standby H-8s and '89s that many of us cut our teeth on are getting few and far between, CP/M is not known by many of the people now in the computer industry. (In a short five years they are museum pieces.)"

"The industry today is MS-DOS, OS-2, and UNIX. The Apple Macintosh now talks to MS-DOS machines and all talk to and use each other's data. If you listen to the CHUG members' comments about

what they are using at work and at home, the list covers all makes and models of PCs. With this in mind and wanting to give the Washington area the benefit of as many different Vendors as we could, the CHUG Board changed the scope of CHUGCON88 to the Exposition you are attending."

"Some of the old faces that specialized in Heath/Zenith specific hardware and software are not here. These hardware vendors that supplied many of the Heath/Zenith surplus boards are missing, as there is nothing coming in the way of surplus for them to sell. With the price of memory, any board they had has been sold for memory chips. The Zenith factory is not selling any surplus or drop-outs, so there is nothing new in the pipeline. Zenith has also sent much of the refurbished and last year's model products to new stores as advertising for the store openings, so the local Heath/Zenith stores do not have the merchandise to sell as we have seen in the past."

While I can rather heavy-heartedly accept the Heath stores gradually devoting more and more floor space to Apple/Macintosh machines and perhaps the elimination of surplus equipment sales at their own computer conference (that hurt), I feel the change in the conference format may have unintentionally struck a death blow to the Heath/Zenith hobbyist. The DC area already has an annual Computer Exposition. Neighboring Baltimore also has its own. Both cover the complete spectrum of computers and accompanying peripherals. Both are several times larger than Heath/Zenith could ever hope to do. They show latest technology, new uses, new companies. They don't, however, provide the hobbyist with used equipment, computer specific parts, and the opportunity to talk to vendors who know the "museum equipment" inside out. This latest HUGCON nearly failed in the same regard.

Whether it was the high cost of booth space, or a decision of CHUG to discourage the smaller vendors and surplus dealers, the result was the same. Gone were the deep discounts, gone were the opportunities to hand pick, cheap used equipment, gone were the bargain hunting crowds (the lack of parking didn't help this either). I have two H-89s, an H-100 and a new Z-181. They are all thankfully in excellent shape and I use the H-89s more than ever. I shudder at the prospect of having to maintain these machines by mail order, unable to hand pick spare boards and discuss problems and possible solutions.

I plead that the hosts of the next Heath/Zenith conference not try to imitate or compete with the other expositions (those vendors are still welcome), but return to the former format, encour-



age participation by the little guys by reducing or waiving booth fees, and let's try to maintain a "different", hobbyist attitude. It will be a lot more fun.

Sincerely,

Steven W. Vagts  
2409 Riddick Road  
Elizabeth City, NC 27909

### A Fix for a Patch

Dear Dr. Mueller:

Reference your "Patch for MASM V4.0 Problem" in the August 1988 issue of REMark magazine, page 45-46.

I made your patch as per the above article and everything seemed to work fine — that is until last week. After a little investigation, I found that I did not get the annoying "extra characters on line" error message so long as the INCLUDE files always contained the multiple end-of-file bytes (1A Hex) padded out to the end of the block. However, the message returned again if the INCLUDE files had only one end-of-file character and was padded out with nul bytes to the end of the block.

I stumbled into this when one day last week I hurriedly prepared an INCLUDE file with EDLIN — it appears that EDLIN uses only one 1A Hex byte and then pads out the rest of the block with nul (00) bytes. There may be other editors that do the same thing.

Here is the fix that I installed and it seems to work very well (all changes in the area of offset address 72CE-72DE in your code). This fix just makes it loop out on either character (1A or 00):

2318:72B8 8B1ED609	MOV	BX,[09D6]
2318:72BC C606C00100	MOV	Byte Ptr [01C0],00
2318:72C1 E97502	JIP	7539
2318:72C4 C45706	LES	DX,[BX+06]
2318:72C7 FE06C001	INC	Byte Ptr [01C0]
2318:72CB E97402	JMP	7542
2318:72CE 807CFF1A	CMP	Byte Ptr [SI-01],1A
2318:72D2 7409	JZ	72DD
2318:72D4 807CFF00	CMP	Byte Ptr [SI-01],00
2318:72D8 7403	JZ	72DD
2318:72DA E99702	JMP	7574
2318:72DD 4E	DEC	SI
2318:72DE EBEE	JMP	72CE

Many thanks for the original article and I hope the above saves someone else some heartburn.

Best regards,  
Bob Pennington  
1087 Southgate Drive  
Charleston, SC 29407

### First Capitol Computer

Dear HUG:

Not many companies in the micro business have been around as long as First

Capitol Computer. Indeed, not many have deserved to be. In my dealings with FCC over the years, they have not only been extremely helpful and supportive, but have also gone out of their way to make life a little easier. I have worked with no similar company where in trying to work out a particularly interesting problem a technician has given out a private number or has been willing to stay until the problem is either fixed or defined. FCC has not always been the cheapest, but when one considers the service and support they throw in the deal, they have always been a bargain.

In one recent dealing with them, in addition to providing a good price on some Zenith equipment (their specialty), they also donated four computers to our hospital. Another time they called me to thank me for recommending them to another new customer. If only more companies paid attention to the details and the personal touch. This is a company you will enjoy dealing with. Note: First Capitol computer can be reached at (800) 862-8948.

Sincerely,  
Dave Ussell  
Physicist/Coordinator  
Research & Development Engineering  
Children's Hospital and Health Center  
8001 Frost Street  
San Diego, CA 92123

### First Capitol Computer

Dear HUG:

Not many companies in the micro business have been around as long as First Capitol Computer. Indeed, not many have deserved to be. In my dealings with FCC over the years, they have not only

been extremely helpful and supportive, but have also gone out of their way to make life a little easier. I have worked with no similar company where in trying to work out a particularly interesting problem a technician has given out a private number or has been willing to stay until the problem is either fixed or defined. FCC has not always been the cheapest, but when one considers the service and support they throw in the deal, they have always been a bargain.

In one recent dealing with them, in addition to providing a good price on

some Zenith equipment (their specialty), they also donated four computers to our hospital. Another time they called me to thank me for recommending them to another new customer. If only more companies paid attention to the details and the personal touch. This is a company you will enjoy dealing with. Note: First Capitol computer can be reached at (800) 862-8948.

Sincerely,  
Dave Ussell  
Physicist/Coordinator  
Research & Development Engineering  
Children's Hospital and Health Center  
8001 Frost Street  
San Diego, CA 92123

### More HUGCONS

Dear HUG:

Hear it is, the very first day of the new year and it is again time to renew my membership in the Heath Users' Group. I probably am among the first of the membership still remaining a member after all of these years, but I continue to enjoy HUG and REMark and all your efforts in making them a useful addition to the Heath community of computer users. Enclosed please find my check and renewal form for another year's membership.

As long as I have taken a moment to write, please listen to my plea for a renewal of our annual weekend of maximum enjoyment in August at the Chicago Hyatt O'Hare. I would never think of missing one of your fine, enjoyable, and informative HUGCONS and am probably among many who mourn its demise. It left me wondering why I never took the time to thank all of you for your hard work on past conventions.

I did take your advice and attended the convention in Washington D.C. this summer, but left after only a few hours, very disappointed that we had taken the time to travel there. I could find nothing there to compare to the exceptional job you folks have done in the past.

There, I said it and I'm glad!!! Thank you for your wonderful efforts on behalf of the Heath/Zenith Computer users. I know that it is easy to say thank you and tough to contribute time and energy to the effort, but it is difficult to know what one can do to help. If you ever, in re-thinking, decide to "DO-AGAIN" the HUGCON and there is anything I can do to help, (I am a Professional Photographer with very professional equipment, ie: Hasselblad Camera system, etc.), please let me know and I will certainly do my best.

Regards,  
James E. Bovenkerk  
81 S. Broton Road  
Muskegon, MI 49442-9417

Continued on Page 63



# Is This Disk Formatted?

Pat Swayne  
HNG Software Engineer

One of the most frustrating things about MS-DOS is the time it takes to determine that a disk that you are attempting to access is not readable. For example, if you place an unformatted disk in drive A:, and then enter DIR A:, a period of several seconds will pass before MS-DOS prints an error message and returns control to you.

If you have one of the excellent programs from Paul F. Herman, Inc., then you are aware of the newsletter he mails to all of his customers. In the latest edition of that newsletter, he presented a little program for Z-100 series computers that can be used to tell if a disk is formatted or not, much more quickly than MS-DOS (or Z-DOS) can determine it. I liked it so much that I wrote a similar program for PC-compatible computers. He gave me permission to list his program in REMark, so I will list both of them here. I modified

his program so that it is a bit shorter than the original (less typing), and so that the operating syntax is identical to my program.

The programs are both called IS.COM, and to use them, you just type

IS d:

where d: is the designation of the drive to test. The program will print "This disk is formatted.", or "This disk is not formatted.", depending on the condition of the disk. The programs work by trying to access track zero sector 1 on the disk at the BIOS level. (The BIOS is the Basic Input/Output System that MS-DOS uses to communicate with the outside world via disk drives, the keyboard, the screen, and printers.) The BIOS's of Z-100 and PC-compatible computers are entirely different, which is why two different programs are needed.

Normally, when MS-DOS attempts to read a disk, it will make more than one attempt if the first attempt fails. The IS pro-

grams limit the read attempts so that you get a result back faster. In the Z-100 version of the program, a table is altered so that the attempt to read the disk will only be tried once. The PC-compatible BIOS only makes one attempt normally, so the PC version of the program only has to command the BIOS to access the disk, and does not have to modify anything. I found, however, that I had to perform the test twice in the PC version, to make it work with dual capacity drives. For example, if you have a high capacity drive that can read both 1.2 megabyte and 360k disks, an attempt to read the 360k disk when the BIOS tables are set up for a 1.2M disk will fail. On the second attempt to read the disk, the BIOS will have altered the tables to attempt a 360k disk read, so the second attempt will succeed if the disk is good.

Below is the source code for the two versions of IS.COM. These programs are not only useful in themselves, but the code could be incorporated into other programs, such as directory programs, so

To create the PC-compatible version of IS.COM, type in and run this BASIC program:

```
10 PRINT "CREATING IS.COM (PC VERSION)"
20 OPEN "O",1,"IS.COM":L=100
30 FOR I=1 TO 12 :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:SYSTEM
100 DATA 160,92,0,10,192,117,6,180,25,205,987
110 DATA 33,254,192,254,200,138,208,82,180,0,1541
120 DATA 205,19,90,179,2,184,1,4,50,246,980
130 DATA 185,1,0,83,81,82,205,19,90,89,835
140 DATA 91,10,228,116,9,254,203,117,232,186,1446
150 DATA 63,1,235,3,186,91,1,180,9,205,974
160 DATA 33,205,32,84,104,105,115,32,100,105,915
170 DATA 115,107,32,105,115,32,110,111,116,32,875
180 DATA 102,111,114,109,97,116,116,101,100,46,1012
190 DATA 36,84,104,105,115,32,100,105,115,107,903
200 DATA 32,105,115,32,102,111,114,109,97,116,933
210 DATA 116,101,100,46,36,0,0,0,0,0,399
```

To create the Z-100 version, type in and run this BASIC program:

```
10 PRINT "CREATING IS.COM (Z-100 VERSION)"
20 OPEN "O",1,"IS.COM":L=100
30 FOR I=1 TO 18 :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:SYSTEM
100 DATA 160,92,0,10,192,117,6,180,25,205,987
110 DATA 33,254,192,254,200,162,89,1,60,7,1252
120 DATA 118,5,186,100,1,235,56,176,8,154,1039
130 DATA 72,0,64,0,160,89,1,50,228,209,873
140 DATA 224,3,216,38,139,31,38,255,119,16,1079
150 DATA 38,198,71,16,0,6,83,30,7,187,636
160 DATA 89,1,176,4,154,72,0,64,0,91,651
170 DATA 7,38,143,71,16,186,121,1,114,3,700
180 DATA 186,149,1,180,9,205,33,205,32,0,1000
190 DATA 1,0,1,0,0,0,0,0,9,0,11
200 DATA 66,97,100,32,100,114,105,118,101,32,865
210 DATA 115,112,101,99,105,102,105,101,100,46,986
220 DATA 36,84,104,105,115,32,100,105,115,107,903
230 DATA 32,105,115,32,110,111,116,32,102,111,866
240 DATA 114,109,97,116,116,101,100,46,36,84,919
250 DATA 104,105,115,32,100,105,115,107,32,105,920
260 DATA 115,32,102,111,114,109,97,116,116,101,1013
270 DATA 100,46,36,0,0,0,0,0,0,0,182
```



that the program itself could warn you if a disk is unreadable. If you would like a copy of IS without having to type in a listing, get HUG disk 885-3052. It contains a version of IS that combines the code from both versions here, and will run on either PC-compatible machines or Z-100's.

```
;      IS -- IS THIS DISK FORMATTED?  (PC VERSION)
;      THIS PROGRAM TESTS IF A DISK IS FORMATTED.
;      TO USE THIS PROGRAM, ENTER
;
;      IS [d:]
;
;      WHERE d: IS THE DRIVE TO TEST. IF THE DRIVE
;      IS NOT SPECIFIED, THE DEFAULT DRIVE IS USED.
;      INSPIRED BY Z-100 IS.COM BY PAUL F. HERMAN.
;      PC VERSION BY P. SWAYNE, HUG SOFTWARE ENGINEER.
;      COPYRIGHT (C) HEATH/ZENITH USERS GROUP 1988.

CODE    SEGMENT
        ASSUME CS:CODE,DS:CODE,ES:CODE,SS:CODE
        ORG 5CH
FCB     LABEL  BYTE           ;DEFINE DEFAULT FCB
        ORG    100H

START:  MOV     AL,FCB         ;GET SPECIFIED DRIVE
        OR      AL,AL         ;ANY SPECIFIED?
        JNZ     GOTDRV        ;YES
        MOV     AH,19H
        INT     21H           ;ELSE, GET DEFAULT DRIVE
        INC     AL            ;NEGATE NEXT DEC
        GOTDRV: DEC     AL     ;MAKE DRIVE NO. ZERO BASED
        MOV     DL,AL         ;DRIVE TO DL
        PUSH    DX            ;SAVE DRIVE
        MOV     AH,0
        INT     13H           ;RESET DRIVES
        POP     DX
        MOV     BL,2          ;SET RETRY COUNT
        RETRY: MOV     AX,401H ;FUNCTION 4, 1 SECTOR
        XOR     DH,DH         ;HEAD 0
        MOV     CX,1          ;TRACK 0, SECTOR 1
        PUSH    BX
        PUSH    DX
        INT     13H           ;TEST THE DISK
        POP     DX
        POP     BX
        OR      AH,AH         ;TEST RESULT
        JZ      DSKGD         ;DISK IS GOOD
        DEC     BL            ;TRY AGAIN?
        JNZ     RETRY         ;YES
        MOV     DX,OFFSET NFMSG ;ELSE, SAY 'NOT FORMATTED'
        JMP     PMSG
DSKGD:  MOV     DX,OFFSET FMSG
PMSG:   MOV     AH,9
        INT     21H
        INT     20H           ;EXIT

NFMSG   DB      'This disk is not formatted.$'
FMSG    DB      'This disk is formatted.$'

CODE    ENDS
        END      START
```

```
;      IS -- IS THIS DISK FORMATTED?  (Z-100 VERSION)
;      THIS PROGRAM TESTS IF A DISK IS FORMATTED.
;      TO USE THIS PROGRAM, ENTER
;
;      IS [d:]
;
;      WHERE d: IS THE DRIVE TO TEST. IF THE DRIVE
;      IS NOT SPECIFIED, THE DEFAULT DRIVE IS USED.
;      THIS IS A MODIFIED VERSION OF A PROGRAM BY
;      PAUL F. HERMAN. MODIFIED BY P. SWAYNE
;      ORIGINAL PROGRAM COPYRIGHT (C) PAUL F. HERMAN 1988.
;      USED BY PERMISSION.
```

```
BIOS    SEGMENT AT 40H
        ORG     48H
DSK_F   LABEL  FAR           ;DEFIND BIOS DSK_FUNC
BIOS    ENDS

CODE    SEGMENT
        ASSUME CS:CODE,DS:CODE,ES:CODE,SS:CODE
        ORG     5CH
FCB     LABEL  BYTE           ;DEFINE DEFAULT FCB
        ORG     100H

START:  MOV     AL,FCB         ;GET SPECIFIED DRIVE
        OR      AL,AL         ;ANY SPECIFIED?
        JNZ     GOTDRV        ;YES
        MOV     AH,19H
        INT     21H           ;ELSE, GET DEFAULT DRIVE
        INC     AL            ;NEGATE NEXT DEC
        GOTDRV: DEC     AL     ;MAKE DRIVE NO. ZERO BASED
        MOV     DRIVE,AL      ;SAVE DRIVE CODE
        CMP     AL,7          ;TEST DRIVE CODE
        JBE     DRVOK         ;IT S OK
        MOV     DX,OFFSET BADDRV ;ELSE, SAY IT S NOT
        JMP     PMSG
DRVOK:  MOV     AL,8
        CALL    DSK_F         ;GET DRIVE PARAMETER VECTOR TABLE
        MOV     AL,DRIVE      ;GET DRIVE
        XOR     AH,AH         ;AX = DRIVE
        SHL     AX,1
        ADD     BX,AX         ;FIND VECTOR FOR THIS DRIVE
        MOV     BX,ES:[BX]    ;GET START OF DISK INFO BLOCK
        PUSH    ES:16[BX]    ;SAVE ORIGINAL RETRY COUNT
        MOV     BYTE PTR ES:16[BX],0 ;SET NO RETRIES
        PUSH    ES           ;SAVE BLOCK START
        PUSH    BX
        PUSH    DS
        POP     ES           ;PUT ES HERE
        MOV     BX,OFFSET DSKPR ;POINT TO OUR TABLE
        MOV     AL,4          ;VERIFY FUNCTION
        CALL    DSK_F         ;VERIFY SECTOR 1 ON DISK
        POP     BX           ;RESTORE BLOCK START
        POP     ES
        POP     ES:16[BX]    ;REPLACE OLD RETRY COUNT
        MOV     DX,OFFSET NFMSG ;ELSE, SAY 'NOT FORMATTED'
        JC      PMSG
DSKGD:  MOV     DX,OFFSET FMSG
PMSG:   MOV     AH,9
        INT     21H
        INT     20H           ;EXIT

DSKPR   LABEL  NEAR           ;LOCAL DISK PARAMETER TABLE
DRIVE   DB      ?            ;DRIVE CODE
SECTOR  DW      1            ;SECTOR 1
COUNT  DW      1            ;ONLY ONE SECTOR
DTAOFF  DW      ?            ;BUFFER ADDRESS DOESN T MATTER
DTASEG  DW      ?
LEN      DW      9           ;LENGTH OF TABLE

BADDRV  DB      'Bad drive specified.$'
NFMSG    DB      'This disk is not formatted.$'
FMSG     DB      'This disk is formatted.$'

CODE    ENDS
        END      START
```

\*

# POWERING UP

William M. Adney

P.O. Box 531655

Grand Prairie, TX 75053-1655

## Adding More Memory to Your Computer

Although most of today's computer systems come with an adequate amount of memory (i.e., either 640 K or 1 MB) for most applications, that hasn't always been true. Even today, you can still find some computers that have a "standard" (i.e., included) memory of 256 K or so, and it is sometimes a real trick to know how much and what kind of memory you need to add to your system. This article will help you figure out how to do that. In addition, you will learn how to determine whether you can add some newer memory capabilities, such as extended or expanded (sometimes called EMS) memory. We will also look at some myths surrounding the use of memory in a computer, and you will see why adding memory is not quite as simple as plugging in some memory "chips" or adding a memory board. Let's begin by looking at some general information about memory in your computer.

### Remember the Terminology

Virtually all computers have two general types of data storage devices: nonvolatile and volatile. A NONVOLATILE storage device is one that retains (i.e., does not lose) its data when the computer power is turned off. We use a floppy or hard disk as a nonvolatile storage device so that we can keep programs and important data in a form of "permanent" storage. Another very unique storage device, called a ROM (Read Only Memory), is also used to keep special "programs" that are used to start up (i.e., boot) and run your computer. Virtually all DOS-based microcomputers contain a special ROM, called a MONITOR ROM or SYSTEM

ROM, that contains these special programs. And since we need these programs each time we power-on the computer, they are stored in a nonvolatile ROM to save them when the computer power is turned off. Moreover, these programs are so important that they can be READ (hence the term Read Only), but not overwritten or erased.

On the other hand, a VOLATILE storage device DOES LOSE data when computer power is turned off. Most of the memory capacity in your computer consists of volatile memory, usually called Random Access Memory or RAM, that is used to store programs and data when you are using your system. When you use an application program, such as a word processor, it is important to save your data from memory to disk (i.e., nonvolatile storage), because that data in memory will be lost when the power is turned off. The best way to prevent data from being lost is to NEVER power-off your system until you have exited from an application program and returned to the DOS command prompt.

A PC compatible computer may have up to 640 kilobytes of Random Access Memory, or RAM, that is specifically used for the storage of programs and data. But what does 640 kilobytes really mean?

When you discuss memory (or disk) capacity for a computer system, you will normally be talking about measurements in terms of kilobytes (KB or just K) or megabytes (MB or just M). In computer terminology, however, the KILO prefix is defined to mean 1,024, and the MEGA prefix is defined to mean 1,048,576 (1024 x 1024). You can easily demonstrate this

fact to yourself in your own computer system.

Let's say you have 640 K of memory in your system. Run the CHKDSK command, and you will see that it reports "655,360 bytes total memory" in your system. If you multiply the 640 kilobytes times 1024 bytes/kilobyte, you will see that the result is indeed 655,360 as displayed by CHKDSK.

One of the best capabilities of many of the older Heath and Zenith PC compatible computers (e.g., the '151) was that it was extremely easy and inexpensive to add memory to the system up to the maximum of 640 K. All you had to do was buy the appropriate "size" memory chip and plug it into the appropriate places on one of the existing boards. On most other systems, such as the IBM PC or XT, you had to buy a separate add-on "memory board" because there was no space to plug in additional memory chips. And in some cases, you still had to buy the memory chips as a separate item anyway. But what kind of memory do you need?

### RAM, DRAM, and SRAM

Most of today's computers use memory chips identified as "RAM" (Random Access Memory). The specific kind of chip is technically identified as a DYNAMIC RAM, or DRAM, to differentiate it from the STATIC RAM, or SRAM, chip. DRAM chips are less expensive because they tend to "forget" information and must be constantly refreshed to prevent that. For those of you familiar with electronic components, a DRAM stores information by essentially charging up very small capacitors. And like a capacitor, the voltage de-



creases over time, and the voltage must be constantly refreshed. The "Dynamic" in DRAM refers to the constant refreshing required by that type of memory, and you will sometimes see the term "refresh rate" in computer documentation which specifies how often this occurs. Unless some documentation specifically mentions SRAM, you can reasonably assume the term RAM refers to Dynamic RAM.

Static RAM does not require constant refreshing because data is stored in special circuits (called flip-flops) that function like a light switch: either on or off. That is, SRAM holds its data without the refreshing required for DRAM, and that data remains "static" until they are changed or the power is turned off. Because of the technology required to manufacture SRAM devices, they are more expensive than DRAMs.

Both types of RAM are volatile storage devices and lose their data when power to them is turned off. And remember that a ROM is a nonvolatile storage device that does NOT lose its data when the system is powered-off. Now let's take a look at the various types of memory that you can add to your computer system.

### Computer Memory

In general, there are three basic types of computer memory: conventional memory, expanded memory, and extended memory. The idea behind the design and use of each one is different, and we'll look at how and when to choose each one. The most important type of memory that you can add is conventional memory.

### Conventional Memory

Conventional memory is any memory that is used to "fill out" the computer's memory to the maximum of 640 K in a PC compatible system. Conventional memory is also known by a number of other terms: system memory, DOS memory, memory, main memory or RAM. Or it may be an appropriate combination of those, such as "system RAM" to differentiate it from the system ROM.

This 640 K memory limit is sometimes mistakenly called the "DOS limit" even though the limiting factor is based on hardware that really has nothing to do with DOS. Most owners of the Zenith Z-100 MS-DOS computer already know this because the Z-100 can use up to 768 K of system RAM. In the final analysis, however, both limits are actually imposed by hardware, specifically the design of the CPU chip used in the system, and it is useful to understand this concept because other types of memory that you can add are based on the type of CPU chip you are using. Let's see why there is a limit on system memory in the first place.

### System Memory Limits

A PC compatible's 640 kilobyte limit and the Z-100's 768 kilobyte memory lim-

it are based on the hardware limitation in the 8088 CPU (Central Processing Unit). By design, the 8088 cannot address more than one megabyte (1024 kilobytes) of total memory. That total includes all memory that is in the computer: conventional system memory (i.e., RAM), video memory, ROM, and some other things. Without going into a detailed memory map of a PC, suffice it to say that IBM decided to reserve the top 384 K of memory for these functions. The remainder (640 K) was available for conventional system memory or system RAM. That's the reason for the 640K limit in the original PC systems. For the Z-100, only 256 K was reserved for the other functions, leaving 768 K available for system memory.

The 80286 CPU has a different internal design which allows it to address (i.e., use) more system memory. And the 80386 CPU is even more powerful (it uses 32 bits) so that it can address even more system memory. Both newer CPUs can talk to significantly more system memory than the old 8088 because they can use "bigger" addresses. For example, the 80386 CPU can talk to four gigabytes (4,096 megabytes) of memory. By way of comparison, many large mainframes use some multiple of 16 megabytes for main memory (i.e., RAM), and it is not unusual to find 64 MB of memory on a mainframe.

How much memory can be used by a specific application program depends on what program you are using and what type of memory you have available. Whatever operating system and application programs are used, of course, must be capable of addressing the memory in the system.

It is unlikely that any PC compatible DOS version will break the 640 K system memory limit on any system since it also has to run on 8088-based systems. The 80286 and 80386 systems essentially break that barrier because they can address much more than one megabyte of memory. And finally, new operating systems, such as OS/2, are written to take advantage of the increased memory addressing capability of the newer CPUs, and in order to take advantage of that additional memory, application programs must also be specially written to use it.

As new applications are written, they require more and more system memory. Some applications of today's software, using spreadsheets, for example, are so large that it is not at all difficult to exceed the system memory limit of 640 K. That's because these memory-intensive applications retain ALL data in memory, and this is primarily done to keep calculations as fast as possible, particularly with spreadsheets. But even with the 640 K memory limit imposed by the 8088 computer's design, there is still a clever way to add more memory to these and later model computers (e.g., 80286 and 80386).

### Expanded Memory

EXPANDED MEMORY is used to "expand" conventional system memory beyond the normal 640 K barrier, and it can be used on nearly any PC, AT or any other generally compatible system like the Z-200 series or the Z-386. This type of memory is frequently called EMS for Expanded Memory Specification that was developed jointly by Lotus, Intel, and Microsoft (LIM). For that reason, it is sometimes referred to as the LIM/EMS. There is also an Enhanced Expanded Memory Specification (EEMS) that does not seem to have achieved much popularity. But the installation of expanded memory is more than just a special memory board.

Virtually all expanded memory boards include a special software (called a device driver) that must be installed in the CONFIG.SYS file in order to use expanded memory. In most cases, the order of the commands in the CONFIG.SYS file does not matter, but when expanded memory is used, the order is absolutely critical. You MUST install the EMS device driver BEFORE you attempt to install any other device driver that uses it. For example, the CONFIG.SYS file would contain some lines like:

```
DEVICE=EMS.DRV (Right way)
DEVICE=VDISK.SYS /A
```

The EMS.DRV is a made-up example of the device driver name for expanded memory, and I haven't included all of the parameters for the VDISK device driver, except for the /A switch which tells Zenith MS-DOS 3.21 to use expanded memory. A /E switch is used for extended memory. But if you have the lines in the reverse order, such as:

```
DEVICE=VDISK.SYS /A (Wrong way)
DEVICE=EMS.DRV
```

Here, you have attempted to load VDISK in expanded memory before initializing the memory with the EMS driver, and it simply won't work. Most of the documentation for expanded memory boards points out that the EMS driver must be installed before any other driver that uses the memory, but sometimes it is hidden or omitted. In general, you need to be somewhat careful about the order of the DEVICE= commands for device drivers, and the EMS driver is an example of how critical that can be. If you have added a new device driver and something doesn't work, try changing the order of the DEVICE= commands. In the worst case, you may have also discovered an incompatibility between device drivers you use. Remember that device drivers are essentially memory-resident programs, and you can have conflicts among them just like the ones you load on a command line.

You may notice that the subject of expanded memory drivers was also mentioned in the article about the CONFIG.SYS file. The placement of the DEVICE=

command for the EMS driver is so critical that it is again included here so that you will have all essential information about EMS in one article.

When you buy a board to add expanded memory to your computer system, there are some cautions that are discussed later in this article. For now, be sure that the memory board you buy is designed to work with the CPU in your computer, and be sure that the memory board is also capable of working at the clock speed that your computer uses. We'll talk about some easy ways to figure this out later. Memory boards of this kind are easy to find, and they are usually in well-marked boxes that prominently mention the words "expanded" or "EMS". But now let's look at the other type of memory.

### Extended Memory

In contrast to expanded memory that is used to "expand" the 640 K limit, EXTENDED MEMORY can only be added to a computer system that is capable of addressing more than one megabyte of memory. That means you MUST have at least an 80286 CPU (or an 80386) in your computer because of the one megabyte limitation of the 8088-based computer. For example, you can add extended memory to a Zenith Z-200 series computer, or a Z-386, but you cannot add it to older PC compatible systems, like the Z-151, '158, '148, and so on.

You can buy boards to add extended memory to a computer, and they are usually well-marked as "extended" memory so that they won't be confused with expanded memory boards. When you add extended memory to your computer, there is no particular need for loading special software (i.e., a device driver) to use it. Nevertheless, you need to be aware of the uses and limitations of both expanded and extended memory.

### When to Add Expanded Memory

There are few things that must be as disappointing as spending hundreds of dollars for additional memory and then finding out that you cannot really use it. The purpose of this section is to help you determine if you really need expanded or extended memory, and what you can do with it once you get it. But before you even consider getting either one, be sure you have the maximum amount of system RAM (e.g., 640 K) before you consider anything else. Aside from the fact it is usually less expensive to add system memory, some of the expanded and extended memory boards may require that you have the maximum amount of system RAM.

When should you consider getting an expanded memory board? The answer to this question will generally be found in your application software manuals, and

they will mention if that program can use expanded or EMS memory. This is true because having an expanded memory board and its device driver installed is simply not enough for you to effectively use expanded memory — your software MUST also be written to use it, too. For example, most current versions of today's spreadsheet programs can use expanded memory because they are especially written to do so. Older versions of the same program may not be able to use expanded memory, and you may have to purchase an upgrade to your current program in order to use it.

One reason to consider the possibility of expanded memory is when your current software displays an error message similar to "Insufficient memory" or something like that. If your program already supports expanded memory, read the documentation carefully to be sure that additional memory will indeed solve the problem. If the documentation says nothing about expanded memory, then it is almost certain that it will not solve the problem. At that point, you may want to check to see if the current version of the program can use expanded memory by calling or writing to the manufacturer.

In short, be sure you know specifically what program or programs you want to use that can use expanded memory to solve a problem you have. Of course, you can always use expanded memory as a virtual disk with the VDISK device driver that is supplied with current Zenith MS-DOS versions, but that is an expensive way to speed up disk access. If you are considering that kind of application, you are much better off spending the money for a hard disk instead.

### When to Add Extended Memory

What about adding an extended memory card to your system? The answer to this question is even more limited than expanded memory because there is even less software available that can currently use extended memory. The best reason that I can think of to buy extended memory is if you are also considering buying a new operating system (not DOS) that can use it.

For example, the new OS/2 operating system requires at least 1.5 megabytes of memory (most manufacturers recommend a minimum of 2 megabytes). That includes the usual 640 K, plus an extended memory board to bring your total memory up to that. Extended memory is not particularly useful in the DOS world unless you need a large VDISK, but it is essential if you intend to run OS/2 on your system.

To summarize, extended memory begins at one megabyte and can only be used on Z-200 and newer systems. Extended memory is not particularly useful in a DOS system, since there are few prog-

rams that can use it, but it is required for OS/2. On the other hand, expanded memory can be added to nearly any PC or AT compatible system, but a special device driver must be installed before you can use it.

For both types of memory, you must have software that can actually use that specific type of memory. Expanded memory is perhaps the most limited in that regard, because an application program must be specifically written to use it. For example, the current version of Lotus 1-2-3 is one that can use expanded memory. My suggestion is to not buy any type of memory unless you have a specific use for it, and you know what kind of memory you need. You may have spent hundreds of dollars only to find that you can't use it in the way you thought.

Knowing which type of memory you need to solve a problem is only half the battle: you must also know some other information about memory, in general, to be sure that the memory chips or boards will work in your computer system.

### Memory Size and Speed

Memory chips are rated in terms of "size" and access time. Typical size ratings include 64 kilobit and 256 kilobit, although the one megabit chip is now beginning to become more popular. Notice that all ratings are in BITS, not bytes.

You may remember that a byte consists of 8 bits (binary digits), which is usually equivalent to a character that you see on the CRT. So, if you want to add 256 kilobytes of memory to your computer, it will take at least eight 256 kilobit memory chips to do so. In some computers, nine 256 kilobit chips are required for a 256 kilobyte bank of memory because the ninth bit is used for a "parity" bit to ensure memory integrity.

When you buy memory chips for your computer, you will need to get them in groups of eight or nine to add a specific amount of memory to your system — you can count the number of sockets on the memory board to find out the number of memory chips to buy for your system. But you also need to know something else before you invest in some additional memory: What "speed" do you need?

That depends. Dynamic RAM chips are also rated in terms of access time, and these ratings are in nanoseconds (abbreviated ns). Common access times range from a "slow" of 150 ns to a "fast" of 70 ns. Lack of knowledge of the speed of memory chips, and the capability of memory boards to work at a certain speed, has created some problems for a few owners of Zenith computers. For example, the original Zenith Z-151 PC compatible computer was designed to operate an 8088 CPU at a clock speed of 4.77 MHz just like the original IBM PC. Then, the 80286 computers were released. Zen-



ith had the 6 MHz Z-241, and IBM had the 6 MHz AT. Both manufacturers made their machines faster, which resulted in the 8 MHz Z-248 and the 8 MHz IBM AT. In the midst of all this, Zenith also released a dual-speed 8088-based computer (the Z-158) that could run at either 4.77 MHz or 8 MHz.

Many users were quite distressed to find that a memory board that worked fine in one computer would not work in another one or that the memory would not work at a faster clock speed. There were even some reports from bitter Zenith Z-158 computer owners alleging that the '158 was not PC compatible because it could not use IBM PC memory boards. Similar reports were made concerning boards that worked in the IBM AT models, but would not work in the Z-200 models. What happened?

In general, there were two kinds of problems. The first problem was that some third-party manufacturers designed memory boards to operate specifically at a certain clock speed, such as 4.77 MHz. Because of that design limitation, some of the boards would just not work at the Z-158's 8 MHz clock speed, and a few would not even work at the IBM AT's 6 MHz clock speed. This was a specific case where Zenith's better engineering, in terms of faster computers with higher clock speeds, was ahead of most other manufacturers. But there was one other problem.

There was also at least one memory board that would not work at the Z-241's 6 MHz clock speed, although it worked just fine in the 6 MHz IBM AT. And some memory boards that worked in the Z-241, and the 6 and 8 MHz IBM ATs, would not work in the 8 MHz Z-248. In some cases, this problem could be "fixed" by simply changing the existing slow DRAMs to faster ones. That is, change the 150 ns DRAMs to 120 or 100 ns DRAMs. In other cases, that did not work because Zenith again used more current engineering designs than other manufacturers. That involved something called a "Wait State".

#### Wait States and Clock Speed

One reason to introduce a wait state is to help keep the cost of memory down. Simply defined, a WAIT STATE is an extra processing cycle (that is totally useless) used to slow down a fast microprocessor (e.g., an 80286 CPU) so that it can use "slower" memory chips. For example, let's say a 4.77 MHz 8088 CPU required zero wait states for 150 ns DRAM chips. In order to use those same 150 ns DRAMs at 8 MHz in a fast 80286 CPU system, we might have to add one (or perhaps two) wait states to slow the 80286 down to the point that it could use these slower DRAMs. The other alternative is to use faster DRAM, such as 120 ns or 100 ns, but that is more expensive, and some manufacturers decided to cut costs this

way.

When Zenith introduced the Z-200 series of 80286 computers, some ads mentioned that these computers had "zero wait states." Unfortunately, Zenith failed to mention at the time that many other similar computers, such as IBM AT, had "one wait state," although most ads did mention that Zenith computers were fast. Because many computers were designed with one or more wait states — and many memory boards were designed to operate at that one wait state specification at a specific clock speed — some of these memory boards would not operate at the same clock speed with a zero wait state for memory on Zenith computers.

Although it may not seem like it is important to a new computer user, it really is important to know something about the technical specifications about your system, especially if you want to buy a memory board or a new computer. For starters, you need to know what kind of CPU (e.g., 8088, 80286 or 80386) and the clock speed of your computer. And you need to know how many, if any, wait states your computer has. You can generally find this kind of information in your owner's manual or other system documentation.

Despite the emphasis on clock speed in our speed-oriented computer age, that is not the whole story. Given a computer system with the same clock speed, it is relatively easy to show that a computer with zero wait states is effectively 25% to 33% faster than a computer with one wait state. For example, a national magazine mentioned that the 16 MHz Zenith Z-386 computer (zero wait states) was at least as fast as one 20 MHz Compaq 80386 computer (one wait state). The addition of a wait state slows down the actual processing power of a computer because its EFFECTIVE processing speed is really a combination of clock speed and the speed of memory access. It does no good to have a fast CPU in a computer if its memory cannot also operate at that same speed.

#### How to Buy DRAMs or a Memory Board

The whole point of this discussion is to help you choose DRAMs or a memory board that will work with your Zenith computer. DRAMs rated at 100 ns or 120 ns will work fine in an 8 MHz Zenith system. When you buy DRAMs, the speed of the unit is generally marked on each one, but you have to know how to read it. For example, you might find a DRAM marked as "4164-15" or "4164-12". The "4164" means that it is a 64 kilobit chip, and the "dash" number is its rated speed. Here, the "-15" means 150 ns, and the "-12" means 120 ns. Similarly, a 256 kilobit chip with a 150 ns rating may be marked as "41256-15".

When choosing a memory board of any kind, keep in mind that it should be chosen first based on the clock speed of your computer, and then find out if it is

rated for zero wait states. For the most part, a board rated at a higher clock speed than your computer has should also work fine. Because there are so many different brands and types of memory boards, I will not try to suggest specific brands, but I do recommend that you take a look at the various ads in REMark if you want to add memory to your system. Vendors who advertise in REMark understand the requirements for Zenith computer systems, and they will also be able to provide you with advice and technical support if you have any problems.

As you can see, there are lots of things you need to know about buying memory for your computer system. Unfortunately, some of this information is technical and requires that you know something about the hardware characteristics of your computer, but knowing these details, such as clock speed, will help you become a more knowledgeable computer user. In addition, I have also tried to present some background information that explains why things are the way they are. I usually find it is easier to remember something if I know WHY something was done that way. The 640 K memory limit for the PC is one example of that. Perhaps you will find this information useful in remembering why there are limitations and what kind of memory is used for which purpose, too.

#### Next Time

Choosing an application program, such as a word processor, is a difficult task at best. There are so many programs to choose from that it is nearly impossible for a new user to select the "best" one. Still, there are a couple of ideas that can help you choose application software that is useful both today and tomorrow. And despite the fact that everyone has specific opinions on which program is best for a given purpose, you will see how to develop an objective way to help you select any type of application program that will serve your needs. This selection technique can be used to select a word processor, spreadsheet, database or integrated software packages.

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

HS-386 Computer	
(Kit version of Z-386)	\$3349.00
Memory Cards (EMS/extended memory)	
1 MB (Z-505)	799.00
4 MB (Z-515)	2999.00
Heath Company	
Hilltop Road	
St. Joseph, MI 49085	
(800) 253-7057	
(Heath Catalog orders only)	



# SOFTENING THE HARD DISK

Before buying my first hard disk, I marveled at the patience of all the users already battling with the complexities of the MS-DOS hard disk interface. Some of them wrote some very helpful articles, programs even, detailing their miseries, though in a tone which barely concealed the fact that they were not overly displeased at having these difficulties, and having to find solutions, and thus being able to hand down help to ordinary mortals. "Put yourself in the hands of the DOSo-disko-sado-macho and you have nothing to fear."

The prospects of replacing floppies with a hard disk, for those who have not yet taken the plunge, are not just bright, they're resplendent. All your files instantly available; only one copy of each, instead of multiple ones on different floppies; high speed loading and saving.

A hard disk is not, though, all pleasure. First, there is the business of changing from one directory to another — CD \WP\LETTERS\PERSL is an example. Having gone to the trouble and expense of buying and learning a fast and capable spreadsheet or word processor, you have to slow almost to a halt when you want to do something as simple as access a different directory — a maneuver you cannot avoid, short of having only one directory.

And, of course, there are backups. They are already necessary when you use floppies, but hard disks concentrate the misery into one large lump, instead of leaving it piecemeal, and bearable.

There is little doubt that you do chores best when you can do them quickly and easily. When they are difficult to do, you don't do them, you put them off, and you may do them badly, for you are more likely to make mistakes when you do them. We need easier access to subdirectories. And easy ways of doing the important things, like backups, too. And if there is an easier way of getting information on the state of the whole system, that would not be bad either, something like global DIR and CHKDSK commands for the whole disk.

MS-DOS (or PC-DOS, for this article concerns PC clones, and may not apply to the Z-100) offers a hierarchical, multi-level, directory system, which allows you to divide a physical disk, be it hard or floppy, into separate compartments, only one of which is visible at a time. It is an extension of the default disk idea, where only one of the different disks is available by default, without specifying a diskname, but this time the unit is inside the disk, at the level of the subdirectories. One of the

## PART I

**JOHN A. NEGUS**

**07150 VALLON PON D'ARC  
FRANCE**

sub-units into which your disk is divided is always the current one, accessible by default, without adding a naming parameter to your access command.

DOS in a sense localizes the user: it assumes, unless told otherwise, that the user is 'in' the current, default, directory on the default disk.

When you first access your disk after booting, or loading it in the case of a floppy, you are in the root directory and it is the root directory that is the default. DOS pretends, until you change to another subdirectory, that the physical disk contains no more than that one subdirectory. If you do a DIR C:, for example, DOS lists the names of the files present in only the current subdirectory, plus the name of any directory or subdirectory linked directly to it. All the other subdirectories are temporarily invisible, until you name them, as the object you wish to access, or change to a different default directory.

If you create, then change to, a subsidiary subdirectory, again DOS keeps its blinkers on, pretending it is the only directory on the disk in question. You could have several disks with several subdirectories each, and one of the subdirectories is always the default for its disk. So the default subdirectory on each disk temporarily adopts the name of the disk: DIR C: will display the files of the default subdirectory on C:. Only if the root directory is the default directory will its files be displayed. If you want to see the files in the root directory when another subdirectory is the default, you have to name it — DIR C:\, with the backslash, the character DOS uses to separate subdirectory names.

In the string:

`\WP\LETTERS\PERSL`

WP, LETTERS and PERSL are each subdirectory names in the same directory, each one one level lower than the preceding one. It is a convenient way of dividing your large disk up into smaller, ordered, packets. Most particularly, it limits the number of file names likely to appear on the screen at one time. But it also restricts the ease of access.

All this order though is only apparent. What you see is true logically, because DOS has presented it that way. Physically, on the disk, the order is not present, the files are not stored in separate, neat subdirectory cupboards. The reason lies in the way the disk space allocation routines were designed. If you add new material to a data file from time to time, DOS, unlike some operating systems, has no choice but to allocate the next bit of disk space available at that moment, starting from the low-numbered sectors. If you have several files that grow like this, the result is best not seen: it is a spaghetti jumble, bits of the files intermingled on the disk haphazardly, irrespective of which subdirectory they are in.

At formatting time, DOS divided the disk surface up into three parts, one for bulk storage and two for record-keeping and control. The latter are first the Directory and second the FAT, or File Allocation Table, which are both located at the start of the disk surface, occupying a few tracks on the outside of the disk. All the remaining tracks are available for bulk storage.

Figure 1 is an illustration, schematic, of a small part of the bulk storage section of an imaginary disk surface which is occupied by several files. The numbers represent the clusters that DOS divides the disk surface up into, logical units that almost, but not quite, correspond to the physical units the formatting process has created; DOS will have written the equivalent numbers on the disk during formatting. The letters are labels showing which file each of the fragments belong to, but they do not exist in reality, they are for illustration purposes only, because, in practice, the file fragments are anonymous. The fragments could be either program code or data from your address list, there is nothing in the bulk storage section that enables DOS to identify a fragment. DOS reassembles your file solely by means of a string of cluster numbers.

The number with no letter next to it is an unoccupied cluster, the result of deleting the file that formerly occupied it. It forms a hole in the disk allocation, which will be filled with the next thing that you add to the disk, be it a new, or an addition to an existing file. MSDOS fills the disk from the bottom up, on demand, starting with the lowest cluster numbers, only filling the high numbered clusters when there is no vacant cluster below.

One slight weakness in DOS, that you may have noticed from time to time, is the occasional incorrect handling of these



'holes' in allocation, the result of deleting a file while leaving its 'lower' and 'higher' neighbors in place; this leaves a hole in the middle of an occupied block that sometimes causes problems when further material is added later, problems easily solved by CHKDSK.

Even the distribution of Figure 1, complicated though it is, is not the whole story, because it does not show the order inside each file: the third cluster occupied by file E may contain material from the start of the file, while the first and second clusters — the first and second 'E's in the diagram — may contain the end of the file; all depends on which cluster was vacant at the time of writing. It ignores also the fact that the 'empty' cluster will not be empty; if you delete a file, DOS alters only the control parts of the disk, and does not touch the bulk storage part, so the file content is still where it was before deletion. Physically full, but logically empty.

DOS is able to reconstruct order out of this apparent chaos with the help of the file string in the FAT. DOS keeps count of the whereabouts of bits of files on the disk surface by keeping a record of the number of each of the clusters it occupies, this in the File Allocation Table, which has a slot for each of the clusters on the disk. As DOS writes the file, it is writing two things — the content of the file to the bulk storage part and the location information to the FAT. Into any newly occupied FAT slot it places information enabling it to find and reassemble the pieces of the file later. At the end of the write, it also updates the Directory, chiefly with the new file length.

The Directory contains the general details, like the file's name, length and date, plus the position in the FAT of its first cluster. The FAT contains one sort of information only, the cluster mapping.

On an empty disk, the FAT is empty. DOS writes the first files you copy to a disk into the first, empty, clusters in an orderly fashion, one after another, and fills a single entry in the File Allocation Table, the FAT, for each cluster it writes. There being a one-to-one correspondence between slots in the FAT and clusters in the bulk storage section, DOS fills the FAT slot corresponding to the cluster in question. This entry, actually a 12-bit 'character', plays several roles: the position of the entry written corresponds to the cluster number; the content, however, of the entry written corresponds to something else, the number of the next cluster occupied by the file.

Figure 2 illustrates the principle, not the reality, of the first few entries of an imaginary FAT, using symbols that are simpler and smaller than the real ones DOS uses. The dashes represent empty slots, the alphanumeric symbols occupied slots. Here, as in reality, there are

**Figure 1**  
**Part of the Disk Bulk Storage Surface**

```
40 e 41 m 42 l 43 e 44 z 45 p 46 p 47 p 48 p 49 p
50 p 51 e 52 e 53 54 x 55 m 56 f 57 f 58 q 59 l
```

no numbers showing us where we are, DOS keeps count of the positions by counting, not by placing and later reading numbers on the surface; similarly, when you read the diagram, bear in mind that the array is accessed by counting from the first element, zero. Each succeeding line contains ten slots, so the lines are implicitly numbered units, tens, twenties and so on. (We ignore the difference between decimal and hexadecimal).

**Figure 2**  
**The File Allocation Table**  
**(Principle Only)**

```
01 02 03 04 05 Z 07 08 09 Z
-- -- -- -- --
-- -- 35 32 36 34 Z -- -- --
```

Notice that the 01 occupying the first slot is not the same as the number of the slot, which is 00. The symbols DOS writes in the FAT have two meanings, one implicit, one explicit: each number written points to the next slot in the current file string, occupied by the same file. So the presence of 1 in the first slot, slot number zero, tells DOS several things:

- Its presence, in place of an 'empty' sign, indicates that the slot is occupied, and that the corresponding material is in file storage cluster number zero;
- Its value, 1, indicates the number of the next FAT slot in the file string.

The Z represents the special characters DOS uses to signify the end of a file string. So Figure 2 represents a very simple, clean, situation, with two integral, unfragmented, files followed by empty space followed by a third integral file string whose clusters were written in abnormal order.

For any one file, therefore, the entries in the Allocation Table are strung together, each pointing to the next one, the final one of the file having an entry which defines it as the last of a file. So a file is locatable by DOS primarily by its Directory entry, which gives, among many other things, the first of the string's FAT slots, then by the string of FAT entries, finally by the end-of-file FAT entry.

At any one moment, the FAT contains a totally anonymous batch of numbers; they need translation if they are to be of any use, and above all they are all useless without an element that resides elsewhere in the Directory.

That element is the number of the first of the FAT slots occupied by the file, the first link of each chain. If we pick this out for the three files mentioned above,

we would find 1 for the first file and 6 for the second file. The latter's first cluster and FAT slot, for instance, is number six; the one containing the value 7 in the diagram. Perhaps, if your head is not swimming already, you can work out the corresponding number in the third file's Directory entry. (It will probably help to pencil in the column numbers).

The FAT is effectively a sort of map of the disk surface, a small scale representation not of roads, but of file strings, linking the different locations occupied by each of the files. And it is the only means of identifying the 'owners' of the different clusters on the disk; these are anonymous, possessing no identifier of their own.

If you place pins in a map showing the whereabouts of all the members of several families, then link the pins of each family by a different colored thread, you have done what DOS does when it writes a FAT. Except that the thread DOS works with is implicit in the numbers it places in the FAT, it is not visible as such. And the 'map' DOS is using is a simple numerical array.

Which is fine, while it works. But it is not a good method of keeping valuables; if the string breaks, you probably lose the valuables. Nobody keeps valuables on a string unless they have a very good reason. But one good reason is access. An example is diamond necklaces — valuables on a string — but they are placed where they are for a reason, easy access by the eye. Files, too, are kept on a string for a reason — but this time it's speed of access by DOS. But the weakness is the same, if anything happens to a file's string, that file is lost. The clusters, like the diamonds scattered on the floor, are still around, but it needs a lot of work to reassemble the string.

The day your disk is damaged and you have no backup, you are going to have to try and reconstitute your files, either by simply tracing the contents on the main disk surface or by first tracing the file location strings in the FAT and then collecting the pieces they refer to on the main surface. Arduous. Anything we can do to keep the spaghetti jumble within limits is a good idea.

As is often the case, you can take measures either to prevent or to cure. The choice is yours. This in addition, it goes without saying, to normal backups; those enter into another category — reconstruction. If your hard disk goes down, reconstituting it by copying, or restoring, all the files back on to it will effectively rebuild it

like new, with no jumble, each file string being contiguous, each family clustered together. On the disk surface the clusters occupied by the file are all together, and in the FAT the corresponding entries for each file are also grouped integrally. Our aim though is to do everything to make recovery from accidents easier than that, to avoid the need for total reconstitution, and, incidentally, to maximize the chances of having a complete and recent backup, minimizing the amount of data lost because of entry subsequent to the last backup.

Cure is defragmentation, using programs which rewrite whole files on to empty parts of the disk, grouping all the parts together. A chore, but worthwhile if for nothing else than speed of access. Integral files are quicker to load than unintegral ones. As a by-product, they are also easier to reconstitute when you run into trouble, and have no backup copy.

Prevention, total prevention, is not possible, but partial measures are. The chief means is to divide the disk into hardware partitions at the start. Heath/Zenith DOS 2.11 led the way, now all versions of DOS 3.2, and later, permit it. Instead of having hard disk C:, you can have disks C:, D:, E: and F:, all on the same hard disk. Four smaller spaghetti jumbles perhaps, but not just that. For you may no longer need a defragmentation program. It is enough to move all the altered files of one partition to another partition and back again to reduce the jumble to zero. You need of course enough empty space on the other partitions; it is no use if you have only 200k free on your 30M disk.

There are other advantages, too. If you do not partition the disk, there is only one Directory and it gets a lot of use. With more than one partition, the main root directory, C: in most cases, gets less use, and the heads of the drive are that much less likely to be over it when the incident on the mains takes place and the head crashes on to the surface, breaking your precious string of valuables.

So partitions were a must for my new hard disk. But though they reduced it, they still did not solve the problem of changing from one directory to another quickly and easily, in a way that does not break the train of thought.

Happily, someone at Microsoft dislikes typing as much as I do, and dreamed up the SUBST idea. The SUBST program, one of the auxiliary ones of DOS 3.1 and up, substitutes an ordinary drive name, like P:, for a subdirectory name, like \wp\letters\persl. Not only do you never have to type the long name, you don't even have to type the CD, change directory, command. If you know how to get along with floppies, you have almost nothing new to learn. You type P:, that's all. Bliss.

Putting all this into operation is not

difficult. The machines being computers, however, there are several stages, usually one or more to set up the new structures, others for using them.

Hard disk installation should be covered adequately in the manuals of the disk or DOS supplier, but in practice there is sometimes a gap covered by neither, especially when changing to a later version of DOS or buying add-on accessories. What is not always very clear, in the case of a hard disk, is that there are always two different steps to initial setup and, if you want to partition it, there is a third. In order of execution, they are:

- Low-level formatting, writing tracks on the disk surface.
- Partitioning the surface into two or more 'sub-disks', optional.
- Formatting for the operating system, dividing the disk surface into control and bulk storage parts and dividing the tracks up into the units the OS recognizes.

The first of these three is often done by the manufacturer; if he leaves it to you, he may supply a specific program on disk or he may ask you to activate the program he has provided, in ROM, by a 'goto' you set in motion from DEBUG. The other two are often left to the user. The details of each operation should figure in the manuals you receive with the hardware; there are too many variants to cover the subject in detail here. What we are going to do is cover the operations subsequent to the above, after initial formatting, partitioning and DOS-formatting of the hard disk.

Sufficient to mention that you have to carry out the third step, formatting, on each of the partitions before being able to use it, and you may have to reboot before being able to do it — after partitioning and before formatting that is — in order for the OS to recognize the partitions. DOS may need access to code that it can only load from CONFIG.SYS, during the boot-up process; which requires you to place the requisite line in CONFIG.SYS and then reboot.

It is when you DOS-format the partition that DOS transfers its system tracks — boot sector, Directory and FAT. When you DOS-format the primary partition, you will probably also, if the hard disk is bootable, place there a copy of DOS itself, by adding the suffix /S, as in FORMAT C:/S for instance.

It is a good idea to give a little thought to the sizes of the partitions, because some partition sizes are unavoidably wasteful of space, not because of their size, but because of the size of the pigeon-holes they divide the surface into. DOS does not give each file a space precisely the same as its length in bytes, instead it fits them into its fixed-size clusters. This means that files almost always include a bit of unused disk space in their final cluster, a 30 byte file, for instance, al-

ways occupying one whole cluster.

If the cluster is 2k in length, the waste is less than with 4k clusters; in order to keep cluster size down to 2k, as you will already know if you follow William Adney's articles, you should use a partition size of either no more than 4 or, curiously, greater than 16 Megabytes. (See his July 1987 and January 1988 articles for more detail). This suggests dividing a 20 Megabyte disk into three 4-Megabyte and one 8-Megabyte partitions, for instance. The public domain program, SD, mentioned below, reports file sizes correctly. It will show different copies of the same 10-byte-long file as occupying different amounts of space according to where they are: 2k when the cluster size is 2k, 4k on a different hard disk whose cluster size is 4k, 1k on a 360k floppy, whose cluster size is 1k, 8k on a 20 Megabyte partition.

Once the partitions exist, initial setup is over, but there is another setup step to take, this time every time the computer boots — setting up the logical mechanism which names the partitions on the hard disk. DOS cannot do this unaided. It may seem ridiculous, for the partitions are there, on the surface, but that is the way DOS was designed. (It was modified originally, in fact, not to support several DOS partitions, but to be able to divide the drive into two partitions, one for DOS and one for another OS.) The program that does the naming of the partitions is sometimes an ordinary program and sometimes a device driver. A device driver is a unit that modifies DOS, extends DOS, so it has to be loaded early on in the boot sequence, by means of a line in CONFIG.SYS similar to the following, which is taken from Epson's version of DOS 3.2:

```
device = hdm.sys
```

In the case of H/Z DOS 2.11, you name the partitions with the help of the program ASSIGN, with entries in AUTO-EXEC.BAT of the form:

```
assign 0:2 d:
assign 0:3 e:
assign 0:4 f:
```

The 0 is the first hard disk; the numeral 2-4 the serial number of the partition; the letter, the drive name. (Of course, using DOS 2.11 will deny you the advantages that SUBST brings; as a general rule these improvements added to DOS will refuse to work with the wrong version of DOS.) Epson's device driver assigns all the extra partitions automatically, in addition to C:, requiring only the one command line, while H/Z's requires a line for each extra partition. The result in each case is the same, four 'disks', C:, D:, E: and F:, residing on a hard disk which originally was C:. The latest versions of DOS use the name ASGNPART in place of ASSIGN, which is the name of a new program with a different purpose.

Once partitioning and DOS formatting of each partition is over, the next step



## Prime H/Z Enhancements!

### Clock Uses no Slot

**FBE SmartWatch:** On-line date/time. Installs under BIOS/Monitor ROM. Ten year battery. Software included. Works with all Heath/Zenith MSDOS computers. For PC's \$35; Z-100 \$36.50. Module \$27.50

### H/Z-148 Expansions

**ZEX-148:** Adds 1-1/2 card slots. \$79.95. ZEX-148 + SmartWatch \$109.95  
**ZP-148:** PAL chip expands existing 640K memory to 704K. \$19.95

### H/Z-150 Stuff (Not for '157, '158 or '159)

**VCE-150:** Eliminate video card. Install EGA/VGA card. All plug in. Includes VEM-150, RM-150. Requires SRAM chip. VCE-150 \$39.95, SRAM Chip \$15  
**VEM-150:** Card combines existing two BIOS ROM's into one socket. \$34.95  
**RM-150:** Decoder PROM used in removing video card. With detailed instructions. \$9.95

**ZP640 PLUS:** Expand to 640K/704K by adding 2 banks of 256K RAM chips (not included). ZP640 PLUS \$19.95 (first one); \$9.50 thereafter.

**LIM 150:** 640K RAM plus 512K of simulated Lotus/Intel/Microsoft EMS v3.2 expanded memory. Installs on H/Z-150/160 memory card. No soldering. Requires forty-five 256K RAM chips (not included). LIM150 \$39.95

**Mega RAM-150:** Get 640K/704K main memory plus 512K RAM disk on H/Z-150/160 memory card. No soldering. Without RAM chips \$39.95

**COM3:** Change existing COM2 port address. Internal MODEM at COM2. Don't lose serial port. COM3 \$29.95

### Maximize Your Z-100

**ZMF100A:** Put 256K RAM chips on "old" motherboard (p/n 181-4917 or less). Expand to 768K. No soldering. Without RAM chips. \$65.00

**ZRAM-205:** Put 256K RAM chips on Z-205 board. Get 256K memory plus 768K RAM disk. Contact us for data sheet before ordering. Without RAM chips. \$49.00

### Z-171 Memory Expansion

**MegaRAM-171:** Put 256K RAM chips on memory card. Get 640K memory plus 384K RAM disk. \$59.95

### H/Z-89 Corner

**H89PIP:** Two port parallel printer interface card. With software.

H89PIP \$50.00; Cable \$24.00

**SPOOLDISK 89 and SLOT 4:** Cards still available. Contact us for information.

Order by mail, phone or see a Heath/Zenith Dealer. UPS/APO/FPO shipping included. VISA or MC. WA residents add 8.1% tax. Hours: M-F 9-5 PST. We return all calls left on answering machine!

# FBE

**FBE Research Co., Inc.**  
P.O. Box 68234, Seattle, WA 98168  
**206-246-9815**

Reader Service #104

is to create the subdirectories, using the MD, make directory, command. This is another set-up operation, done once only, each time you format the disk. Before embarking on it, though, it is worth giving some thought to how you will be working and how to optimize your personal organization of the hard disk.

If you have ever used subdirectories on floppy disks, you will know how slow they are; each hierarchical level slows down file access. There seemed no point buying a hard disk and asking it to go through these contortions, which would nullify part of the expensive/speed advantage. So it seemed better to avoid too many levels of subdirectories, and also long PATH strings, those that allow DOS, instead of having just one default disk, to grind through a series of them that you specify each bootup time, searching for its programs. PATH is an offering that opens a chink in DOS's blinkers, enabling access to program files, but not to data files, without each time naming the place where they reside. If you upgrade to DOS 3.3, you will find an equivalent that works with data files. Slowly, very slowly, DOS is losing some of its early rigidity, inherited from CP/M, though it still requires an unreasonable amount of expertise on the part of its weary user.

Avoiding long PATHs meant treating the hard disk subdirectories in exactly the way one handles floppies, not dumping all programs into one huge C:\BIN, but grouping them into different application disks; one for the spreadsheet, one for the word processor, and so on. The price of this speedup is that you may need to specify the drive along with both program and data file names, rather than make the data drive the default one and demand that DOS go through the labor of finding the PATH to the program files — an acceptable price in view of not having to type 'CD'.

To me, the hierarchical capabilities of DOS are of no importance, in fact, they are a hindrance. What interested me was to get the use of all the remaining disk drive letters through to z:, the easiest possible access to the different compartments on the disk, in fact. In practice, I was forced to use the hierarchical capabilities because the only directories you can create are subdirectories of the root directory, but once the drive letter substitutions are made, there is no hierarchy visible — disks g: to z: are just as available as a: and c:. Practically invisible is the reality that there are four root directories, C:\, D:\, E:\ and F:\, and twenty 'subdirectories', G: to Z:. As the root directories are always also the default directories, you never, after initial substitution, even have to type the backslash, C: becomes equal to C:\. One more economy.

The main choice, as in all programming, was to find the right names. To

keep things as simple as possible, and for reasons that will appear later, the subdirectories received single-letter names, corresponding to the drive letter they would eventually 'own'. My assembler floppies, for instance, were fixed in my memory as 'asm' so my assembler subdirectory became \z, sufficiently alliterative to be easy to remember. It later receives, through the services of the program SUBST, the drive letter z:. Taking the time to baptize your subdirectories with 'names' that you can remember is the way to render use almost painless, to flatten that learning curve.

Subdirectory creation, of \p on disk d: in this case, was a series of commands of the sort:

```
D>md p
```

where there was a line for each letter from g to z. To avoid mistakes, I created a batch file, see program Listing 1, where these lines were punctuated by changes of default disk (the D> DOS prompt in the example) at the right moment to ensure that the subdirectory was created on the right partition. This is, of course, a once-for-all process, or at least once-till-the-next-disk-format process, so you may prefer not to create a batch file. It left me with the subdirectories \g to \z, distributed suitably, but arbitrarily, over the partitions c: to f:.

Program Listing 1: Makdir.bat  
Subdirectory Creation

```
c:
md g
md k
md t
md x
d:
md m
md p
md q
md r
md v
md w
md y
e:
md h
md n
md o
md z
f:
md i
md j
md l
md s
md u
c:
```

I placed only infrequently used subdirectories, like archive files, on the c: sub-disk, to lighten the load on the main root directory tracks, making no attempt to keep things alphabetical.

Being out of space, we pause, having finished dealing with the operations necessary once only each time we format the disk. Next time, we will continue with the operations necessary each time we boot, looking too at ways of configuring the disk to make it easier to use.



# ENABLE - A Tutorial

## Part 15

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

# MENUS

ENABLE has two features that make it possible to develop applications for non-computer users. In part 14 of this series, the macro capability was discussed. A companion to the macros is the menu capability. User developed menus can point non-computer users in the right direction and give them the same "expert" capability as a long time user.

I have developed several applications for non-users, including one for my younger son's school. This application fills in many different reports, both State of Ohio and reports required by the Archdiocese of Cincinnati. The school can track funds and other statistics with the press of a key on a menu. The secretary that runs the computer is not computer literate and, with a limited amount of training, can get through the program.

Another application is the payroll program for our office. This program is also completely menu driven. This program runs three data bases and outputs over 12 reports along with the checks. The reports are outputted with one keystroke that runs a macro that is nearly 80 lines long.

These two examples give you an idea of the capability of the ENABLE menu and macro capability. The development of the menus is a short process that can be completed in less than ten minutes per screen for simple applications. The menu development process is within the word processor so that all of the attributes are available to enhance the final product.

Menu generation, like macros development, is covered in the gray ENABLE System Overview manual. Like basic macro development, menus are developed from the Master Control Modules or MCM on the ENABLE main menu. You can establish the size of the screen and control the "look and feel" of the menu to have a standard menu for all applications. You can develop "Help" screens for the non-computer users to assist them while running the menus. Standard colors can be used. I have found that it is possible to "paint" and make "new" colors.

Before creating the menus, you should have the basic application complete. This includes all reports, macros, data bases, and other related requirements. These should work without the menus. Actually, before you get this far you should have developed a basic flow for the entire process. This flow chart will help you visualize the process. This should help you see problem areas where you missed a requirement in order for the entire application to run correctly. This step SHOULD be the first step of any programming effort. We all say that this is important but how many of us actually do it. Once this is completed the development can start.

ENABLE's menus can be named in one of four ways. The method of naming depends on how you plan to use it in the application. If you plan to use the menu in any of the modules, any single character will do for the name. ENABLE will put an extension of ".MNU" on the completed menu. You are limited to the same number of names as macros. Shifted keys will not work to increase the number of menus available.

The next method of naming menus is for one that will be run from only ONE module. Again, you can name these menus with any one key but you must include the extension, which depends on the module. Menus named in this fashion can have any one key as the name. To develop a default menu for any one module the name must have an extension of ".WP" for word processing, ".SS" for the spreadsheet, ".DB" for the data base, ".TP" for the telecommunication, or ".MC" for the Master Control Module.

The third method of naming a menu is for one that will be used as the default menu for a module. This menu will be called first in a module. To name these menus you must name them "WP" for word processing, "DB" for data base, "SS" for spreadsheet, "TP" for telecommunication, or "MCM" for the Master Control Module. ENABLE will put the extension of ".MNU" with these menus.

The last method of naming a menu is for a menu that will be called from a macro. Menus that are called in this manner can have names up to eight characters long. This is the same manner that macros can be named.

To start the development of a menu, you must be at the Main menu screen. There, select (M)CM and (T)ools. Here, select (C)reate. ENABLE will now display a screen to give you help with the naming convention. This screen tells you that by using "Shift/F10" or "Ctrl/F10" (Shift/F0 F10 on the Z-100) it requires certain names and extensions. After you have named the menu, ENABLE will display another screen with more information. This screen will provide information as to how to develop the menus. This includes how to setup the fields and how to "mark" them. Pressing <RETURN> will display one more screen that tells you to establish the size of the menu screen first.

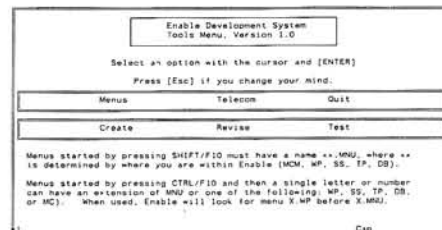


Figure 1  
Menu Creation Screen One

For this first menu, press (S)hrink at the first screen. This will permit you to reduce the size of the menu to something less than a complete screen. This is the same set of commands that are used to control the size of the windows, which you are actually doing. For a menu that will overlay the entire ENABLE screen, just press <RETURN> and you can start creating the menu. To shrink the size of the screen, use the arrow keys. As you press the keys, the double lines will move in from the sides, indicating the size of the window. Note in this example that the menu screen has been reduced in both

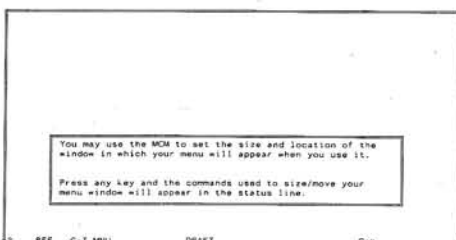


#### Menu Development System

1. Use the MCM commands to set size and location of the window in which the menu will be displayed.
2. Enter (revise) your menu options as if you were revising a Word Processing document. Do not use rulers.
3. Position the cursor JUST BEFORE the first character of the top option within the menu. Press [Shift] and [F9] and you will be prompted to define the characteristics of the option (or menu field).
4. When you use the menu, "marked" areas will appear in reverse video when the cursor is on a menu option. Use the " " symbol to mark areas to the right of where you pressed Shift/F9.
5. Repeat the process for each option in your menu. If you wish to delete or revise the characteristics of a menu option or field, press [F10] and select FORM DESIGN OPTIONS for instructions.
6. When you're finished, press [F10] and select FORM DESIGN OPTIONS and then SAVE. Close the window to return to the Tools Menu. Press any key to continue or Esc.

**Figure 2**  
**Menu Creation Screen Two**

horizontal and vertical size. This menu will appear in the middle of the screen near the bottom. The final location and size of the menu window will be how it is positioned on the screen. After pressing {ESC}, the new menu window will be correctly positioned and sized and ready for the rest of the procedure.



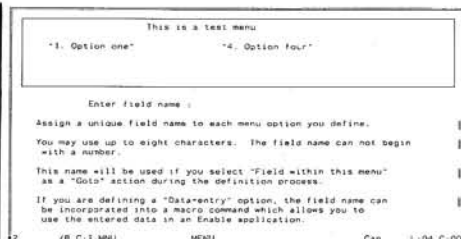
**Figure 3**  
**Menu Creation Screen Three**

Because you are in the word processor, all commands and attributes are available. If you want to use the block character set, it is available. Type in the fixed information in the desired position. You can insert anything you want as long as you remain within the window. You would type in the basic and selection information at this time. The menu is designed to permit the user to make selections. ENABLE's menu generator will assist you in building menus like the rest of ENABLE. Depending on how you designed the menu, the choices could be either in vertical rows, across the screen, or a combination of these. As you insert the prompt text place a " " or caret at the start and end of the text that will be highlighted during the selection process. If you have more than one selection per line, make sure that there is at least one empty space between the carets.



**Figure 4**  
**Menu Creation Screen**

When you are ready for the "active" fields to be inserted, the same command set you used to develop the input and



**Figure 5**  
**Basic Menu Without Active Fields**

output report screens for the data base are used. Again, this is part of the once you learn it, you can use it again capability of ENABLE. To insert these "active" fields, position the cursor immediately to the left of the caret for the first menu selection choice. Press "Shift/F9" and ENABLE will prompt you for a field name like in the data base input screen. Each field must have a unique name of up to eight characters. This name can not begin with a number. ENABLE will use these names to identify actions that you can specify.

After inserting this name, ENABLE will prompt for added information as to what the field is to do. There are up to three screens that you will have to work your way through. The first screen will prompt on actions to be taken when the field is selected. The first choice is type option desired, "Select only" or "Data-entry".

The *Select only* option is exactly that. It permits you to specify an action that is to occur when a key is pressed. Nothing more can be accomplished with this key. I use this key to call up a macro procedure in the menus I write. If selected, ENABLE prompts for the key that you want to use to call up the procedure. If you wish to use a function key or one of the other special purpose keys, ENABLE will permit this to a limited degree. Certain keys, ESC, Num Lock, Scroll Lock, Ctrl, Alt, Shift, Caps Lock or the function key F1 can not be used. Any of the other keys may be used to call the indicated procedure. To use one of the special purpose keys, you must use the format listed for the macros. (See article 14). If you use a single key, press it and press the <Return> key for the next prompt.

ENABLE will now prompt for the action required when the above key is pressed in the menu. You can tell ENABLE to Goto either a "Another Menu" or to a "Field within this menu". I use the Goto another menu if the action required is long or complex. In the school information system, ENABLE loads to a main menu. From this menu a basic selection is made to "Edit Records", "Input new Records", "Product Reports", "Go to Basic ENABLE", or "Return to DOS". Selecting one of these options, send ENABLE to the next menu where further selections can be made. The menu selected from this se-

lection can have a name up to eight characters and must have an extension. The second choice is to Goto another field within the menu.

The second choice on the *Specify Action* line is "Perform Commands". I use this option if the menu choice is to run a macro procedure within ENABLE like producing a report. If you select this option, the next line in the menu generator prompts for more responses. The first choice is "Standard ENABLE". If selected, ENABLE will run all screens just like you typed them in. I use this selection when I design a basic operation for somebody who is learning ENABLE. This will permit them to watch the entire procedure and learn from it. The second choice is "Menu". Selecting this is like inserting the {Voff} command in a macro. This turns off the ENABLE screen updating and will run ENABLE with only the menu displayed until a response is required. This command would be used where the menu is built for non-computer users who do not want to learn anything or you want to prevent from trying something. This will permit ENABLE to run faster as it does not have to continually update the screen. The last selection is "Message". If this is selected, ENABLE will prompt you for a special message that will be displayed on the screen while the procedure is being run. You can select one of four locations for this message. You must include a {Von} statement in the macro procedure that is called when "menu" or "message" are used. Failure to do so may result in the screen being "hung up" after the procedure is completed.

After completing the Screen Display line, you will drop to another line which will prompt you for the location of the macros or commands that the menu will run. These can be typed into the menu on the next screen or can be called from another file. If they are to be entered in the menu, the next screen will provide four lines in which to type in the commands. The normal macro procedures and structure are used for these commands. The lines are limited to 100 characters each, so procedure developed here can not be extensive. In article 14, a macro was built for inserting message headings. Because there could be multiple headings, it would be hard to remember which one to use. By developing a menu with the name of each heading, they can be called by using one keystroke. These macros are long so they must be called from a file. This option would be selected and the entire name of the file, including the extension, would be inserted.

The third option on the *Specify Action* line is "Run program." This is different than the macro procedure because this calls a MS-DOS program outside of ENABLE. These programs are available through the MCM-DOS window capabili-

Specify option type: 1=Select-only option 2=Data-entry option  
Enter key used to select this option:  
Specify action: 1=Goto 2=Perform commands 3=Run program  
Specify GOTO destination: 1=Another Menu 2=Field within this menu  
Enter menu name:  
When the menu is used, this option can be selected by the key you specify or by moving the cursor to the option and pressing enter. The action you specify will be taken. However, no data can be entered.

Defining field (option) ONE Cap

**Figure 6**  
Printout of Test Menu

ty of ENABLE. In order to use this capability you must have set aside some memory during the startup of ENABLE. If selected, you must enter the entire name of the file including the extension. The only acceptable extensions are ".COM", ".EXE", or ".BAT". This is another capability of ENABLE that is not used often. Version 2.15 will even permit capturing data in a DOS window for use in a document.

After completing the action screens, including the macro commands, ENABLE's menu generation program takes you on to the information screen. The two lines required here will provide an enhancement to the overall menu. As you work through ENABLE, for most selections, some information is provided in the area at the bottom of the screen. ENABLE's menu generation program will permit you to insert the same type information. The first line prompts you for the location of the message if one is to be displayed. All of the options, except one, limit you to a one line message. These one line messages can be displayed on the next line, on the last row, of the menu or on the status line. If you would like to display more than one line, you must select *Window*. If this option is selected, the size and location of the window will be required. I use this option to provide a limited amount of information to users of programs that I write.

[unc] ]  
[ ] ]  
[ ] ]  
[ ] ]  
[ ] ]

ENTER YOUR MACRO COMMANDS: You may enter up to four lines of commands. Letters within a single [ ] command must all appear on the same line. Line one cannot be blank.  
To reference data "passed" to the MCM from an "Entry of data allowed" prompt, use [%FIELDNAME] where FIELDNAME is the name of the menu option. This will be replaced by the entered data when the macro is executed. (IMPORTANT: PLEASE NOTE that [ ] are used here, NOT { }.)

Defining field (option) ONE Cap

**Figure 7**  
Macro Input Screen

The next line permits you to insert a HELP message for the user. If this option is selected, the user of the program can request "HELP" by pressing the "F1" key. This help screen can be used to provide guidance to the user so that you will not receive all of the questions. The HELP function of ENABLE is created in this fashion. With the cursor on a menu choice, pressing the F1 key will open a window with help on the next step. Like the mes-

sage line, the help message can be limited to one line or an entire window.

If you select one of the message options, ENABLE will now prompt for the message. If you have selected one of the one line messages, you will be provided with a one line input screen. Whatever can be inserted on this line will be displayed. If you have selected the *WINDOW* option, you will be prompted to size and position the window and then add the text. After you have completed the message, press Shift/F9 to redisplay the menu screen to enter the next menu item. After you have completed the entire menu, press F10 and the "Form (Menu) Design Options." Here, select (S)ave to save your work. ENABLE will save the menu as a word processing document that you can view. More information on the menu in the word processor later. Note that when you look at the menu file in the word processor, the command set is located inside of a comment area. This must be moved or copied into the body of the text in order to print it. ENABLE will return you to the menu screen where you will have to press F10 and "Form (Menu) Design Options" and quit to move back one menu level. To test your work you can press the (T)est option or (Q)uit back to the main menu.

Info msg: 0=None 1=Next row 2=Last row 3=Status line 4=Window  
Help msg: 0=None 1=Next row 2=Last row 3=Status line 4=Window

Select "None" if you do not wish to display an explanatory message when the cursor is moved to this option on the menu.

Revising field (option) ONE Cap

**Figure 8**  
Field Message Prompt Screen

A second option in developing a menu is the data entry capability. Using this option you can use the menu to prompt for an input which can be used in macro procedures. If you plan on using this option, you do not have to indicate the size of the field or any words on the menu. The size of the field will be determined when you indicate the length during the definition prompt. If this option is selected, ENABLE will prompt for the type data to be used. You can specify numbers, text, or other. If *numbers* are desired, you can indicate the number of decimal points required. If *text* is specified, the type character that can be inserted is required. You can specify anything, numbers and letters, letters only, or special field definitions. If special fields are desired, you can indicate the character(s) desired in each position. This is like the picture capability in the data base field definition. All separators can be inserted during this definition. The last type of data is *Other*. Under the *Other* option three types of data can be inserted,

date, time, and file. Like the special fields above, ENABLE permits you to indicate the format or picture the data *MUST* take during the input. After inserting the definition for the field, ENABLE then requests information as to Error messages. Then the remaining questions are the same as the basic menu entry definition.

One way to use this data entry procedure in a program would be like the payroll program that I wrote. This program uses the pay date information while it runs the checks and the twelve fiscal reports. Using this capability, after selecting the reports options from the report menu, the menu cursor moves to the data input line and there you would insert the check date. ENABLE would then use this information in the reports so that you would not have to insert it several times. The output of the menu data input can be used in other ENABLE functions by inserting [%fieldname] where it is needed. Note that it is enclosed in brackets and NOT braces that are used in macros. I also use this same field name on the report label line for the "As of" date.

This is a test menu  
"1. Option one" "4. Option four"  
INFORMATION MESSAGE DISPLAYED WHEN CURSORS RESTS ON OPTION.  
[Last Row]  
Only one line of text will be displayed when you use the menu, regardless of the number of lines you enter here.  
After you have entered (or revised) the one line of text, then press SHIFT/F9 to save the text and continue.  
This is option one

2 /B C.T.MNU MENU Cap L:01 C:08

**Figure 9**  
Message Input Screen

If you are using a color monitor, you can develop the menu in color as an enhancement. At the menu generation screen press F10 "Form (Menu) Design Options" and then "Color" from this menu. ENABLE also permits you to access the color options by pressing "Shift/F1" at anytime in the menu generator. You can then select the available colors which are displayed on the bottom of the screen. Again, as with all of ENABLE, the attribute, in this case color, can be applied to any part of the text. The text segment control commands are used for this purpose. Remember these commands are Shift/F0 W (Ctrl/W for the PC) for word, Shift/F0 L for line, etc., and they are the same as the word processor commands. The first choice in the color option is the area you want the color applied. Pressing "0" will permit you to define exactly where you want the colors to be applied for the background. You have five options here. The first option controls the basic background of the menu and the background color around the letters. The second options permits you to change the background color of the characters on the menu. This will provide a block of color around any characters on



the menu. The third option permits you to change the window background color only. The fourth option will change the background of the characters only. The fifth option will permit you to change the intensity of the character color. The basic PC seven colors are available along with high intensity colors, and ENABLE will use this capability.

When I had the color monitor on the Z-100, I was able to paint unique colors for the field backgrounds. I have tried this on the PC version and it also works. Colors can be overlaid on top of one another to create a new color. By using red and green, a yellow color can be displayed.

## Great Deals

2400 Baud Modem	\$90.00
WD Hard Disc Kit	250.00
Lazer Printer	800.00

**BEA-SOFT Computers**  
**P.O. Box 9193**  
**Gosnell, AR 72319**  
**(501) 532-5946**

Reader Service #168

## FORM FILL-R

### NOW WITH FORM EDIT-R

**FORM FILL-R:** Fills in pre-printed or user-created forms up to 132 columns by 132 lines.

**FORM EDIT-R:** Creates forms; handles field placement and attributes; cut & paste rectangular regions of text; draws lines and boxes using line-drawing characters, and automatically joins lines together for simple form layout; many other features.

**CREATE:** Custom forms, charts, work sheets; your data entry screen can be either the printed form or a data entry screen which you design.

**PRINT:** Forms directly, or save in disk files.

**SAVE:** and Recall to change previously filled forms.

**EXAMPLES:** Several are included.

**REQUIRES:** IBM Compatible or Z-100 Computer; MS-DOS 2.0 or greater.

**\$49.95 Postpaid**

Reader Service #136

LINDLEY SYSTEMS 4257 Berwick Place  
 Woodbridge, VA 22192-5119 (703)590-8890

You will have to work with this to get the hang of the new colors. Note, the Software Group does not suggest doing this.

Using the color options, you can create very nice looking menus. With the capability to change the color of the letters, background of the letters, and information areas, helpful menus can be developed. Using the attribute commands, you could even change the colors within words if you so desire. To change the colors or to remove them, use the F0 N (Alt/N for the PC) and then the attributes commands.

```

Test Menu
*1. Number One *4. Number Four*
*2. Number Two *5. Number Five*
*3. Number Three* *6. Number Six*
Enter Date
*
Comment:
.MENU DIVISION VER=1.0
.W=(010,014,020,070) C=3
.O 1 (003,006,003,021) ONE (004) V=1
.CO=UWC(ESC)
.IS C=1 T=This is option one
.O 4 (003,035,003,049) FOUR (033) V=1
.CO=UWC(ESC)
.IS C=1 T=This is option two on the status line
.O 5 (005,035,005,049) FIVE (033) V=2
.CO=UWC(ESC)
.O 3 (007,006,007,021) THREE (004) V=1
.CO=UWC(ESC)
.IS C=1 T=This is option three
.O 6 (007,035,007,049) SIX (033) FI=MDATE
.O 0 (010,028,010,032) MDATE (023) V=1 DA=(008,008,010,000) PIC=YY/MM/DD
.CO=UWC(ESC)
.END

```

**Figure 10**  
**Test Menu**

To change any of the menu selection you must use the top line menu in the "Form (Menu) Design Options". The options available here will permit you to delete a field, revise a field, and permit marking of a field. You can use the word processing commands to move fields if necessary.

The menus created using the menu generation program can be called from ENABLE at anytime. As discussed earlier, the naming of the menu will determine how it can be called. The default menus are named with the two letter identifier of the application they are to run in. These menus are invoked by pressing Shift/F10. There can be only ONE such menu per application. The other menu is the one letter type that can be called from any application. To access this menu press Shift/F0 F10 (Ctrl/F10 for the PC) and the letter on the menu.

As I said earlier, it is possible to work with a menu in the word processor. Figure 7 is the menu file displayed in Figure 5. The first line is "MENU DIVISION" title required by ENABLE. This is the same type first line as the data base procedural language uses. In fact the menu structure is very similar to the dot commands of the data base procedural language. The next line is the window line. This line defines the size and location of the menu window on the screen. It is defined by line and column for the top left and bottom right corners. ENABLE uses the top left corner as position 1,1 and the bottom right hand corner as position 23,78. At the end of this line is the background color of the window. The next line is the first option definition line. After the "O" for op-

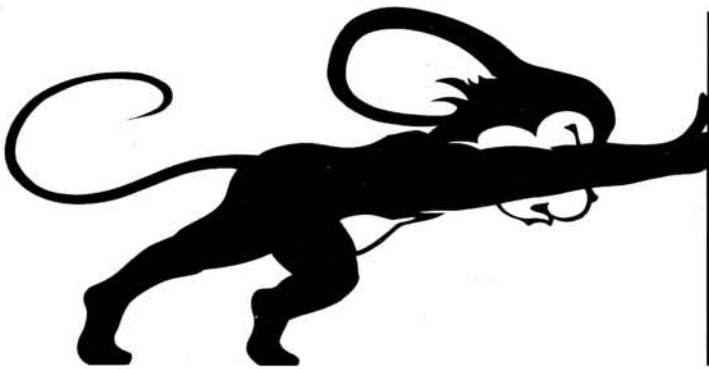
tion, the next item is the key used to call the menu item, in this example the number "1". Next comes the cursor position but this time relative to the window corner. Following the position is the field number, in this example "ONE" and the position on the line (1 after the start). The last command on the line is how the commands is called for as the menu/macro is performed. In the example, the "V=1" indicates that the ENABLE will display all screens as it runs. The next line, ".CO=UWC(ESC)", is the macro command inserted into the menu program. This will be the series of commands that ENABLE will follow when the "1" is pressed in the menu. The next line is the information code. When I built this menu, I indicated that when this menu item was highlighted, an information line would be displayed on the last line of the window. Five lines below this is the information line for the second menu selection, "IS C=1 T=This...". This is different in that I selected the line to be displayed on the status line (IS). The next item on the line is the color of the text, C=1. Following that is the text, "T=This is option one", for the information line. The same type coding would be used for error messages (.Ex), help messages (.Hx), or execution messages (.Ex). The second part of the message is the location, line below the option (R), last line of the window (B), the status line (S), or in a separate window (W). Near the bottom of the menu is option six, which is GoTo call. The "FI=" is the name of the menu field that the call has to go to, in this example "MDATE". The next line is the location that is called and is a data input request. Again it has the location, field name, offset and indicator to run normal ENABLE screens. Next comes the information on the data input, the minimum and maximum length, data type, and an ENABLE reserved field of "000". Since I indicated a picture, that is also inserted. The next line is the actual macro commands.

If you use the word processor to edit your menu, be sure to test it from the MCM Tool Menu Test function. If an error is detected, it will be highlighted so that corrections can be made.

This has been a brief discussion on the ENABLE menu capability. This capability adds a lot to ENABLE and makes it an even better product. It is a function that most people will not use, although if you continue to do the same task over and over again, this will help. The menus and macro capability of ENABLE make it possible to develop and run very complex functions from within ENABLE without the user knowing what is going on.

If you have questions or comments on this series, please feel free to drop me a line. I will be glad to help or answer your question or will get an answer if I do not know. Until next month. \*

William M. Adney  
P.O. Box 531655  
Grand Prairie, TX 75053-1655  
Copyright © 1988 by William M. Adney. All rights reserved.



## Microsoft Bus Mouse, Logitech Serial Mouse, Softterm PC, Z-100, PC Compatibles, Look and Feel Issue, MCA Versus EISA

# On the Leading Edge

As I write this, 1988 is fast drawing to a close. For most computer users, 1988 was a year of contradictions. Very little really new software was announced, and what was announced was late — I call that “oilware” because the originally announced release dates keep slipping. Lotus 1-2-3 version 3 and SuperCalc 5 are a few good examples of oilware, not to mention dBASE IV. One could also call that frogware because the release dates keep jumping around . . .

For hardware, there really weren't too many surprises either. Many manufacturers have announced 25 MHz 80386-based computers, and rumor has it that 33 MHz units will probably be available in 1989. And we can probably look for the first 80486-based computers to be available near the end of the year. Perhaps the biggest hardware-related news of 1988 was the announcement of the EISA (Extended Industry Standard Architecture) bus that was intended to counter IBM's PS/2 series computers with the Micro Channel Architecture (MCA) bus. We'll take a look at what I think these hardware changes mean later in this article.

Speaking of hardware, I suppose that it was inevitable that it would happen to me sometime — the company that manufactured Felix (February 1989 REMark) is out of business. If you call the number listed for Lightgate in last month's column, you will get one of those “disconnected or out of service” messages. I discovered that because I usually check back with a vendor when a product will be discussed in my column, especially if the vendor is new. Although Felix had some significant advantages, as I reported last month, it appears that its appeal was not

wide enough to justify staying in business. At this point, the only real competitor to the mouse is the trackball approach, and I think that the Mouse-Trak is the best of them all.

Perhaps the most disturbing news of the year is that, in retrospect, it appears that 1988 is the Year of the Lawsuits. Apple sued Microsoft for alleged copyright infringements in MS Windows. And Ashton-Tate (dBASE) sued Fox Software in a look-and-feel suit involving the Foxbase+ data base software. I expect that the outcome of these lawsuits will probably have lasting and profound effects on us all.

### The Look-and-Feel Issue

Depending on its outcome, the Ashton-Tate lawsuit will probably have more influence on the look-and-feel issue than just about anything to date. The initial allegations in the lawsuit seemed to center on a statement that some of the Foxbase+ screen displays were identical to dBASE. Now it appears that Ashton-Tate is also claiming that the dBASE language itself is also protected by copyright which, according to the modest research I have done, is contrary to currently accepted legal practices, customs, traditions, and laws.

Even though I am not a lawyer, it appears that both human and computer languages are specifically excluded from copyright protection, and the common sense of that needs no explanation. Can you imagine where we would be without the development of better computer languages? That issue is particularly important as we move on to the 4th Generation Languages (4GL), Artificial Intelligence (AI) programming languages, and SQL (Struc-

tured Query Language).

In short, it seems to me that the Ashton-Tate lawsuit is akin to using a steam roller to crush an ant. Reports are that Ashton-Tate has something over 60% of the data base market for the dBASE products, and Fox Software has something like 5 percent. I have fooled around with both dBSAE III+ and Foxbase+, and even though some of the display screens are quite similar, they are also clearly different products that have different features. In my opinion, that explains why Foxbase+ was developed, and my experience is that Foxbase+ is a better product, primarily because it has far fewer bugs and more features than a comparable version of dBASE.

The popularity of a graphics computer interface has been another issue, and Apple sued Microsoft for the similarity of the Windows interface to that used in the Mac. The outcome of this lawsuit is still up in the air, and there is no telling how it will be resolved. Apple's position is that Microsoft Windows, and presumably the OS/2 Presentation Manager as well, has a look and feel like the Mac's Finder program. I think that is really stretching a point because it is really difficult for me to see how the look and feel of the Finder program is similar to Windows.

As you might expect, opinions on each of these lawsuits vary widely and pretty much depend on which side you are on. For those of you who don't know any Mac users, they seem to have almost a religious fervor in their dedication to the Mac. Indeed, many Mac users seem to think that Apple single-handedly invented microcomputers. Although Apple is making some inroads into business



computing because there are certainly some applications that the Mac does quite well, the PC compatible is still being used by most businesses. And the popularity of a graphics interface, like the Finder or Windows, is increasing because many users find that these interfaces are easier to use than the standard command prompt used in PC compatibles.

There will always be some Mac and dBASE users who will contend that their interfaces are the best and should not, and cannot, be "copied" in any form by other manufacturers. I think that is a very short-sighted and head-in-the-sand approach to computing, in general. When I was teaching at UTA, my department head had a Mac, and he had used it to write several books that have been used for various courses. His view was that a good user interface should be available to everyone, and although he thought the Mac interface was better than the DOS command prompt, he recognized that the Mac interface was not for everyone.

There is one other facet of both these lawsuits that stands out in my mind because it is perhaps the common problem that I see with both of them. Neither the graphics interface, nor the dBASE language were "invented" directly by either of these companies. The model upon which the Mac's Finder program is based was developed at the Xerox Palo Alto Research Center (called PARC, as in "park"). One only needs to take a look at the Xerox Star computer system to see the striking similarities between it and Finder. Similarly, the model for the dBASE language was developed by Wayne Ratliff while working at the Jet Propulsion Laboratory (JPL) in Pasadena. The basic model was developed on a mainframe, and it seemed like a good idea to develop something similar for a microcomputer which later was called dBASE. Wayne eventually sold this product to a company which is now known as Ashton-Tate, and he also worked for them for quite a while in the development of dBASE.

While it is true that both products were actually developed by the respective companies into the forms we know today, it is also true that both products did not actually originate in either company. For that reason, I admit that I really do not understand these lawsuits, and I suppose that much of the basis for them must be buried in some musty, old law books somewhere. I think it is unfortunate that the rationale and common sense behind some of these laws seems to have disappeared somewhere. And there is also the added problem that the courts and laws are unable to keep pace with the rapidly changing technological environment that we find ourselves in. That's no real surprise because even computer specialists and experts have difficulty keeping up with the current technology.

## **MCA Versus EISA**

Many of you are probably aware of the controversy surrounding the issue of which bus is the best for a personal computer. The original PC and AT bus was considered to be a standard which is generally referred to as Industry Standard Architecture or ISA. In an apparent attempt to recoup losses from all of the clone vendors, IBM developed the Micro Channel Architecture (MCA) bus for their PS/2 series computers and announced that they would demand royalties from anyone who dared to copy it.

Despite IBM's claims to the contrary, the PS/2 MCA has not yet demonstrated any significant advantage over the basic ISA bus that was used in the PC/XT/AT series except that it is proprietary. Bill Lowe is no longer president (he resigned) of the IBM division that makes small computers, and sales of the PS/2 series have been dropping, mostly in favor of other manufacturer's compatible systems. One can find some evidence of the bad business decision to discontinue the IBM AT by the fact that IBM now has a new "AT compatible" PS/2 model available, too.

To counter the so-called "MCA threat", several vendors have announced the Extended Industry Standard Architecture (EISA) bus. Zenith is one of the major participants in this effort, along with Compaq and Tandy. There seems to be a considerable controversy as to which is better, MCA or EISA. I have read several articles which state unequivocally that MCA is clearly the better choice. I am amused when I read an article like that because there is no logical way that anyone can make that statement yet. Why? Well, it turns out that the EISA specification has not been finalized as of the time of this writing, and I have not figured out how anyone can compare a developed specification for MCA with one that is probably in the draft stages. For my part, I have adopted a wait-and-see attitude before I make any kind of judgment about which bus is better. I believe that the advantages of one or the other will become quite clear this year, and for now, I suggest staying away from the MCA bus until we know more about the alternative.

So much for 1988, let's move forward to 1989 with a look at the most common pointing device you will find today. And I have included a look at two different brands and types of mice to give you an idea of the possibilities.

## **Of Mice and Mouse**

Since much of today's software can use a mouse to make doing things a lot easier, there are at least two possibilities for connecting a mouse to your system: a bus mouse and a serial mouse. In most cases, I recommend a serial mouse for the reason that you can connect it to almost any computer, even most laptops. Not

true with a bus mouse. When I buy equipment of any kind, I try to find something that will work with all my systems with a minimum amount of fuss. Even my Z-171 has a serial port, and I can use a serial mouse on it if I want. For similar reasons, I usually recommend an external modem so that it can be used with nearly any computer by simply using the appropriate cable. You won't be able to install a desktop's internal modem in most laptops, but there are more obvious reasons for having an internal modem with a laptop than there are for a desktop. And if you find that your desktop needs a serial port for both a mouse and a modem, a simple A-B switch can be used to select the device you need. In most cases, you do not really need a mouse to work with most telecommunications programs.

In any case, let's take a look at the Microsoft Bus Mouse and the Logitech C-7 Serial Mouse.

## **The Microsoft Bus Mouse**

The old Microsoft Bus Mouse works just fine in both my '248 and '386 computers. I say "old" because it is not the latest style, but I am not convinced there is any particular advantage to the very latest version. I have used the Microsoft Bus Mouse with a wide variety of software, and as you might expect, it works with all programs I have tried, whether or not they support a mouse. One of the reasons that I have been able to do so much with the Microsoft Bus Mouse is because I also ordered the optional (for \$25) Programmer's Package. I presume that Microsoft is still including the ordering information with each mouse, but I have also included the ordering information for it at the end of this article.

Installing the MS Bus Mouse is reasonably straightforward and essentially amounts to plugging in a half-card to an unused slot in your computer. Like many cards of this type, the MS Bus Mouse has a jumper that allows you to set the hardware IRQ (interrupt request) to be used for it. The manual suggests that IRQ 2 is fine in most cases. In general, I suppose that's true, but there are a couple of minor things you need to know about IRQs before configuring it. And because of some differences between the PC and XT compatibles (e.g., the Z-150 series, etc.) and the AT, you need to understand a couple of things about IRQs, in general.

Interrupt requests, or IRQs, are hardware features that are based on a special chip inside your computer called an 8259 Interrupt Controller. In general, one of the key differences between an XT and AT compatible is the number of 8259s that each computer has. A PC or XT compatible, such as a Z-151 or '158, has one 8259; an AT compatible has two 8259s. And there is a difference as to which IRQ is used for what between the two types of

systems. Because of that difference, you must know what you are doing when you set up a new internal board, especially a bus mouse.

The Microsoft Bus Mouse allows you to change the IRQ number used in the 2-5 range. In most cases, 2 is recommended and works fine, but there are some exceptions. Listing 1 shows the IRQs used by the PC/XT compatible Z-100 PC series (i.e., all Z-15x model computers) and the AT compatible Z-248/386 systems.

	Z-100 PC	Z-248/386
IRQ	Usage	Usage
2	Z-319 card	2nd 8259 Controller
3	COM1	COM2
4	COM2	COM1
5	Hard drive controller	LPT2

**Listing 1**  
**IRQ Usage by Zenith Computers**

As long as you are not using a Z-319 card, the IRQ 2 works fine in any of the Z-100 PC computers. If you are using a Z-319 card, you will probably want to use the IRQ 4 setting so long as you are not using the COM 2 port. If you are using a Z-319 and the COM2 port, you will have to make a choice as to which one you want to keep since you can't have both. Perhaps this example makes it clearer why I usually recommend a serial mouse in the first place.

For the Z-200 series and Z-386, you can choose IRQ 3 if you are not using the COM2 port or IRQ 5 if you are not using the LPT2 port. Since these IRQs are not commonly used in most computers, you should not have much difficulty with them. For those of you interested in some additional discussion of IRQs, I am currently working on an article that describes the standard eight IRQs used in 8088-based systems (e.g., '150 series, '148, etc.), as well as the 16 IRQs implemented in the newer systems. With that digression, I will get back to the Microsoft Bus Mouse.

Once the bus board is installed, all you need to do is plug in the MS Mouse, set up the driver in CONFIG.SYS, and you are ready to use it. Of course, your software needs to be able to use a mouse, but even if it is not, you can use the optional Programmer's Package to set up just about any software. I have used the package to set up menus and such for WordStar, and the package includes plenty of examples of many types of menus that you can implement. All "menus" are written in an easy-to-understand source code, although it takes some practice to use it effectively.

The Microsoft Mouse is a 2-button mouse, and it has a good, solid feel. I have used it for a lot of different tasks, but I have found that I really prefer a 3-button mouse.

## The Logitech Serial Mouse

I have used a lot of different mice, and my favorite of all is the Logitech Serial Mouse. Although this mouse is also available as a bus mouse, I have found the serial version is much better for my needs. And if you decide to get a Logitech mouse, be aware that there are actually two different packages that you can find.

The first is the standard Logitech Serial Mouse period. The one you probably want is the Logitech Plus package that includes all of the software required to generate your own menus, plus a remarkable number of examples for just about all major software. In other words, this package includes all of the "optional" software that was not included in the Microsoft package, and the price for the Logitech Plus package is a little more than the mouse by itself. I think the extra cost is worth it, particularly if you like to fool around with these kinds of things.

Interestingly enough, the "programming" language used for the Logitech mouse is very similar to that used by Microsoft which is no real surprise. Perhaps the major difference is that you can program "chords" which are combinations of the three buttons: right and left buttons, right and middle buttons, left and middle buttons, and all three buttons. Then, of course, you can program each of the three buttons individually for a total of seven button functions. With a 2-button mouse, you can only program three possibilities: right button, left button, and both buttons. How important this is to you depends on your particular application, but I have found the 3-button Logitech mouse to be exceptionally useful for drawings.

Implementing the Logitech mouse is quite simple. First, connect it to your system, and then copy the appropriate driver from the disk. If you have a hard drive system, you can copy everything, and take a look at the many things you can do.

Because of the 3 buttons, the Logitech mouse obviously has a different "feel" than the 2-button Microsoft Mouse. Like most of these kinds of things, what you like is primarily a matter of personal preference, but I have found that the Logitech is best for me. In choosing a mouse of any kind, I suggest that you find a computer store that sells them so that you can get an idea of what they feel like before you buy one.

In the previous two articles, I have discussed two different kinds of pointing devices: the Mouse-Trak (January 1989) which is a trackball-based pointing device, and the Felix (February 1989) pointing device. In this article, I have mentioned two different brands and kinds of mice. What do I like best?

## Choosing a Pointing Device

For general purpose stuff — ranging

from word processing, spreadsheets, and drawings (e.g., CAD) — the Mouse-Trak wins hands down. True, you can do some special things better, like draw a curve, with a mouse or Felix, but I think that the trackball approach is best for most general applications. As I mentioned in the first article, it does not take much space (neither does Felix), but the Mouse-Trak is able to use a relative reference (like the mouse) which is probably easier for most users. And the fact that the Mouse-Trak seems to work just fine with the standard Microsoft and Logitech menu software is also a big advantage.

Felix has the capability to program "hot spots" that can be the equivalent of function keys, and for some users, this may be a real advantage. The hot spots can contain macros that allow you to perform your work a lot easier, and there is little argument that, in many cases, Felix is probably faster because its movement is basically limited to a single screen. That is, you can't get the continuous movement with Felix like you can with a mouse or a trackball. With a little practice, you can really move around using Felix, and I found it to be an interesting experience. Although I experienced no problems with Felix, I am still waiting for information about the programmer's package that is apparently available.

Last, but not least, is the standard kind of mouse. Although the Microsoft Mouse is nice, I prefer the 3-button Logitech Mouse. Both are compatible with nearly everything.

What do I recommend? Perhaps the most important thing is to make sure that whatever kind of pointing device you get is compatible with either Microsoft or Logitech mouse software. I do not recommend buying anything that is not compatible with both — that is one reason that I really like the Mouse-Trak. It seems to be compatible with all of the common things you will find in any software that supports a mouse. In short, my first choice is the Mouse-Trak.

My second choice is the Logitech Serial Mouse in the Plus Package because it is easy to implement, works on nearly all computers, and allows you to do all kinds of neat things with the software.

The Microsoft Bus Mouse works fine in my desktop systems, but I don't really find it appropriate for me because it is difficult or impossible to move to my other systems, such as my Z-171.

## Softerm PC

When you start thinking about expanding your own computer world, one of the most obvious ways to do so is adding a modem and a communications program. I suggested earlier that an external modem is probably the best overall choice because you can move it to a different computer. Choosing a communica-



tions program is not nearly so easy. There are a wide variety of communications programs available, and in general, their prices reflect their complexity and capability. The lower-priced programs do not generally have all of the bells and whistles of the more expensive versions, but they do the job quite adequately. And since the less expensive programs are not as complex, they are also easier to learn.

In the less expensive communications program category, I think that the HUG Modem Communications Program (HUGMCP) is one of the best bargains available. It has many of the features of the more expensive programs, and it has the added advantage of being easy to learn for beginners. Even advanced users may find that HUGMCP is the only communications program they will ever need.

When you move to the real power-user category, you may find that you need or want a communications program that is more powerful. One good choice is HyperAccess that I have mentioned in this column on many occasions. But not too long ago, another program, called Softerm PC, was brought to my attention, and it may be the ultimate in communications programs.

Softerm PC is a heavy-duty communications program that has a number of features which carry it far beyond what you will find in most similar programs. In case you have a Z-100, you will be pleased to know that Softerm PC supports it, except that you cannot use color in the version 3 which I tested on my Z-248 and Z-386. You will need an absolute minimum of 256 KB of memory to run Softerm PC on any computer, and more memory is better, especially if you intend to use the Communications Agent in the memory-resident mode. The literature says that Softerm PC will run on a DSDD floppy drive system, but I tested it by loading it on my hard drive.

Softerm PC comes with 8 distribution disks which explains why I recommend a hard disk. Although that sounds like a lot of disks, three of them contain over 40 terminal emulations, and the configured program does not require as much disk space as it sounds like. And Softerm PC directly supports over 20 different modems including all Hayes compatibles, as well as direct support for my own Prometheus ProModem 1200. By the way, if you are considering buying a modem, be sure that you get a Hayes compatible modem to ensure that it will work with just about all communications software.

The power of Software PC does not become evident until one gets into the details of the system. Yes, Softerm PC is more of a communications system rather than just a program. And it has two major functions called the Communications Agent and the File Agent. The Communications Agent is the control center for the

communications function and manages all of the computer's resources. Because it is so powerful and can be used for other functions, the Communications Agent can be installed as a memory-resident program in addition to the normal stand-alone mode.

Basic functions managed by the Communications Agent include: terminal emulations (if needed), background job control, queue management, keyboard macros (with a learn mode), and the DOS interface. You can also access the included disk utilities, built-in text editor, and perform local file transfer. As you might expect, Softerm PC included a fully developed script language that is so powerful you can use it to write a bulletin board system. If you need that kind of application, you might want to check with Softronics because they have already written the script for a bulletin board. And like any good communications software, you will find an electronic phone book.

The File Agent basically is used to transform your computer into a terminal for access to a host computer, such as the DEC VAX or HP 3000. You can access data on a host computer, and also receive electronic mail with automatic notification of receipt. If you have any special needs for mainframe communications, you may want to check with Softronics to see if your requirements can be, or already have been, implemented.

A powerful system of this kind is quite complex, and I recommend that you get some experience with a simpler program before you consider Softerm PC. Although I would imagine that a beginner could learn and use Softerm PC because of the very good documentation provided, it is not something you will be able to completely learn overnight or even in a week. For example, the script language is, well, a complete language; and it takes some time to learn the basics, let alone master it. The language is powerful enough to do some things that you might not normally expect. To give you one example, Rik Gugeler (Colorado Springs, CO) tells me that he wrote a class tracking and grading program using this script language. Now that's power!

If you have a need for real heavy-duty communications, Softerm PC can certainly handle it. Highly recommended when you need this kind of power.

### Powering Down

1988 was an interesting, but fairly dull year as far as computers were concerned. For 1989, I already have some interesting products that may help you solve a problem. From a user perspective, it still appears that the OS/2 operating system is a solution looking for a problem to solve. I believe that, until there is some spectacular new application that requires OS/2, it will probably not become very popu-

lar because of its cost: both hardware and software. In the meantime, you still might want to have multitasking on your system, and I have found a good way to do it: Quarterdeck's DESQview. Unlike OS/2, DESQview does not require an 80286 system or later, and you can use it on your 8088-based systems. Since it will probably take me at least another month or so of testing, I expect that my comments on DESQview will probably appear in the early summer. Although I am actually testing DESQview 386 on my Z-386 system, this version includes DESQview 2.2 which runs on the Z-150 series (and other 8088-based PC compatible computers), as well as the QEMM-386 control program that is specific to the 80386 systems. If you REALLY need multitasking, I am already convinced that DESQview is the best choice available in both price and performance.

Speaking of performance, I will address the question of whether or not you really need to upgrade to a more powerful computer next month. Is the speed of the computer really that important? Find out next time.

For help in solving specific computer problems, be sure to include the exact model number of your system (from the back of the unit), the ROM version you are using (use CTRL-ALT-INS to find it), the DOS version you are using (including both version and BIOS numbers from the VER command), and a list of ALL hardware add-ons (including brand and model number) installed in your computer. The list of hardware add-ons should specifically include memory capacity (either added to an existing board or on any add-on board), all other internal add-on boards (e.g., modems, bus mouse or video cards), the brand and model of the CRT monitor you have, and the brand and model of the printer, with the type of interface (i.e., serial or printer) you are using. Also, be sure to include a listing of the contents of the AUTOEXEC.BAT and CONFIG.SYS files unless you have thoroughly checked them out for potential problems (e.g., TSR conflicts). If the problem involves any application software, be sure to include the name and version number of the program you are running when the problem appears.

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

### Products Mentioned

HUG Software  
HUGMCP (885-3033) \$40.00  
Heath/Zenith Users' Group  
P.O. Box 217  
Benton Harbor, MI 49022-0217  
(616) 982-3463 (HUG Software only)

HS-386 Computer Kit \$3349.00  
 1 MB Memory Board for Z-386 (Z-505) 799.00  
 4 MB Memory Board for Z-386 (Z-515) 2999.00  
 Cache Memory Board for Z-386 (Z-525) 599.00  
 FTM Monitor (ZCM-1490) 999.00  
 Heath/Zenith Computer Centers  
 Heath Company Parts Department  
 Hilltop Road  
 St. Joseph, MI 49085  
 (800) 253-7057  
 (Heath Catalog orders only)

Logitech Serial Mouse Plus \$109.00  
 Logitech, Inc.  
 805 Veterans Blvd.  
 Redwood City, CA 94031

Microsoft Bus Mouse \$139.00  
 Microsoft  
 13221 SC 26th St., Suite L  
 Bellevue, WA 98005  
 (800) 426-9400 (Orders only)

Softerm PC \$195.00  
 Softronics, Inc.  
 7899 Lexington Drive, Suite 210  
 Colorado Springs, CO 80290  
 (800) 225-8590  
 (719) 593-9540

\*

## Most home finance programs have serious limitations.

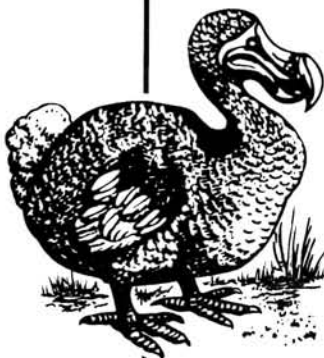
Like the dodo bird, they're slow, awkward and not very smart. You'll waste eons of time, only to realize they're dead ends.

HFS-III, the Home Finance System, is fast, loaded with features and intuitively smart.

- Easily manage up to 100 checking, saving, CD, IRA and other accounts.
- Print checks on any form.
- Organize financial summaries and activity reports based on 100 expense and 15 deposit codes you define.
- Flag tax-deductible or medical expenses.

In short, control your finances to save money, save time and simplify taxes.

Find out why so many leading computer authorities and users are calling HFS-III the "superior species." See it today at participating Heath/Zenith Computers and Electronics Centers, or call toll free for more info.



**Only \$49<sup>95</sup>**

Add \$3.50 handling & shipping.  
 Iowa residents add 4% sales tax.

Requires: DOS 2.0 or higher, IBM\*  
 PC/ XT/AT or compatible or Zenith\*  
 Z100 computer; 2 disk drives or hard  
 disk, 256K RAM.

Reader Service #137



Jay Gold Software, Inc.  
 P.O. Box 2024  
 Des Moines, IA 50310  
 (800) 541-0173

## New System Prices Getting You Down? Then Why Not Buy a USED System?



Complete Z-241 512k AT System,  
 including 1.2Meg 5.25" floppy,  
 mono card and amber Samsung(tm)  
 monitor, only \$1295 plus shipping!

Ready-to-run Z-151 128k XT with  
 360k 5.25" floppy and amber  
 Samsung monitor only \$695!

Many Others Available!

### Fully refurbished by authorized service technicians!

All systems come cleaned and thoroughly tested for operation to original manufacturer specifications. With the exception of a few minor scratches, most of these systems would pass for new equipment!

### Compare to New Systems Costing Twice as Much!

That's right!! The new models will cost you roughly twice as much as our refurbished units. Sure, they're a little faster, and you could be the first on your block to

own one - but *is that worth* an extra \$1000 or more.  
**Fully guaranteed for 30 days by First Capitol!**

We want you to be happy with our equipment. All units come with a 30 day replacement guarantee.  
**And for a limited time only, a 15 day trial!**

That's right! For the next 30 days only, you can purchase one of our refurbished units with a no-risk trial offer. If within 15 days of shipment you decide for any reason that you are unhappy with one of our units, you may return it for a full refund (exclusive of shipping charges) with *no hassle whatsoever!*

**Supplies ARE Very Limited! !**  
**...So, Order Yours TODAY!**

**First  
 Capitol  
 Computer**

#16 Algana Drive  
 St. Peters, MO 63376  
 Orders: 1-800-TO-BUY-IT  
 Tech: 1-314-447-8697

Reader Service #184



### \*\*\* Z-100 SERIES SOFTWARE \*\*\*

PART NUMBER	DESCRIPTION	LIST PRICE	SALE PRICE
MS-463-1	Z-Basic (16 bit)	\$175.00	\$12.00
MS-463-7	Multiplan	\$195.00	\$12.00
MS-253-1	Basic-80 (8-bit)	\$175.00	\$12.00
CD-463-2	Condor File Manager	\$299.00	\$12.00
LT-Z100	All 4 Listed Above	\$819.00	\$40.00

### \*\*\* IBM COMPATIBLE SOFTWARE \*\*\*

PART NUMBER	DESCRIPTION	LIST PRICE	SALE PRICE
MS-5063-30	Microsoft Windows	\$ 99.00	\$ 24.00
NU-413	Norton Utilities Adv.	\$150.00	\$ 99.00
WP-528	WORDPERFECT 5.0	\$495.00	\$269.00
BO-290	QUATTRO	\$239.00	\$179.00
CP-311	PC TOOLS DELUXE	79.00	\$ 68.00

### \*\*\* ZENITH LAPTOP COMPUTERS \*\*\*

<b>SUPERSPORT 184-1</b>	2 3 1/2" Floppy Drives, 640K RAM	\$1587.00
<b>SUPERSPORT 184-2</b>	1 Floppy, 20 Meg Hard Disk, 640K RAM	\$2373.00
<b>SUPERSPORT 286-20</b>	12/6 MHz, 80286 CPU, 3 1/2" Floppy, 20 MEG Hard Disk	\$3292.00

### \*\*\* VIDEO MONITORS \*\*\*

ZCM-1490	ZENITH Color Flat Screen VGA	\$718.00
MA2565	SAMSUNG Amber TTL 720x350	\$89.00
CW4644	SAMSUNG Color RGB 640x200	\$274.00
CM4531	SAMSUNG Color EGA 640x350	\$389.00
CN4551	SAMSUNG Multi-sync VGA 800x560	\$489.00
NC800	NEC Multi-sync II 800x560	\$639.00

### \*\*\* ZENITH PC COMPUTER UPGRADES \*\*\*

**SmartWatch from FBE Research** Installs in ROM Socket on CPU Board in Zenith computer series Z-100/138/148/150/160. This clock/calendar contains a ten year battery and keeps your computer informed of both time and date at each boot-up. Instructions and software included. . . . . \$38.00

**Z-150 Series Hard Disk Drive Kit** Includes new generation High Speed (28 MS) Seagate Drive with Auto Park heads. Each kit is complete with controller card, cables, hardware and instructions to mount the Hard Disk under your two floppy drives in the Z-150 series computers. 32 MEG ST-138/150 Kit . . . . . \$383.00

**Z-148 Hard Disk Drive Kit** Includes the Hard Disk Drive and controller in the kit above plus the Z-148 Expansion card described below. Each kit includes all cables, hardware and instructions required to replace one floppy drive with a high speed low power Hard Disk Drive. 32 MEG ST-138/Z-148 Kit . . . . . \$459.00

**ST-138/Z-148 Kit With SmartWatch** . . . . . \$489.00

**Z-148 Expansion Card** adds 2 IBM expansion slots . . . . . \$79.00  
with SMARTWATCH clock/calendar. . . . . \$109.00

**INTERNAL MODEM** Fully Hayes compatible (software included)  
1200/300 baud . . . . . \$94.00  
2400/1200/300 baud . . . . . \$159.00

**EXTERNAL MODEM** Fully Hayes compatible (software included)  
1200/300 baud . . . . . \$137.00  
2400/1200/300 baud . . . . . \$199.00

**VCE 150 Video Eliminator for Z-150**  
Allows use of EGA or any video card. Required memory chip included. . . . . \$54.00

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

### \*\*\* Z-100 SERIES COMPUTER UPGRADES \*\*\*

**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 Single Drive Unit 234.00 . . . . . \$351.00  
Bare Drive and Cable for internal mount . . . . . \$136.00

**SmartWatch by FBE Research.** If you don't have a clock for your Z-100, get this one. More details under PC upgrade listings . . . . . \$38.00

**Gemini Emulator Board.** Makes the Z-100 compatible with the IBM PC library of programs. . . . . \$432.00

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

**UCI Easy87.** Add an 8087 Numeric Coprocessor. \$69.00 for the board without an 8087 Chip. With 5 MEG 8087 \$188.00 or with 8 MEG 8087 installed . . . \$234.00

**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  
Requires 27 256K RAM chips to complete the kit.

**UCI Memory Upgrade Pal Chip Set** For the Z-100's with the newer motherboard part number 181-4918 or greater. This chip set allows the installation of 256K RAM chips on the motherboard. With the addition of 27 256K RAM chips, a total memory of 768K is obtained. PAL Chip Set . . . . . \$64.00

**UCI Memory Upgrade Card** We recommend this one highly. The board has sockets for up to 2 MEG of RAM. With no RAM installed \$288.00. Add \$35.00 for EasyDrive RAM Drive Software if desired. Either 64K or 256K RAM chips may be used to complete this kit.

**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 \$598.00, 31 MEG \$634. System without Drive . . . . . \$317.00

**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 . . . . . \$44.00

### \*\*\* FLOPPY DISK DRIVES \*\*\*

MITSUBISHI MF501	5.25" 48 TPI DS/DD 320K/360K	\$ 89.00
MITSUBISHI MF504	5.25" High Density 360K/1.2 MEG	\$106.00
MITSUBISHI M-353	3.5" in 5.25" frame 720K	\$98.00
MITSUBISHI M-355	3.5" in 5.25" frame 1.44 MEG	\$129.00
	M-355 Software Driver	\$ 19.00

M-355 runs on AT compatible or special controller only.

### \*\* SEAGATE HARD DISK DRIVES \*\*

ST-125	21 MEG, 28 MS, Auto Park Heads With Controller & Cables	\$ 275.00 \$ 329.00
ST-138	31 MEG, 28 MS, Auto Park Heads With Controller & Cables	\$ 329.00 \$ 383.00
ST-238	31 MEG, 65 MS, RLL With RLL Controller & Cables	\$ 258.00 \$ 309.00
ST-251	42 MEG, 40 MS, Auto Park, Software With Controller & Cables	\$ 384.00 \$ 438.00
ST-251-1	42 MEG, 28 MS, Auto Park, Software With Controller & Cables	\$ 465.00 \$ 519.00
ST-4096	82 MEG, 28 MS, Auto Park, Software.	\$ 647.00

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

### \* PAYLOAD CUSTOM ASSEMBLED COMPUTERS \*

We can assemble 8088 XT, 80286 AT or 80386 IBM compatible computers to your specifications. We use high speed turbo motherboards with Phoenix BIOS. Other features include clock/calendar, Keytronic 101 key keyboard, Mitsubishi floppy drives, Seagate hard disks, front panel mounted turbo and reset buttons, memory and I/O ports, all to your specification. Please write or call for price information. We can send you a work-up sheet showing items available and prices.

**Example #1**  
IBM XT Compatible 8088 10 Meg CPU, two 360K floppy drives, 640K RAM, one serial port, one game port, two parallel ports, 101 keyboard, clock/calendar, 10 Meg CPU and amber TTL monitor . . . . . \$831.00

**Example #2**  
Same as above with the two floppy drives plus a Seagate ST-238 30 Meg RLL hard disk drive . . . . . \$1126.00

**Example #3**  
IBM AT Compatible 80286 12 Meg CPU, 2 Meg RAM, clock/calendar, one each serial, game and parallel ports, one 360K floppy, one 1.2 Meg floppy, 101 keyboard, Seagate ST-251 40 Meg hard disk, EGA color monitor . . . . . \$2234.00

MS-DOS 3.3 Operating System . . . . . \$85.00

## PAYLOAD COMPUTER SERVICES



15718 SYLVAN LAKE  
HOUSTON, TEXAS 77062  
PHONE (713) 486-0687



Your satisfaction is guaranteed. All hardware carries a 90 day Payload warranty. VISA and MASTERCARD orders welcome with no surcharges. Add \$5.00 to all prepaid orders for handling and shipping in Continental USA, we pay the balance. Actual shipping costs for foreign, overseas and net billing orders to approved accounts. We accept purchase orders from schools, government and approved accounts. Mail or Phone your order for prompt service. Texas residents please add 8.0% state sales tax.

# Didactic Demonstration Of Outlining With WordPerfect 5.0

William N. Campbell, MD  
1063 Green Glen Drive  
Boothwyn, PA 19061

## Introduction

Many times, after installing a new editor or word processor into his computer, a neophyte learns some *fundamentals* which include activation of the new program, the basic mechanics of entering a document, some elementary method of inserting and deleting text, and finally, a method of saving the precious data to disk and printing it for later use and distribution. Unfortunately, too often the niceties of the program are overlooked.

The primary implementation of an editor or word processor is probably for the production of simple memoranda and correspondence. The formatting used for these is relatively rudimentary. "Parallel column" format is another familiar format where columns (often created by changing tab settings and typing Tabs) are utilized. An example would be the preparation of inventory lists. "Newspaper column" format is very commonly seen but probably rarely used by the average person.

An often overlooked but frequently employed text format is that of *outlining*. Outlined text is most often applied to the preparation of reports, guidebooks, primers, handbooks, manuals and other similar documents. It is characterized by different levels of indentation, frequently from the left, and sometimes from both left and right sides of the text.

Although quite commonly used, *outlining* is too often performed by using spaces and/or tabs at the left margin to produce the desired indented appearance, and this would be fine if the document were never to be altered, never to be edited. Unfortunately, most manuscripts that require an outlined type of format are later found to require multiple editing. This is where much time is wasted as the user tries to delete/insert spaces/tabs while at the same time deleting/inserting text, all in order to maintain the desired outline format. Most of the time utter chaos results. I know, because this has happened to me many times in the past.

Rather than discuss generalization of word processing at this time, I thought it might be wise to try to save some folks a great deal of time by addressing "Outlining", the easy way! I will assume that the "absolute basics" of your editor or word processor have already been reasonably mastered by you, and that your program is capable of generating "indent" codes.

Most word processor codes are entered with the keyboard function keys or alpha-numeric keys (often in combination with the Shift, Alternate, or Ctrl keys.) Frequently the codes are invisible to the user, although some Word Processors such as WordPerfect can provide a screen display of these "hidden" codes, and/or display the results of the codes. Since you won't see the actual codes in this treatise, I will describe them as I enter them. For example, if I write that "a **Shift-F4 ->Indent<-** code was inserted" that simply means that I held down the Shift with one finger and tapped the F4 function key with another, this combination putting an "**->Indent<-**" code in this document at the location where I was! You will be able to see the results of the codes.

## All Those Codes

A newcomer to word processing may well ask how one can remember the innumerable codes that a word processor must use. This is quite simple. Many word processors are supplied with "templates" which are cleverly designed strips or "shapes" of plastic or composition cardboard that are imprinted with the various codes and fit right on your keyboard. WordPerfect's templates use a red color to signify the CTRL key must be held down, blue for the ALT key, green for the SHIFT key and black to indicate just the function key itself need be pressed - this color coding, and imprinted code names makes it easy to refresh your memory. Also, a small Reference Card or Folder is furnished with most word processors. Most word processors have "on line" "context sensitive" HELP screens that are easily accessed from the keyboard

and display on your monitor screen rather detailed information about any given code, command, or function you may desire to check out. And, of course, there is the "MANUAL" which gives more comprehensive and detailed information!

## Outlined Text

Typed or printed material using an outline format can be produced with Spaces (an extremely hard way), with Tabs (a very hard way), and (with many word processors) with "Indent Codes" (the easy way). WordPerfect 4.2 and 5.0 have two such codes. One is the **F4 ->Indent** code and the other is the **Shift-F4 ->Indent<-** code. With WordPerfect, the former code indents from the left margin, while the latter indents equal amounts from both left and right margins. Both codes can be used in conjunction with WordPerfect's 'Outline' mode.

## Background

Equipment used to produce this document is a Heath 386 computer with 1 Meg of memory, a 40 Meg hard disk, and a color monitor; software is DOS 3.21 and WordPerfect version 5.0. This is being typed with left and right margins at default settings of 1 inch; justification on; default Tab settings are every 0.5 inch.

The term "word wrap" is commonly used to mean that when you are entering text using a word processor, and when the text reaches a certain predetermined (and user definable) area of the screen on the right, the word processor *automatically* returns to the left margin of the screen with the screen cursor on the next line - you don't have to "return" or press "Enter" yourself (unless of course you desire to do so.) WordPerfect's default is with "word wrap" enabled, and every time it "wraps" it inserts an "invisible" "Soft Return" into the text at that point at the end of the line. If you press the **Enter** key, you return the cursor to the left margin of the next line and you are inserting an "invisible" "Hard Return" in the document at that point. Ordinarily, one only presses



the **Enter** key to end a paragraph or to add blank space between paragraphs.

### Illustrations

These indented lines are preceded by one **F4 ->Indent** code before the T in the word 'These'. Word wrap is taking care of the right margin and no Hard Returns have been put in by me (using the **Enter** key) until after the next period.

The **Enter** key was pressed just after the period at the end of the last sentence which cancelled the effect of the one initial **F4 Indent->** code that caused the left indentation of all lines just above. Note that the indented lines above could also have been produced by typing 5 spaces before each line, or by typing 1 Tab (set for 5 spaces) before each line — the **hard** ways! Now, the **Enter** key will be pressed two times, for spacing purposes.

One **Shift-F4 ->Indent<-** code was inserted just before the O in the word 'One' at the beginning of this paragraph. Note that both the left and right edges of all lines in this paragraph are indented by this single code. Note that no **Enter** codes (Hard Returns) have been used in this paragraph until after the next period now.

The **Enter** key was pressed just after the period at the end of last sentence and that cancelled the effect of the one **Shift-F4 ->Indent<-** code that caused indentation of both edges of the preceding paragraph. Now, for spacing purposes, the **Enter** key will be pressed two times.

1. This line was typed with a 1, then a ., then an **F4 ->Indent** code and word wrap has automatically brought the other lines being typed to the new indented left edge. The single **F4** code has caused this. Now the **Enter** key will be pressed two times.

1. This is similar to the previous paragraph except that instead of an **F4 ->Indent** code, a **Shift F4 ->Indent<-** code was used. This single code has caused indented left and right edges. Pressing the **Enter** key will put us back to the default margins.

The **Enter** key was pressed just after the period at the end of the last sentence (following the word 'margins') and this caused a **Hard Return** code to be inserted. We'll now press **Enter** key two times, for spacing purposes.

The above illustrations should help explain the differences between the two unique **Indent** keys, **F4** and **Shift-F4** that WordPerfect uses. **F4** produces an indent only on the left while **Shift-F4** creates concurrent indents of left and right sides.

You can also use the **F4 ->Indent** code to produce the following type of outline code.

I. The **Enter** key was pressed after the period in the last sentence, then the following were typed - I, ., **F4, The...** etc.

A. The **Enter** key was pressed after the period in the last sentence, then the following were typed - **F4, A, ., F4, The...** etc.

1. The **Enter** key was pressed after the period in the last sentence, then the following were typed - **F4, F4, 1, ., F4, The...** and so on. Note that once a level of indentation has been established, you just keep typing and when you approach the right margin, word wrap takes you to the next line at the same level of indentation until you press the **Enter** key as will be done after the next period.

And here we are back at the defaults once again. Now two **Enters** for spacing purposes.

**Shift-F4 ->Indent<-** codes could have been used in the above illustration and would have produced a different format, indenting both the left and right edges of the material as depicted in the following examples.

1. This is a paragraph produced by typing a 1, then a ., and then a **Shift-F4 ->Indent<-**. That single code indented the paragraph to the first 0.5 inch factory default tab setting on the left and typing was just continued to produce the effect seen. The same single code also indented the right paragraph edge by 0.5 inch.

2. After the period at the end of the last paragraph, a **Hard Return (Enter)** was typed. That returned the cursor to the default left margin, and then a 2, a . and another **Shift-F4** were typed and typing just continued as above, letting word wrap produce the effect seen. No tabs were entered, just the **Shift-F4 (->Indent<-)** code was typed as noted. Now I type another **Hard Return (Enter)**.

3. Same start with this paragraph and we shall produce another level of indentation.

A. After a **Hard Return (Enter)** following the period in the last sentence, we wound up at the left margin and did 2 **Shift-F4s**, typed an A, then a . and then another **Shift-F4**. Then just type and let word wrap do its thing. Note that this produces an equal indentation on each side. The preceding 2

paragraphs are also indented from both left and right sides. Now we will do one **Hard Return (Enter)**.

And, a single **Hard Return** cancels the indents and brings us to the default left margin where we shall continue typing and letting word wrap bring us back to the left margin. After the next **Hard Return** (produced by pressing **Enter** key 1 time) located just after the next period, I will use both **Outline** and **->Indent<-s**.

I. Here, after . at the end of the last paragraph, we did 1 **Hard Return** (pressed **Enter** key), then pressed **Shift-F5** (the **Outline** key) and selected 4 from its menu and the Outline appeared in red at the lower left of the screen. Next another **Hard Return** was inserted (pressed **Enter** key) and 'I.' automatically appeared and I typed an **->Indent<- (Shift F4)** and have continued typing. The right margin was reset by the program. Now I'll press **Enter** key.

II. The 'II.' was inserted and displayed by WordPerfect. Then I put in a Space and then a Tab (Pressed **Space Bar**, then pressed **Tab** key). That cleared both indented margins as can now be seen. I will do another **Enter (Hard Return)** but followed by an **->Indent<-** now.

III. Back to the previous type of format. An **Enter**

A. produced a IV. at the left margin, then a **Tab** automatically deleted the IV., and printed the A and its following period. I then typed an **->Indent<- (Shift-F4)** and it produced a new constant level of indentation, the first word of which was 'produced'. Note that every time I have done an **Enter** since turning on **Outline**, the program has automatically put in the "Paragraph Number" (or Paragraph Letter). Now I will turn **Outline** off by typing **Shift-F5** and then after typing a 4, it will be off after I press **Enter**.

I just pressed **Shift-F5**, typed 4, pressed **Enter** and the Outline demonstration is almost over. I will now press the **Enter** key two times for spacing purposes.

One last type of indenting is known as a "hanging indent". This paragraph was produced in WordPerfect by typing an **Indent** code immediately followed by a "Margin Release Code" (**Shift-Tab** in Word Perfect). In this paragraph, for example, just before the first word 'One', an **F4 ->Indent** code was typed and then I immediately typed a **Shift-Tab**,

followed by One and the remainder of this paragraph. This also works with the WordPerfect **Shift-F4** code and produces a "hanging indent" format with equal indentations from left and right sides. This type of formatting is less commonly seen than the other types of indenting.

### Conclusions

This has demonstrated how **Outline** and **Indent** might work to the user's advantage with Outline text and/or in Outline mode using WordPerfect Version 4.2 and 5.0. I know from experience that using indents rather than using spaces or tabs when it is desired to format text in an outline form certainly speeds up your editing, whether the editing is one of deletion or one of insertion of text.

Some specific WordPerfect codes used have been **HR** (Hard Return) produced by pressing the Enter key, **SR** (Soft Return) which the program routinely puts at the end of a line when it "word wraps", **->Indent** produced by pressing F4 function key, **->Indent<-** produced by pressing Shift-F4, **Outline** produced by pressing Shift-F5, and **<-Mar Rel** produced by pressing Shift-Tab. The five codes mentioned in this paragraph are all the codes needed to create outlined documents using WordPerfect (I didn't count the **SR** (Soft Return) because this code is auto-

matically inserted by WordPerfect when necessary.)

### Advantages of Indent Codes

The main advantages of creating *outlined text* using indent codes (rather than with spaces and/or tabs) are the ease of using the indent codes, the fewer total codes used, and the ease of document revision.

### Errata

Other codes used in this presentation have been **Enter**, **Bold** and **Italic**.

WordPerfect normally presents you with a whole screen to work with. However, if you do a "**Reveal Codes**" by pressing Alt-F3, you are presented with a split screen. The upper half displays normal appearing text material, while the lower half displays the same text material **plus** any embedded codes that you (or WordPerfect) has placed in the document. The cursor is displayed in both halves of the screen, and you can scroll up or down, examining text and/or codes, inserting or deleting codes if you so desire. Pressing Alt-F3 again returns you to the usual screen display.

These last few paragraphs are presented in a "hanging indent" style. Please note that I used *no* Space codes (the only spaces used separated words and sentences) and *no* Tab codes for "indenting purposes" anywhere in this manu-

script.

This meant that it was easy to insert, delete, or alter text material, anywhere that I desired. The original text was approximately 400 words, and the final text totaled over 2700 words. Obviously much material was inserted. However, it was inserted within original WordPerfect factory default limits, or within previously defined levels of indentation. Therefore no "Indenting" Spaces, Tabs, or additional Indent codes were needed.

I kept track of the number of separate editing sessions that took place during this text preparation and these totaled over 40! But, without Space or Tab codes, it was easy to do. If the 'Outlining' of the original 400 words had used Spaces and Tabs for indenting purposes, this would never have been written!

Although I prefer a "ragged right margin" style, I intentionally have used WordPerfect's default, a right justified (even right margin) style for this essay since this more clearly defines any right indentations that were used. Three keystrokes could immediately change the entire document from the right justified default to a ragged right margin.

Enjoy!



## When Your Company Needs the Best in Local Area Networking Services ...Go for the Gold!



Authorized  
This Novell Gold Reseller has achieved the highest level of authorization and is recognized as a full partner with Novell.

NOVELL



Call or write for First Capitol Computer's unique new Disk-Based Catalog, FREE for the asking!  
...Or download from our 24-hour BBS system at 1-314-928-9228

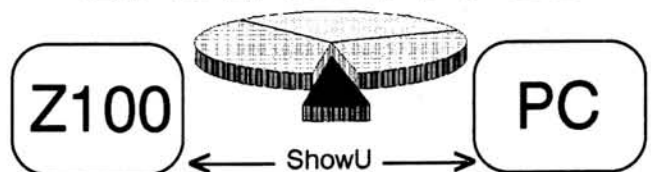
When you want the best in networking design, service, and support - call First Capitol Computer. First Capitol is a Novell Gold Reseller, the *highest level of authorization* from the industry's #1 LAN manufacturer. Our experienced staff can provide assistance from initial design through on-site installation and training, as we have for more than five years - for companies from small law firms through Fortune 500 monsters.

Sure, we'll save you money on the components in the system, but more importantly, when we're done, the network will work like a dream - not like a nightmare! Experience shows!

First Capitol Computer  
#16 Algana Drive  
St. Peters, MO 63376  
Order Line: 1-800-TO-BUY-IT  
Other Calls: 1-314-447-8697

Reader Service #191

## SHARE A PIE



Now Z100 and PC users can share graphics.

ShowU is the capture and display software you've been waiting for. ShowU lets you capture any PC graphics screen and display it on a Z100 or capture any Z100 screen and display it on a PC. ShowU will capture Z100, CGA, Hercules, EGA, and VGA displays and allow display on either computer.

Create your own personal slide show, mix and match screens from different displays, and show them on a Z100 or PC.

ShowU gives you graphics compatibility between computers. ShowU also allows you to use clip art designed for ShowOff or desktop publishing software on either computer.

ShowU is another quality product from the developers of ShowOff, the high resolution graphic editor for the Z100, and ZIP Image Processing software for the Z100 and PC-compatible computers. ShowU \$34. ShowOff \$95. ZIP \$79.

Hogware Company  
470 Belleview St Louis MO 63119 (314) 962-7833

ShowU requires MS-DOS 2.0, and Z100 with full video ram. or pc with graphics display. To order call (314) 962-7833 VISA/ MC accepted Mail orders add \$2 s/h, in Missouri add 6%.

Reader Service #108



# BUG ZAPPING

**Problem:** The ZKB-2 keyboard cursor keys produce digits when running GW-BASIC, SIDEKICK, or other stay-resident programs.

**Solution:** This problem was caused by the keyboard sending to the host too rapidly. The firmware was changed in the keyboard ROM so a short delay was inserted each time a byte was sent to the host. This delay allows the system enough time to process the key bytes. To correct this problem, change U1 from a ZKBD2, Version 1.2 (444-582-1) keyboard ROM to a Version 1.4 (444-582-2).

**Problem:** MS-DOS PREP hangs when using third-party hard disk drives with #150-330-C1 Western Digital Controllers.

**Solution:** When using an **unformatted** third-party hard disk drive it may be necessary to format the drive, using a low-level format routine before running PREP. The routine WDFMT.EXE can be downloaded from the ZDS Bulletin Board (616) 982-3503. A document file (WDFMT.DOC) that explains the use of the routine can also be downloaded. Install a third-party hard drive by first running WDFMT.EXE and then run PREP in the normal way. WDFMT.EXE formats all cylinders for head 0 through 7 first, and then the cylinders for head 8 and above. Be aware that WDFMT.EXE erases all data that's on the drive!

**Problem:** Configuring the ZDH-1117-B0 Computer to work with the ZVM-1360.

**Solution:** On the PS-7235 video card, set the COLOR/MONO switch to COLOR. On the system (CPU) board, set the switch SW101-4 to ON, and SW101-5 to OFF. Adjust the HORIZONTAL CENTER and VERTICAL CENTER controls on the back of the ZVM-1360 Monitor to center the display.

**Problem:** The ZWL-200-2/4 locks up when using MS-DOS expanded memory driver version 1.4.

When using the expanded memory driver version 1.4 that's supplied with MS-DOS version 3.21, and BIOS version 3.34, the system will lock up. When using EM drivers version 1.6 supplied with MS-DOS version 3.21, and BIOS version 3.37, the following ERROR messages are displayed: "ERROR: Board 0 at I/O base x258 is not responding; ERROR: No EMS boards responding. Driver not installed."

**Solution:** To correct both of these problems, use MS-DOS version 3.21 and BIOS version 3.39. This version has the correct EMM.SYS driver version 1.08. The version 1.08 driver is also available from Zenith Tech Support Bulletin Board (616) 982-3503.

**Problem:** The Z-286 system locks up after reset when an IBM or PARADISE VGA card is installed.

A MicroSoft MOUSE or MicroSoft WORD won't work properly.

Lack of support for ESDI and xT type Winchester Controllers.

**Solution:** On the CPU/Memory board, change the monitor ROMs to the latest versions to correct these, and other problems.

U216 should be Version 2.5 (#444-423-10)

U217 should be Version 2.5 (#444-424-10)

**Problem:** Parity errors occur on the Z-315 memory board.

**Solution:** On the #181-6984 memory board, noise on the RAS2 signal causes parity errors. To correct this problem, two capacitors are installed on the foil side of the board, as follows:

0.1 uF capacitor (#21-786) between U429-2 and the ground lead of C432.

56 pF capacitor (#21-804) between U429-7 and U429-10.

**Problem:** Unable to run MicroSoft Windows/386 with only 1MB of memory.

**Solution:** To enable Windows/386 to run with only 1MB of memory installed, install a #444-670 PAL at U470 on the #181-7045 memory board. Set SW401, section 2, to the OFF position. This PAL change allows an extra 256k of memory to be used as extended memory.

Also, when running Windows/386 and a Z-449 video board is installed, set section 5 of the DIP switch on the video board to OFF. **CAUTION:** Incorrect switch settings on the DIP switch can damage the video card and the monitor.

**Problem:** I have a ZDE 159 that came with an Enhanced Graphics Adapter video card. I set the EGA card to run in CGA mode with my CGA monitor. Some software that I installed for CGA video works fine, but other packages (for example, Lotus 1A) do not run correctly when installed for CGA video. How can I make all my software packages run in CGA mode on the ZDE 159?

**Solution:** Some software packages look directly at the type of video card installed in the computer. These packages will see the EGA card, and pay no attention to how the switches are set. Therefore, they think that your computer has EGA video, and will not run correctly when the software is installed for CGA video. To correct this conflict, leave your computer and video board set in CGA mode, but install the software for EGA video.

There are EGA video drivers available for Lotus 1A (PC version only). They can be downloaded from

Compuserve's "World of Lotus" section, or obtained directly from Lotus Corporation.

**Problem:** How do I tell the difference between the different types of EMS memory boards available for the Z-159 computer system?

**Solution:** Two EMS upgrade kits are available for the Z-159 series computers. The major difference between these two kits is the amount of memory included with the kit. The ZA-315-1 kit comes with 512K of memory; the ZA-315-2 has 256K. The kits can be used with any Z-100 PC series computer, but they will not work in AT- or 386-level machines.

Note that all EMS boards in the Z-159 must have the same page frame address. The Z-159 and the 315 upgrade kits are shipped with the memory board page frame address set for D000 to DFFFF. If the address is different on the second EMS board, the following message will appear when loading the EMM.SYS driver:

ERROR: Second EMS board not responding, only one EMS board with XXXk of memory installed.

**Problem:** Is it possible to install 1.2 meg drives on my Z-159 series computers?

**Solution:** The Z-159 series computers do not support 1.2 Mb floppy drives. Although the floppy controller supports the 1.2 Mb format, the ROM does not. 1.2 Mb floppy drives are not an IBM XT compatible feature. Therefore, to maintain IBM XT compatibility, the Z-159 computers do not support them.

**Problem:** KEYBxxx command on Z-159 causes system to hang.

**Solution:** Executing the KEYBxxx command on a Z-159 series computer with ROM version below 2.9E causes the system to hang. To correct the problem, set DIP 3 of SW101 on the 159's CPU board to ON. ROM versions 2.9E and above (part number 444-494-x) also correct this problem.

**Problem:** I have added a 720K 3-1/2" disk drive to my Zenith system. After booting the system, the first disk that I place in the 3-1/2" drive is read properly; however, the system seems unable to tell when/if that disk is replaced. If I put another disk in the drive and ask for a directory — DIR — the directory of the original disk is displayed. How can I correct this situation.

**Solution:** Some 720K 3-1/2" disk drives do not provide 'change line' support. This means that the drive does not support a line that the system can test to determine whether the drive's door has been opened since the last time that it was accessed. By default, MS-DOS expects this 'change line' support for 720K 3-1/2" disk drives. However, MS-DOS also provides a means of overriding the default 'block device' (disk drives) driver settings, which should allow you to use the drive.

If the drive in question is drive A>, create or edit a file named CONFIG.SYS to contain the following line of text:

```
DRIVPARM = /d:0 /f:2 /h:2 /s:9 /t:80
```

If the drive in question is drive B>, create or edit CONFIG.SYS to contain the following line of text:

```
DRIVPARM = /d:1 /f:2 /h:2 /s:9 /t:80
```

NOTE: The file CONFIG.SYS must reside in the root directory of the drive that you use to boot your system. Your MS-DOS manual explains how to create

and use CONFIG.SYS. It documents CONFIG.SYS commands, such as DRIVPARM (drive parameters).

MS-DOS version 3.2 or higher is required for 3-1/2" drive support.

**Problem:** I still don't understand the differences between "conventional", "expanded", and "extended" memory.

**Solution:** I. Conventional Memory:

The 8088 microprocessor used in a PC (Personal Computer) machine can directly address a maximum 1,048,576 bytes of Memory, or 1,024K, or 1 Meg. This is due to the physical architecture of the PC machine. The memory buses are 20 bits wide. To calculate the full range of memory locations, we raise 2 to the power of 20, and thus, get the number: 1,048,576 bytes.

Memory Map:

```
F: 256K |
E: reserve |
D: |
C: |
B: 128K |
A: reserve |
9: |
8: 640 K |
7: |
6: Convent. |
5: memory |
4: |
3: |
2: |
1: |
0: |
```

After subtracting 256K, and 128K of memory from the addressable 1 Meg of memory, the user is left with 640K of memory (RAM — Random Access Memory). This is now known as Conventional Memory.

This 640K of conventional memory is used by the Operating System, resident programs, and any program and data currently in use.

MS-DOS was written to accommodate only this 640K of user memory.

II. Extended Memory

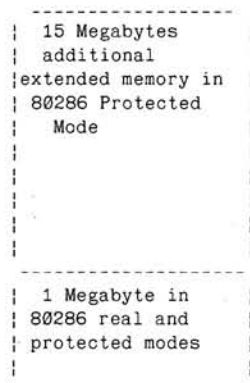
While PCs satisfied users for a while, the demand for a faster, more powerful machine increased. Thus, the ATs were developed utilizing new computer architecture and technology. The AT (Advanced Technology) machines have an 80286 microprocessor. This microprocessor can directly address up to 16 times as much memory as the 8088 microprocessor — a full 16 Megabytes.

The 80286 can run in two modes — a real mode and a protected mode. In the real mode, at boot-up time, the AT is similar to a PC machine. With a few assembly language instructions, the AT can be put in protected mode. In this mode, an additional 15 megabytes of memory can be addressed because the AT uses a 24-bit address bus instead of a 20-bit



bus. 2 raised to the power of 24 is 16 meg (the 1 meg of conventional and 15 meg of protected memory).

Here is a diagram of an AT memory map:



Only the Xenix Operating System can utilize the full 15 Meg of Extended Memory. Because MS-DOS was written for the 640K conventional or 1 Meg (addressable) memory, none of this extended memory can be directly used with the current versions of MS-DOS. MS-DOS v. 3.1 and 3.2 do, however, allow the user to set up a ram disk.

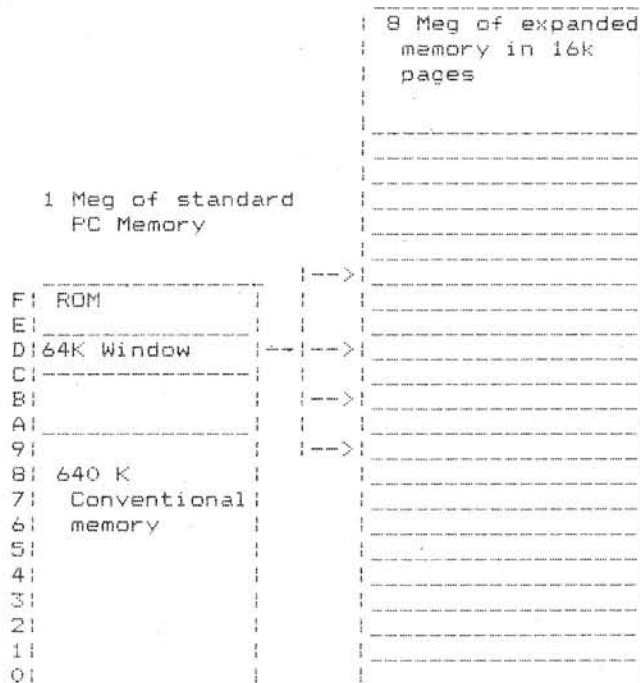
VDISK or RAMDISK are utilities provided with MS-DOS that allows the user to set up a ram disk. The /E switch must be used for the VDISK to load into the Extended Memory available (over 1 meg). By default, VDISK will load into the Conventional Memory (640K and below).

### III. Expanded Memory

PC users felt the effect of the 640K barrier stifling, and clamored for more memory. Primarily due to the fact that large spreadsheets in LOTUS 1-2-3 would give a "memory full error", LOTUS decided that something needed to be done to increase PC memory. LOTUS, INTEL, and MICROSOFT developed a type of memory manager named Lotus/Intel/Microsoft Expanded Memory Specification.

Expanded Memory uses a memory manager program called the Expanded Memory Manager (EMM). This is set up like any device driver in the CONFIG.SYS file. At boot-up time, the EMM searches for 64K of contiguous memory. After it finds a free area, the board will be addressed from this 64K segment of conventional memory.

Expanded Memory can add up to 8 megabytes of RAM to a PC or AT system. After the EMM finds the free 64K in high memory, this 8 megabytes will be addressed through this 64K space. The technique of fitting 8 megabytes into 64K is called bank-switching or paging. The 64K is used as a window to address 16K pages.



Expanded Memory is very useful; however, programs have to be written to request from the EMM memory in a page size of 16K. The following is a list of Software Packages that Zenith is currently selling written to use Expanded Memory: Windows, Supercalc, and Lotus 1-2-3.

VDISK or RAMDISK in MS-DOS 3.2 has a switch /A that will use the Expanded Memory.

ZSPOOL in MS-DOS 3.2 has switch /A that will load the print buffer in Expanded Memory rather than Conventional Memory.

### IV. Definitions

**Conventional Memory** — The 1 Meg addressable memory or 640K RAM found in a PC and an AT in real mode.

**Extended Memory** — The 15 Meg additional memory that can be addressed in an AT in protected mode. Xenix uses this memory as regular RAM, while MS-DOS v. 3.x can only use it as extended with the use of a VDISK or RAMDISK.

**Expanded Memory** — A potential 8 meg of additional memory that can be addressed with the use of a memory manager, EMM. It can be used in a PC or AT machine by application programs written specifically to use Expanded Memory.

**Editor's Note:** As further service to our readers, I'd like to keep this column in REMark on somewhat of a regular basis. In it, we'll hopefully be able to keep you abreast on ROM versions, simple fixes for known problems, etc. If you have a fix that can benefit other HUGGIES, send in the Problem/Solution on a post card, along with a request for ANY single piece of HUG Library software, and, if we use it here in the magazine, we'll send your requested software product to you FREE!





tion you all may now be asking yourselves. I'll show you that step right now.

You now make use of another DOS utility you all have called DEBUG. This one you have to be a little careful with, but if you follow my steps you'll have no problems.

You start by typing DEBUG GOOD.DAY at your DOS prompt. After a few seconds the DEBUG prompt (-) should have appeared on the screen. At this prompt enter D and return. Your screen should have eight lines of information now displayed on it. The first line should look something like this:

```
XXXX:0100 20 20 20 20 20 20 20 20-20 20
```

You will remember when we first started entering our good-day text I told you to enter ten (10) spaces and return on the first line. The reason will now be explained using the above example. The spaces in text appear above as the HEX number 20. If you entered ten spaces at the start of your text you should have ten HEX number 20's like the example above. The end of the line has several 0D, 0A combinations. The 0D is the HEX number for a carriage return, and the 0A is the HEX number for a line feed. These are entered in your text every time you press return at the end of a line of text. We, at this point, are only going to make changes to the spaces we entered by entering E 0100 return.

You should now see XXXX.0100 20. on your screen. The four X's represent a location in memory where DEBUG has put your file it's working on. Yours might be a number and letter in combination but don't worry about understanding this now. The .0100 is the first byte location in memory of your text and 20. is the information in that byte (20) to be changed. The E 0100 command you entered to get here told DEBUG to go into EDIT mode starting with the byte at location 0100 in memory. I know, "Enough already lets make the change.". The cursor should now be located right after the period next to the number 20. You now enter 1, B, and press the space bar, (NOT RETURN THIS TIME). If you did hit return, re-enter E 0100 to start over, and press the space bar. If all went correctly you should see the following line on your screen: XXXX:0100 20.1B 20. The 1B you entered is the HEX number for the ASCII ESCape code. The period is used to separate the before and after byte information so you can see where changes have been made. Now enter 4, 5, and press the space bar. This is the HEX number for the ASCII character 'E' you have entered. Using the above steps, enter 1B, 46, press return, and you've made the changes. If you now enter D 0100 return, at the prompt you should see this for the first line:

```
XXXX:0100 1B 45 1B 46 20 20 20 20-20 20 0A 0D 0A 0D 0A 0D .E.F .....
```

"OK, I have the above information, but what have I done?" What you have

done is this. The bytes 1B 45 is the ASCII ESCape command and E added to the ESCape command is the ASCII code ESC+E, which to a Z-100 means to clear your screen. The bytes 1B 46 is the ASCII ESCape command and F added to the ESCape command is the ASCII code ESC+F which means enter graphics mode. You have now told your terminal to clear its screen and enter its graphics mode to display the graphic representation of the text file you entered. This will happen whenever you tell it to TYPE GOOD.DAY.

"WOW! That's great, now I'm all done!" NOT QUITE. You still have a

couple of things to do yet. 1) You have not saved the above changes to the disk file yet, and 2) You entered graphics mode but you have not returned to the normal text mode yet. You could save this file right now and type it to display it on the screen, and it would work, but you're still in graphics mode and everything that has a graphic character for it will be displayed as a graphic. This can lead to some rather interesting but unintelligible displays on your screen.

"SO, now what do I do?" Well, do you remember those four spaces that you entered on the last line of the text you keyed in?

That's right, you're going to edit those spaces to enter the escape codes needed to exit the graphics mode. You enter D 0200 at the prompt (-) to display the last part of your text file.

You will now have to look for a sequence of bytes like this:

```
5E 0D 0A 20 20 20 20 0D 0A 1A 00 00 00 00 etc.
```

The above sequence could be split anywhere and on two separate lines.

The 5E 0D 0A are the last three bytes of your graphic display. The four 20's are your four spaces followed by a carriage return and a line feed. The 1A is the end of file marker followed by 00 00 00 which are ASCII nulls or empty bytes of memory. You now have to read on the left side, the line of memory where the first 20 starts, and enter it as E XXXX. In my case the first 20 started on line XXXX:0230, so I had to enter the EDIT command as E 0230. Now press the space bar until the first 20. is displayed on your screen. You now enter 1B for the ESCape code, press the space bar and enter 47; this is the HEX number for G. You have entered now ESC+G, which is the ASCII command to exit graphic mode. Now press <return> to return to the DEBUG prompt. If you are sure everything is done correctly you can now save the changes to disk. You save these changes by entering 'W' return. This

tells DEBUG to Write your changes back to disk, overwriting any existing informa-

tion saved for that file. If you are NOT sure you are right and want to start over, enter 'Q'<return> at the prompt; this tells DEBUG that you want to Quit. If you do this before you enter 'W', NO changes are saved. If you do this after you write to disk, all changes are saved and you Quit DEBUG.

When the above steps have been completed, you can enter TYPE GOOD.DAY at your DOS prompt. The screen should clear and your graphic message should appear along with returning to your DOS prompt. You should now try typing some lower case letters if everything else worked ok, just to see if the display returned to text mode properly. If small letters appear, you're OK.

A chart showing all the graphic symbols and HEX codes can be found in your Z-100 Users Manual in section B. You can use the above procedure to develop menu displays also. Just remember that any writing you want displayed must be in upper case or it will be displayed as a graphic symbol.

I hope the information I have given you helps in some way, that you can use this to develop the menus you need for your particular situations.

HARD DISK MANAGEMENT with MS-DOS and PC-DOS  
Dan Gookin and Andy Townsend  
Copyright 1987  
TAB BOOKS Inc.  
P.O. Box 40  
Blue Ridge Summit, PA. 17214  
Book No. 2897 \$26.95  
ISBN 0-8306-0697-1



# MS-DOS, OS/2, AND THE NUMLOCK KEY

ROBERT A. METZ  
2000 M63  
BENTON HARBOR, MI 49022

I have been using the IBM PC-style keyboard on my Zenith machines for a number of years now, and have become quite accustomed to the arrangement of cursor control keys on the numeric keypad. I can safely say that I have never used that keypad for its numbers (once or twice experimentations don't count), but exclusively for cursor control. I now have a Z-386 with the new style 101-key keyboard. This keyboard contains separate cursor control keys and the same combination cursor/numeric keypad. Quite naturally, my right hand gravitates to the familiar layout of the keypad for cursor control. Only rarely do I use the separate cursor control keys, and then primarily "because they are there".

The older style 83-key PC keyboard's keypad defaulted to cursor keys and it took a manual depression of the NumLock key to transform it into a numeric keypad. This suited me just fine. The 101-key keyboard, however, defaults the keypad to the numeric state, using the logical assumption that since there are separate keys dedicated to cursor control function, the keypad cursor control is redundant. My personal mode of operation forced me to manually take the keypad out of NumLock after every system startup in order to obtain desired cursor control functionality. Understandably, I didn't always remember to do this. My habit of using an editor (BSE from Zenith's MS-DOS Programmer's Utility Pack) to browse files came to haunt me in these not-so-rare occasions as the first line of many files was liberally sprinkled with 2s and 3s before I realized that I wasn't really paging through the file like I wanted. I endured this irritation for many weeks before I finally got fed up.

Realizing my habits were unlikely to change, I needed to find a way to turn NumLock off automatically at system startup. All it took was a little program called from my AUTOEXEC.BAT which turned off the keyboard NumLock state bit in the BIOS data area. Within MS-DOS (IBM-compatible versions), the byte at offset 17h relative to the BIOS data area at segment 40h contains a set of keyboard status flags as follows:

80h - Insert state is active  
40h - CapsLock state is active  
20h - NumLock state is active  
10h - ScrollLock state is active

08h - Alt key is pressed  
04h - Control key is pressed  
02h - Left Shift key is pressed  
01h - Right Shift key is pressed

By changing values of these flags, a program can manipulate the keyboard 'lock' states as surely as it can be done manually by pressing the various 'lock' keys on the keyboard. The keyboard routines in the BIOS will even ensure that the LED lock indicators on the keyboard follow the lock state set by a program. I created the following program using DEBUG (italic text is entered through the keyboard, <CR> represents the Return key, XXXX represents don't-care displayed data, comments are informational only):

```
C>debug<CR> ; invoke 'debug'
-nnumlokof.com<CR> ; name the file
-a100<CR> ; start assembling
XXXX:0100 mov ax,40<CR> ; address the BIOS data
XXXX:0103 mov ds,ax<CR>
XXXX:0105 and byte ptr [17],df<CR> ; turn off numlok flag
XXXX:010A int 20<CR> ; exit to DOS
XXXX:010C <CR> ; stop assembling
-r cx<CR> ; modify CX register
CX XXXX
c<CR> ; set to 0ch
-w<CR> ; write the file
Writing 000C bytes
-q<CR> ; quit 'debug'
C>
```

Alternately, the program may be entered in source form as the file 'numlokof.asm', then assembled and linked into the appropriate 'numlokof.com' file:

```
NAME      NUMLOKOF
TITLE     Program to Turn Off
NumLock
PAGE      ,120

;
; Program to take the Keyboard out of NumLock
;
; Assemble and Link this file ('numlokof.asm'):
;
;      masm numlokof;
;      link numlokof;
;      exe2bin numlokof.exe numlokof.com
;      del numlokof.obj
;      del numlokof.exe
;
CODE       SEGMENT
ASSUME    ;,CS:CODE,DS:CODE
ORG       17H
FLAGS     LABEL    BYTE
ORG       100H
START:
MOV       AX,40H ; Address BIOS Data
MOV       DS,AX
```

```
AND       FLAGS,0DFH ; NumLock off
INT       20H ; Exit to DOS
CODE      ENDS
END        START
```

The resulting 4-instruction, 12-byte program ('numlokof.com' in the current directory), when executed, forces NumLock state off so the keypad has cursor control function.

After placing the 'numlokof' command in my AUTOEXEC.BAT file, my problems with NumLock vanished and I promptly forgot all about it - that is until I installed and started using OS/2. While 'numlokof' works as it always did in OS/2's DOS compatibility box, it is a miserable failure in Protected Mode OS/2. I

again endured being bitten many times over by NumLock while using Microsoft's Protected Mode Editor SDKED (Software Development Kit Editor) and later MEP (Microsoft Editor Protect mode).

I needed a Protect Mode command to turn off NumLock. Once I had overcome the OS/2 development learning curve, this proved to be a relatively simple task. The following is a C program making



use of only two OS/2 Application Program Interface (API) calls which, when executed, will force the keyboard out of NumLock state:

```
/*
OS/2 Protected Mode Program to turn off the
Keyboard NumLock state. Source file is 'numlokof.c'.

Compile and link using:
    cl numlokof.c

Create 'bound'.exe file if desired using:
    bind numlokof doscalls.lib

*/
#define INCL_SUB          /* include KBD subsystem call info */
#include <os2.h>          /* OS/2 API definitions */
main()
{
    KBDINFO kbstKbdInfo;    /* Keyboard Status Structure */

    kbstKbdInfo.cb = sizeof(kbstKbdInfo);
    KbdGetStatus(&kbstKbdInfo, 0); /* Get the current status */
    kbstKbdInfo.fsMask |= 0x10; /* modify fsState please */
    kbstKbdInfo.fsState &= 0xffdf; /* force off numlock */
    KbdSetStatus(&kbstKbdInfo, 0); /* Set the new status */
}
```

This program was written using Microsoft C version 5.1 and using Microsoft's OS/2 Development Kit version 1.00. Probably one of the first things noticeable about the above program are the long names for identifiers. Long, mixed-case names are a fact of life with development under OS/2. The "os2.h" header file provided with the OS/2 Software Development Kit contains definitions for all of the OS/2 API function calls and for the support data structures used by those calls. Explaining the contents of this header file and the conventions it establishes is really a subject best left for another article, or better yet, a whole series of articles. I can heartily recommend Charles Petzold's "Environments" column which has been featured in *PC Magazine* for the past several months. Briefly, for purposes of explanation of the above C program, the following should be said.

Two OS/2 API function calls are used:  
 KbdGetStatus - Get keyboard status  
 KbdSetStatus - Set keyboard status

Both functions work with a keyboard status structure defined as type "KBDINFO" in the "os2.h" header file. Without going into a lot of detail (several pages of the Microsoft OS/2 Programmer's Reference Manual are dedicated to an explanation of items in this structure), this structure contains several short integers, one of which (fsState) contains in its least-significant byte, flag bits identical to those in the MS-DOS BIOS keyboard status byte described above. Program logic is therefore simply to get the current keyboard status structure using the "KbdGetStatus" API function, turn off the Numlock bit in the "fsState" member

of that structure, and then to set that new keyboard status using the "KbdSetStatus" API function call.

Each OS/2 screen group carries its own set of keyboard states, each of which defaults to the NumLock-on state. To en-

sure that the Numlock state of the keyboard is off automatically, the above program would have to be invoked during initialization of each screen group. This is usually done by placing the 'numlokof' command in the "OS2INIT.CMD" batch file which is invoked each time the OS/2 protect mode shell, "CMD.EXE", is started. The OS/2 Program Selector, invoked using the Control/Esc keyboard combination, also contains its own logical keyboard which defaults to the Numlock state. The only way to disable this state within the Program Selector is to physically press the Numlock key. Since the Program Selector doesn't pay any attention to the number keys anyway, this condition is only slightly irritating. When the cursor control keys are used on the numeric keypad without having first taken the keyboard out of Numlock, no damage is done, but the desired action does not occur.

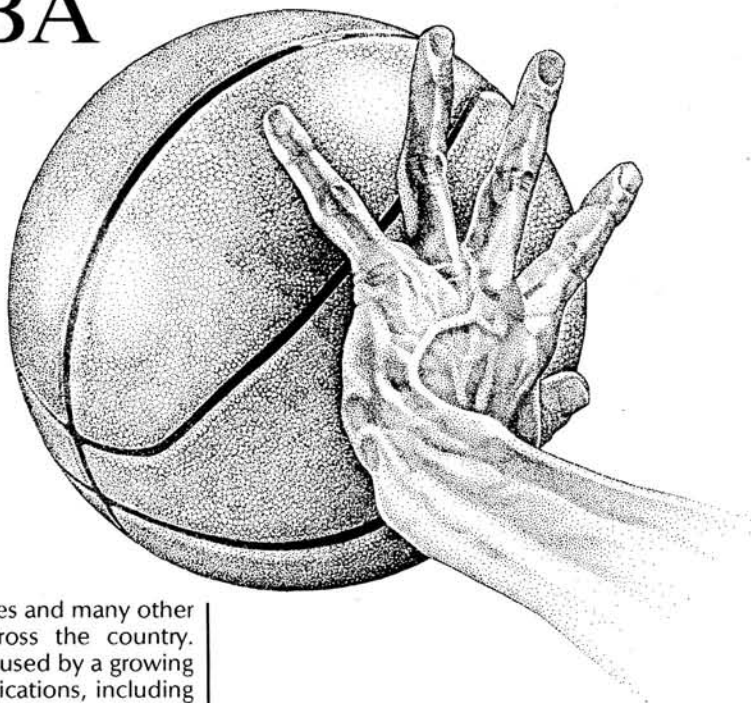
When the above program is compiled and linked with the OS/2 protect mode versions of the C runtime, the resulting ".EXE" file is only 2763 bytes long. While considerably larger than the 12-byte DOS ".COM" file above, it is nevertheless, comparatively short for an executable program compiled with a High-Level language such as C. A major factor in its compactness is OS/2's use of Dynamic Links. The two OS/2 API functions referenced in the program are themselves not linked as part of the program ".EXE" file, but are linked with the program when it is loaded into memory to be run. This makes for smaller program files on disk and for more efficient memory utilization at run time as code in the DynLink modules is sharable among several programs.

We now have two separate and distinct programs to accomplish the same task: the 12-byte 'NUMLOKOF.COM' for use in MS-DOS, and the 2763-byte 'NUMLOKOF.EXE' for use in OS/2 protect mode. It is possible to create a single program file which is usable in either MS-DOS or OS/2 protect mode using standard utilities which are part of the OS/2 Development Kit. The two API functions used in the OS/2 version of the program are members of the "Family API". This "family" of API functions is a subset of the complete OS/2 API set which has a common function between OS/2 and MS-DOS. Programs using only functions in the Family API may be linked in a special way, creating what are known as Bound Executables. Using the 'BIND' utility from the OS/2 Development Kit, the 'NUMLOKOF.EXE' file created by the Linker from the above compiled source code can be turned into a Bound Executable file of the same name. The former 2763-byte .EXE file is transformed into a 12,491-byte .EXE file which is equally functional in MS-DOS, the DOS-compatibility box of OS/2, or OS/2 protect mode. The extra material added to the .EXE file by the BIND utility is primarily the code for the MS-DOS implementation of the OS/2 API functions used by the program. When the program is run in OS/2 protect mode, the extra code is simply discarded by the loader. By paying the price of extra space requirements on disk for the bound .EXE file, a single program file is usable in multiple environments. Many of the DOS utilities provided with OS/2 itself are, in fact, bound .EXE files usable in both OS/2 protect mode and OS/2 real mode.

I have used a simple and somewhat useful utility as an example of the different programming environment in OS/2 as opposed to MS-DOS. Program development under OS/2 is no more difficult than under MS-DOS; it is simply different. Very strong arguments can even be made that development under OS/2 is much easier than under MS-DOS. Take the above example. The MS-DOS 'NUMLOKOF.COM' program used a programming 'trick'. The procedure used is not documented anywhere that I am aware of and is certainly not a capability inherent within MS-DOS. I knew of the specific flag byte through examination of BIOS source listings in the IBM PC Technical Reference Manual. Furthermore, experimentation was required to verify that program manipulation of that status byte had a lasting effect. On the other hand, the OS/2 version made exclusive use of standard, well-documented API function calls. \*

# SuperSporting and XyWriting Through the NBA

Glenn Nelson  
5238 57th Avenue South  
Seattle, WA 98118



An overflow crowd in the ballroom of a downtown Seattle hotel is buzzing with anticipation. It is draft day in the National Basketball Association, and the pieces of a suspected but unannounced trade are falling into place. The complex deal, rife with preconditions, will involve three NBA teams, including the local SuperSonics.

For myself and a Seattle Times sports columnist, the clock is ticking. We have about 15 minutes to make the next deadline of our afternoon newspaper. Based on previous research and the mounting circumstantial evidence, we decide the trade is going down. We will attempt to get details into the day's editions.

Both of us need background material and career statistics on a forward, Michael Cage, the Sonics will acquire in the deal. I've already got XyWrite III Plus humming on my Zenith SuperSport Model 20. In a flash, I switch windows and call up Cage's bio, which I prepared the previous day.

Since the bright-blue screen on the SuperSport absolutely screams, our competitors can see what we're up to. As my fingers dance furiously on the SuperSport keyboard, I hear a couple reporters whispering about our already having Cage's stats at our fingertips. In a more competitive environment, I would have lowered the contrast on my SuperSport display. However, since all my competitors work for morning newspapers whose editions already are dead and delivered, I let them marvel.

Once I complete my story, I access my earlier-edition piece, pluck out a few paragraphs which are still pertinent and slap them on the end of my current file. Then I run XyWrite's spell-checking program to ensure "clean" copy. Now it's time to transmit.

In some cases, since I'm not using the Times' standard issue Tandy Model 200 computer and its resident wordprocessor, there could be some minor problems of compatibility with the file I just composed. Such is not the case with XyWrite III Plus. XyWrite first appeared in 1982 as an offshoot of ATEX, the operating system

used at the Seattle Times and many other major publications across the country. In fact, XyWrite itself is used by a growing number of similar publications, including several of the popular national computing magazines. Also, all XyWrite files are pure ASCII.

To be on the safe side, I use a customized version of XyWrite's STRIP.PRN (which I renamed TIMES.PRN) to strip out of my files any incompatible XyWrite formats. TIMES.PRN also inserts the necessary begin- and end-text codes. Then, I type the story to a file which is ready to transmit.

To do all of the above, I have to load TIMES.PRN, execute it, then use XyWrite's TYPEF command to write the new, transmittable file. Since XyWrite allows one to completely customize the keyboard, all this is accomplished with a single keystroke. Saving, storing and quitting XyWrite (which normally takes 14 keystrokes) also is done with a single keystroke.

(Note: To get the single-stroke action in XyWrite, I copied the IBM.KBD file to GLENN.KBD. To create the transmittable file, I assigned "BC,L,O,A,D,(space),T,I,M,E,S,,P,R,N,XC,BC,T,Y,P,E,F,(space),C,.,\,S,E,N,D,CL,CL,CL,CL,CL,CL,CL,CL," to Alt-F (Key 33 in ALT Table). The eight "cursor left" commands allow you to type in the current name of the file before executing. To get the save, store, quit key, I assigned "BC,S,A,XC,BC,S,T,XC,BC,Q,U,I,T,XC" to CTRL-F11 (Key 87 in CTRL Table). Of course, either of those can be assigned to any key which makes sense to you.

Now I must switch to a subdirectory containing ProComm, the telecommunications program I use to transmit. Then, I have to turn on my internal modem (I usually keep it sleeping to avoid sapping my battery). Finally, I must access ProComm. Again, I am prepared to save keystrokes and thus time, which is precious in my business.

I hit "P," which activates a batch file that will switch to drive P: (using the SUBST Dos command, I've assigned drive letters to the most commonly used subdirectories on my 20-megabyte hard disk). P.BAT also executes "mode modem on," to turn the modem on, and then accesses ProComm.exe, which I've renamed Pro.exe. Now I'm ready to file my story.

The columnist and I are sharing one phone line. This isn't a problem, either. I merely plug the line into the "line" jack on my SuperSport and transmit. Via a remote interface with the Times' mainframe system, I send my story at 2400 cps, call up the story to confirm that it arrived safely and send an electronic message to the sports copy desk, informing them of the story's arrival. All this, I've accomplished before my colleague was through revising his column.

(If we were in need of further background material, we could have accessed wire-service stories via the mainframe interface or previous stories



appearing in the Times via an electronic library.)

Now I'm off to interview the Sonics' coach and general manager, then listen to Cage's comments over a speaker phone. Afterward, I compose an entirely new story, and transmit it to the Times for later editions.

During the course of June 28, draft day in the NBA, Seattle Times readers saw three different stories in our various editions. The early edition contained a piece on the possibility of a trade. The next contained one on the trade going down. The last two had stories about the trade, and what it meant to the Sonics. The last three edition's stories were researched, written and transmitted in a matter of an hour and a half, using the SuperSport, XyWrite, ProComm and assorted batch files.

The five previous NBA drafts I covered were far more involved and time consuming. Before the SuperSport entered my life, I used a Kaypro II at home and a Radio Shack Model 100 or Tandy Model 200 (the industry standards) at remote sites. I'd compose my first-edition story on the Kaypro II, using the reliable but slow and relatively limited Perfect Writer. Then I'd transmit the story, using a 300-baud Hayes Smartmodem, and print it out for reference at the Sonics' draft headquarters.

On draft day, I also lugged with me file folders filled with biographies and statistics on prospective draftees. Then I'd have to completely rewrite stories, on the 100 or 200, for later editions.

Memory - or, the lack thereof, was my biggest problem. The Times-issue computers have 64 K of memory. The Kaypro II has two disk drives, each capable of storing 191 Ks. The SuperSport, with a 20-meg hard drive and a 720-K 3 1/2-inch drive, gives me nearly 45,000 times more memory than the Tandy and Kaypro combined. Plus, with 640 Ks of RAM, I can execute my programs, and access files, on a RAM-drive and further speed up the routine of producing stories on deadline.

One of the reasons I originally purchased the Kaypro II was its purported portability. It is portable, all right. But as soon as I discovered it was nearly impossible to stuff under an airline seat, nearly tugged my arm out of its socket while traversing various airports, and that I had to lug an external modem and cables with me, my Kaypro quickly became stationary. Plus, it was too big (and didn't run on batteries) to use on airplanes and at NBA arenas.

Luckily, times change.

Using two different computers further complicated matters. Now, I have my SuperSport connected to a color monitor, expanded IBM keyboard and dot matrix printer in my office at home. This

arrangement allows me to work on stories and store notes here. Then, when I have to cover events elsewhere in Seattle, or in some other NBA city, I just uncouple my peripherals and take all my work with me in the SuperSport.

To some, this may seem as mere unnecessary, but nice, conveniences. Most reporters, in fact, make do with the limited tools I once worked with. I did, for five years. But this is a highly competitive and stressful business, and any such shortcuts provide an edge.

For example, my SuperSport's first road trip was to Los Angeles for the opening games of 1988 NBA Finals between the Lakers and the Detroit Pistons. Before I left for L.A., I began composing my preview stories on the championship series. During the flight down, I completed the stories. After checking into my LAX hotel, I transmitted the pieces back to the Seattle Times. Then I was off to San Bernardino to research a feature on a hot minor-league prospect belonging to Seattle's baseball Mariners.

(Note: I've found that the SuperSport, because of its screen and otherwise high-tech profile, attracts a lot of attention and questions from fellow passengers and stewardesses on an airplane. If I'm pressed for time, I avoid the commotion by listening to a personal stereo. This is a somewhat rude but effective, and often necessary, solution. The music also puts me in a proper frame of mind for writing.)

(Also, the archaic hotel that provides in-room telephones without removable jacks is the bane of traveling journalists - or anyone remotely accessing bulletin boards or on-line services, for that matter. I guess such hotels fear guests will steal the telephones, though I can't imagine it. What's next? Chained-down towels and shampoo containers? Anyway, to deal with such situations, I carry a Blackjack with me (no, not the kind to pound the negligent hotel officials). The Blackjack, available from micropipherals in Redmond, Wash., screws into the receiver of the telephone's handset and allows direct-connect transmissions).

After Game 1 of the NBA Finals, I wrote and transmitted three stories. Then I stored them on a floppy for future reference. The next day, I transcribed tape-recorded interviews, wrote notes and began composing my baseball story. The day after that, I covered Game 2, wrote, transmitted and stored three more stories. In the past, I would have killed out my previous files to make room for the new ones. The baseball story, a major piece taking up considerable memory, would have required it.

A traveling reporter finds considerable down time on the road. After a while, you find you've already seen all the in-house movies. Often, fatigue dis-

courages sightseeing/shopping trips or delving into a novel. The SuperSport provided me a needed alternative. I stored my copy of Falcon, an F-16 flight simulator, on my hard drive and whiled away hours shooting down MIGs and bombing enemy headquarters. I think releasing some pent-up aggressions helped me better cope with the gaggle of reporters at the Fabulous Forum in Inglewood, site for Games 1 and 2 of the NBA Finals.

During the flight back to Seattle, I was too tired to continue working on my baseball feature. But as soon as I got home, I hooked my SuperSport back to my color monitor and expanded keyboard, finished the story and transmitted it to the Times in time to make my Sunday deadline. Writing at home was far more comfortable because of the setting and the external monitor. It also was slightly faster because the expanded keyboard allows me to take better advantage of the keys I reconfigured with XyWrite.

XyWrite III Plus deserves special mention, though I do not intend to present a full review here. I tried Word Perfect, Microsoft Word and the fine shareware program, PC-Write, before settling on XyWrite. Each has its selling points. But, as far as I'm concerned, XyWrite III Plus is by far the best wordprocessing program for the dedicated writer - that is, those who write stories and books and especially for whom time is of the essence. Obviously, many agree with me as a growing number of writers are making the switch.

I like XyWrite III Plus because it relies less on pop-up menus and assigns far fewer commands to the function keys (in fact, only 29) than the others. Most commands are executed at the command prompt, and the command structure is extremely logical. For some, it may take longer to learn, but once mastered, it zips along. It is a faster program - period. Cursor movement and text manipulation is rapid. Functions such as search and replace really rip. In addition to the command structure, the customizable keyboard and XyWrite's version of macros (save-gets) add even more speed.

XyWrite III Plus takes a little longer to load. That's mainly because it loads personal dictionaries, printer files and other customizing files when it's booted up. On occasion, I've saved a few seconds (it doesn't take THAT .MDUL/.MDNM/long to load) by taking some of the custom files out of the startup program (STARTUP.INT) and loading them when needed. Of course, I've also assigned the loading routine to a single key.

Some believe that programs such as WordPerfect and Word may be more desirable to those who print their work a lot, or need camera-ready copy. I can't really say if that's true. I have no complaints about anything I've printed with XyWrite

Continued on Page 64



# Using HADES to Fix Damaged Floppies

Pat Swayne  
HUG Software Engineer

**Note:** In this article, the terms "sector" and "cluster" are used. If you are not familiar with these terms, read "Powering Up: Important DOS Commands You Must Know" by William Adney, in the October 1988 issue of REMark. All of the "Powering Up" articles will soon be available in book form, and the one referenced above will be Chapter 5 in that book. Watch future REMark issues for availability.

I recently found another use for Jim Buszkiewicz's versatile HADES (HUG Absolute Disk Editing System) program (available from HUG as part no. 885-3040). I discovered this new use when a HUG member called asking about the old demo disk that used to be shipped with the Z-100 computer. I told him to just send me a disk, and I would DISKCOPY mine onto it. I new I had one in a box on the floor somewhere. When I found it and saw how dusty it was, I decided to run my DTEST program (available on HUG disk 885-3052) on it to see if it was still good. Sure enough, DTEST reported two bad sectors on the disk. The bad sectors were in an unused area of the disk, so I could still run it, but DISKCOPY does not care where the bad sectors are, and will not copy the disk.

If you have COPYII PC from Central Point Software, you can use that to copy a disk containing bad sectors (it's not just for copy protected software), but the copy will be an exact duplicate of the original, including the bad sectors. Since I did not think that it would be nice to send

back a disk with bad sectors, I decided to try to fix the demo disk, using a trick I had used back in the old 8-bit days (with UDUMP under HDOS). Obviously, I was successful (or you would not be reading this article), and here is the procedure for repairing floppy disks.

Before you attempt to repair a damaged floppy, you must be reasonably sure that the damage is magnetic and not physical. A disk can show no apparent damage, but if the magnetic particles on its surface are altered by stray magnetic fields or just from losing magnetism with age, the disk may test bad. With my procedure, you can probably fix a disk with such magnetic damage, but if your cat tested his claws on the disk, it probably cannot be fixed.

## Finding the Bad Sectors

In order to attempt a repair on a disk, you must first locate the bad sectors on it. The easiest way to do this is to test the disk with DTEST. DTEST will not only indicate where the bad sectors are using the same absolute numbering system that HADES uses, but it will also indicate whether the sectors are in use or not. If the bad sectors are not in use, DTEST will ask if you want them marked bad. You should enter N so that they will be available after you have repaired them.

If you do not have DTEST, you can use HADES to find the bad sectors with this procedure. First, run HADES, select the bad disk, and enter the sector mode.

Switch to the ASCII edit mode, and search for a string that is not likely to be found ("Rumpelstiltskin" is a good one to use for this). HADES will step through the disk sector-by-sector, and when it comes to a bad sector, it will exit to DOS with a "FATAL ERROR" message. The sector containing the error will be the sector displayed on the screen + 1. For example, if the sector on the screen is 119, then you know that 120 is a bad sector. To continue the search for more bad sectors, restart HADES and use the Go To Sector command to go to the next sector after the bad one (for example, sector 121 if 120 is bad). Then, search for your string again.

After you locate the bad sectors on your disk, you can find out which files, if any, have bad sectors in them by copying all of the files on the disk to the NUL device (COPY \*.\* NUL). If a file contains one or more bad sectors, you will get the usual "Abort, Retry, or Ignore" message, and you can note the name of that file and then press I to ignore the error and continue copying. After you find the files containing bad sectors, you can use HADES in the file mode to search each file (by searching for a string) to locate the bad clusters in each file, and then you can use the sector mode to find out which sectors in the clusters are bad. Use the Go To Cluster command to go to the cluster before each bad one, and then start a search to find the bad sector. On a normal floppy disk, there will be 1 or 2 sectors per cluster. The chances are that only one sector



on a given track will be bad, and therefore, only one sector in a given cluster. (If there are other bad sectors on the disk, the chances are that they will be on adjacent tracks, in the same area of the disk.)

If HADES cannot load up a particular file without exiting with an error message, then you know that the first cluster of that file is bad.

If you are able to copy all of the files of a disk containing bad sectors to the NUL device, then you know that the bad sectors are in an unused data area of the disk, or they are in the boot, FAT (File Allocation Table), or directory sectors. If HADES displayed a cluster number (instead of some dashes), as well as a sector number when it found a particular bad sector, then that sector can only be in the data area of the disk.

### Fixing the Bad Sectors

You can repair a bad sector on a disk if you can rewrite the sector. HADES can write to any sector on a disk, but unfortunately, it cannot be positioned to a sector that is unreadable. But HADES will allow you to position its sector pointer on one disk, remove that disk and replace it with another one, and then write the sector. So the first step in repairing your dam-

aged disk is to format another disk in the same format as the damaged one. You can use CHKDSK to determine the format of the bad disk.

If the bad sectors are in an unused part of the disk, you can repair them by following these steps. 1) First, format a new disk in the same format as the bad one. 2) Place this disk in a drive, run HADES, select that drive, and enter the Sector mode. 3) Use the Go To Sector command to point HADES to a sector that is bad on the damaged disk. 4) Remove the good disk and replace it with the damaged one. 5) Use the Write command to write over the bad sector. 6) Repeat steps 3, 4, and 5 until all bad sectors have been fixed. Then you can use DTEST, or do a search with HADES in the sector mode, to see if your repair job was successful.

If a bad sector is in a file, you need to do some additional work between steps 3 and 4 of the above procedure. You will have to find out what the data in the damaged sector should be, and enter that data into the "new" sector before you write it to the disk. If the file is a text file that you wrote, you may know what the missing data is, and you can just put HADES in the ASCII mode and type it in. But if it was not a text file, or if it was a word processor file

containing special codes in addition to text characters, you will need to examine a good copy of the file with HADES to determine the contents of the bad sector.

If you have a good copy of the file containing bad sectors on another disk, it probably will not occupy the same clusters on the other disk that it does on the bad one. So you will have to step through the file cluster-by-cluster with HADES until you locate the missing data. If you have a printer, you can do a screen dump (by pressing Shift-Prt Sc on a PC-compatible or Shift-F12 on a Z-100 (with PSC.COM loaded)) for each 128-byte page of the missing data. Then you can enter that data into the "new" sector using the hex mode of HADES.

If you are unable to fix your disk using this technique and it still tests bad, and if you have the COPYII PC program, you can use it to make a duplicate of the disk, and then fix the duplicate. Even if the bad sectors are due to physical damage on the original disk, they can only be magnetically bad on the copy (assuming that the disk was good to start with). For that reason, and to ensure that you do not further damage the original, it is a good idea to always attempt your fixes on a copy if you have COPYII PC. \*

### ACCELERATE YOUR PC!

From PC Technologies, the fastest and most proven way to really speed up your Heath/Zenith PC/XT computer, giving it -AT compatible speeds! Turn your turtle into a -286 rabbit with a 12Mhz 80286 accelerator board.

There are ways to speed up your H/Z PC/XT computer, but you spend too much and get too little. The **286 EXPRESS** is the effective solution, making your H/Z150/160/159 series of computers, or any general PC/XT computer in many cases faster than a standard IBM AT type computer! Simple half-card plug-in installation with keyboard software switchable speed control.

Complete with 16-bit -AT -286 processor, CACHE memory for dramatic speed increase, 12Mhz 80286 processor, and software. **You won't believe your stop watch!**

**\$379 SPECIAL** (\$645 list)

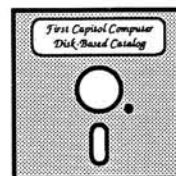
Call or write in to place your order, inquire about any products or request a free no obligation catalog. VISA and Master Card accepted, pick up 2% S&H. We also ship UPS COD and accept purchase orders to rated firms (add 5% to all items for POs). All orders under \$100 add \$4 S&H. Phone hours: 9AM-4:30PM Mon-Thu, 9AM-3PM Friday. Visit our bulletin board: (414) 452-4345.

## QUIKDATA, INC.

POB 1242  
SHEBOYGAN, WI 53082-1242  
(414) 452-4172

Reader Service #193

### This Disk Won't Cost You a Dime... ...But It May Save You Thousands of Dollars!



Yes, First Capitol's new catalog is out - and it's **dynamite!** Imagine *hundreds* of Heath/Zenith and compatible products, and a software program to help you find just what you're looking for! The disk is absolutely **FREE** of charge, as are regular updates to cover new product releases and changes. Need copies for your friends? No problem. We **WANT** you to copy this disk to your heart's content. ...And all products come with the piece of mind of dealing with a proven vendor with over 9 years of experience in the Heath/Zenith community

**First  
Capitol  
Computer**

#16 Algana Drive  
St. Peters, MO 63376  
Order Line: 1-800-TO-BUY-IT  
Other Calls: 1-314-447-8697

**IF YOU CAN'T WAIT** - You can also download our new catalog from our 24-hour Bulletin Board System at 1-314-928-9228. Baud rates supported to 9600 baud.

Reader Service #192

# AN ANSI GRAPHICS HEADER FILE FOR C

**STEPHEN A. JACOB**  
**1038 4TH STREET**  
**HONOLULU, HI 96818**

So, you want graphics, but . . . You don't have an EGA or VGA board. Or you use Patrick Swayne's ZPC — Thank you, Mr. Swayne — with or without a Scottie Board. Or you want simple graphics compatible with many MS-DOS machines, like the Z-110/Z-120. Then you want to use your ANSI device driver.

I wanted lines, boxes and reverse video to start a data base project. My QuickC graphics applications didn't work properly in my emulated PC environment — a Z-120, ZPC and a Scottie Board. So, the ANSI driver seemed to be the ticket. It was.

Several books on C programming mentioned ANSI programming. I've listed a couple at the end of this article. Example programs #define one or two ANSI functions and let it go at that. None came right down to it and wrote an ANSI character header file. So I broke open my MS-DOS Programmer's Utility Pack and created ANSICHAR.h.

ANSICHAR.h (Listing 1) includes all the ANSI functions I will need. And, since I was programming in QuickC, I added a couple of ANSI driver emulations of QuickC graphics routines. You could add more I'm sure. You may want to add ANSICHAR.h to your INCLUDE directory.

ANSICHAR.c (Listing 2) includes the line and box drawing routines I needed. This file can be compiled and placed in your LIB directory, if you wish.

ANSIDEMO.c (Listing 3) is a demonstration of an ANSI driver line drawing. Modify it at will and test the MACROS and functions in ANSICHAR.h and ANSICHAR.c.

• Installing the ANSI driver is not a chore. Installation differs for PC-compatibles and some MS-DOS machines, however.

## Listing 1

```
/*
 * ANSI GRAPHICS
 *
 * ANSICHAR.H Stephen A. Jacob, June 1, 1988
 * Mimics, as near as possible, some QUICKC Graphics commands
 *
 * <PC> Marks PC specific commands.
 *
 * Usage:
 * 1. Place ANSI.SYS, ANSICON.DVD, or ZANSI.DVD in your CONFIG.SYS file.
 * 2. Activate ANSI drivers on Z100 with the line: CTTY=ANSI
 * 3. Deactivate ANSI drivers on Z100 with the line: CTTY=CON
 * 4. Include ANSICHAR.H in your program such as the DEMO program.
 * 5. Include ANSICHAR.C in your compile/link cycle or add ANSICHAR.OBJ
 * to a LIB file.
 */

/* GENERAL DEFINITIONS */
#ifndef TRUE
#define TRUE 1
#endif
#ifndef FALSE
#define FALSE 0
#endif
#ifndef ERROR
#define ERROR -1
#endif

/* DISPLAY CONTROL */
#define ED "\x1B[2J" /* erase display, home cursor */
#define EL "\x1B[K" /* erase to end of line */
#define NORMAL "\x1B[0m" /* normal video, WHITE fgnd, BLACK bkgnd */

#define REVERSE "\x1B[7m" /* reverse video, NORMAL for off */
#define BOLD "\x1B[1m" /* bold text, NORMAL for off <PC> */
#define UNDER "\x1B[4m" /* underline text, NORMAL FOR OFF <PC> */
#define BLINK "\x1B[5m" /* blink text, NORMAL FOR OFF <PC> */

#define BLACK_FG "\x1B[30m"
#define RED_FG "\x1B[31m"
#define GREEN_FG "\x1B[32m"
#define YELLOW_FG "\x1B[33m"
#define BLUE_FG "\x1B[34m"
#define MAGENTA_FG "\x1B[35m"
#define CYAN_FG "\x1B[36m"
#define WHITE_FG "\x1B[37m"

#define BLACK_BG "\x1B[40m"
#define RED_BG "\x1B[41m"
#define GREEN_BG "\x1B[42m"
#define YELLOW_BG "\x1B[43m"
#define BLUE_BG "\x1B[44m"
#define MAGENTA_BG "\x1B[45m"
#define CYAN_BG "\x1B[46m"
#define WHITE_BG "\x1B[47m"
```



On PC-compatibles, like the Z-150 and the Z-248, the ANSI device driver is ANSI.SYS. To install the driver, include ANSI.SYS in your CONFIG.SYS and reboot the system. You can create a simple CONFIG.SYS by typing to the console at the system prompt. Here <CR> means press the RETURN or ENTER key, and <CTRL-Z> means hold down the CTRL key, then press the Z key.

```
copy con config.sys<CR>
device=ANSI.SYS<CTRL-Z><CR>
```

On MS-DOS computers, like the Z-110 and Z-120, the ANSI device driver may be called ANSICON.DVD. You could download ZANSI.DVD from the HUG Bulletin Board as I did. Whatever the name of your ANSI device driver, include it in your CONFIG.SYS and reboot the system. Use the conventions mentioned above to create CONFIG.SYS, if necessary.

Depending on your computer, you may have to activate the driver. To do this, enter CTTY=ANSI. To deactivate the driver, enter CTTY=CON. This can be done in a batch file, like AUTOEXEC.BAT.

ANSI driver graphics have two drawbacks. They are slow and they are... well, simple. If you have an 8 MHz Z-248, slow may not be a concern. And if your application is simple boxes and compatibility is a concern, ANSI driver characters may be for you. Check it out.

Some books include ANSI graphics routines and/or .C programming examples:

Robert Lafore, *The Waite Group, Inc., Microsoft C Programming for the IBM*, Howard W. Sams & Company, Indianapolis, Indiana, 1987.

Carl Townsend, *QuickC Programming for the IBM*, Howard W. Sams & Company, Indianapolis, Indiana, 1988.

MS-DOS version 2 *Programmer's Utility Pack*, Heath Co, Zenith Data Systems.

Continued from Page 4

**Examples** — One or more examples which illustrate typical uses of the command.

**Comments:** As the MS-DOS operating system has evolved, it has become more complex. In addition, most end-users are overwhelmed by the amount of documentation which accompanies the operating system. This package organizes this information and makes it available to end-users in a friendly and easily accessible environment. In addition, the package is both comprehensive and user-extensible, which make it valuable for more sophisticated users.

TABLE C Rating: (10)



```
#define MON040X25      0      /* <PC> */
#define COLOR40X25     1      /* <PC> */
#define MON080X25     2      /* <PC> */
#define COLOR80X25     3      /* <PC> */
#define COLOR320X200   4      /* <PC> */
#define MON0320X200    5      /* <PC> */
#define MON0640X200    6      /* <PC> */
#define WRAPAROUND      7      /* <PC> */

/* CURSOR CONTROL */
#define HVT      "\x1B[1;1f" /* row=1 col=1 (Home cursor) */
#define HVB      "\x1B[20;1f" /* row=24 col=1 */
#define HVP      "\x1B[%d;%df" /* to row and column specified */
#define CUP      "\x1B[A" /* up one row */
#define CUD      "\x1B[B" /* down one row */
#define CUF      "\x1B[C" /* right one char */
#define CUB      "\x1B[D" /* left one char */
#define SCP      "\x1B[s" /* save position */
#define RCP      "\x1B[u" /* restore to saved position */
#define SGR      "\x1B[%dm" /* Set Graphics Rendition */
#define SM       "\x1B[%dh" /* Set Mode <PC> */
#define RSM      "\x1B[%dl" /* Reset Mode, last char is 'el' <PC> */

/* LINES AND CHARACTERS */
#define DBL_TOP_LFT  '\xC9' /* double line corner */
#define DBL_BOT_LFT  '\xC8' /* double line corner */
#define DBL_TOP_RGT  '\xBB' /* double line corner */
#define DBL_BOT_RGT  '\xBC' /* double line corner */
#define DBL_VERT_LN  '\xBA' /* double vertical line */
#define DBL_HORZ_LN  '\xCD' /* double horizontal line */

#define SGL_TOP_LFT  '\xDA' /* single line corner */
#define SGL_BOT_LFT  '\xC0' /* single line corner */
#define SGL_TOP_RGT  '\xBF' /* single line corner */
#define SGL_BOT_RGT  '\xD9' /* single line corner */
#define SGL_VERT_LN  '\xB3' /* single vertical line */
#define SGL_HORZ_LN  '\xC4' /* single horizontal line */

#define DBL_TOP_SGL_LFT '\xD5' /* combination line corner */
#define DBL_BOT_SGL_LFT '\xD4' /* combination line corner */
#define DBL_TOP_SGL_RGT '\xB8' /* combination line corner */
#define DBL_BOT_SGL_RGT '\xBE' /* combination line corner */

#define SGL_TOP_DBL_LFT '\xD6' /* single line corner */
#define SGL_BOT_DBL_LFT '\xD3' /* single line corner */
#define SGL_TOP_DBL_RGT '\xB7' /* single line corner */
#define SGL_BOT_DBL_RGT '\xBD' /* single line corner */

/* FUNCTION MACROS */
#define CLEAR_SCREEN      puts(ED) /* also CLEARSCREEN */
#define LOCATE(row,col)   printf(HVP,(row),(col)) /* also SETTEXTPOSITION */
#define SETGRAPHIC(x)     printf(SGR,(x)) /* <PC> */
#define SETMODE(x)        printf(SM,(x)) /* <PC> */
#define WRAPAROUND_ON     printf(SM,7) /* <PC> */
#define WRAPAROUND_OFF    printf(RSM,7) /* <PC> */

/* SETNORMAL: NORMAL is the only way to reset BLINK, REVERSE, UNDER. */
/* It also resets the foreground and background colors. */
/* So to use BLINK, etc. with colors, you need to reset */
/* it with NORMAL then and restore the desired colors. */
#define SETNORMAL(f,b)     printf("%s%s",NORMAL,(f),(b))

/* "QUICKC" GRAPHICS EMULATION MACROS */
#define CLEARSCREEN      puts(ED) /* also CLEAR_SCREEN */
#define SETTEXTPOSITION(row,col) printf(HVP,(row),(col)) /*also LOCATE */
#define OUTTEXT(x)       printf("%s", (x))
#define RECTANGLE(x,r,c,w,h) draw_box(1,(r),(c),(r+w),(c+h)) /* SGL */
#define DISPLAYCURSOR(x)
#define SETLOGORG(x,y)

/* FUNCTIONS: Declared here. Defined in ANSICHAR.C */
void draw_vert_line(int, int, int, int); /* param 1, 2: row, col to start */
void draw_horz_line(int, int, int, int); /* param 3: length of line */
/* param 4: linestyle 1,2,3,4 */

void draw_box(int, int, int, int, int); /* param 1, 2: row, col to start */
/* param 3, 4: row, column to end */
/* param 5: linestyle 1,2,3,4 */
```

## Listing 2

```

/*
 * ANSI GRAPHICS Functions
 *
 * * ANSICHAR.C Stephen A. Jacob, 31 May 1988
 * * Uses ANSICHAR.H
 * *
 * */
#include <stdio.h>
#include "ansichar.h"

/* functions */
void draw_vert_line(row, col, length, linestyle)
int linestyle; /* SGL=1 DBL=2 SGL_TOP=3 SGL_TOP=4 */
int row;
int col;
int length;
{
    int vline;
    char linechar;

    switch(linestyle)
    {
        case 2:
            linechar = DBL_VERT_LN;
            break;
        default:
            linechar = SGL_VERT_LN;
    }

    for (vline= row; vline <= row + length ; ++vline)
    {
        LOCATE(vline, col);
        putchar(linechar);
    }
}

void draw_horz_line(row, col, length, linestyle)
int linestyle; /* SGL=1 DBL=2 SGL_TOP=3 DBL_TOP=4 */
int row;
int col;
int length;
{
    int hline;
    char linechar;
    switch(linestyle)
    {
        case 2:
            linechar = DBL_HORZ_LN;
            break;
        default:
            linechar = SGL_HORZ_LN;
    }

    LOCATE(row, col);

    for (hline = 0; hline <= length; ++hline)
        putchar(linechar);
}

void draw_box(toprow, leftcol, bottomrow, rightcol, linestyle)
int linestyle; /* SGL=1 DBL=2 SGL_TOP=3 DBL_TOP=4 */
int leftcol, rightcol;
int toprow, bottomrow;
{
    int tlc, trc, blc, brc;

    switch(linestyle)
    {
        case 2: /* DOUBLE TOP *** DOUBLE SIDE */
            tlc = DBL_TOP_LFT;
            trc = DBL_TOP_RGT;
            blc = DBL_BOT_LFT;
            brc = DBL_BOT_RGT;
            break;
        case 3: /* SINGLE TOP *** DOUBLE SIDE */
            tlc = SGL_TOP_DBL_LFT;
            trc = SGL_TOP_DBL_RGT;
            blc = SGL_BOT_DBL_LFT;
            brc = SGL_BOT_DBL_RGT;
            break;
        case 4: /* DOUBLE TOP *** SINGLE SIDE */
            tlc = DBL_TOP_SGL_LFT;
            trc = DBL_TOP_SGL_RGT;
            blc = DBL_BOT_SGL_LFT;
            brc = DBL_BOT_SGL_RGT;
            break;
        default: /* SINGLE TOP *** SINGLE SIDE */
            tlc = SGL_TOP_LFT;
            trc = SGL_TOP_RGT;
            blc = SGL_BOT_LFT;
            brc = SGL_BOT_RGT;
    }

    LOCATE(toprow, leftcol); putchar(tlc);
    draw_horz_line(toprow, leftcol+1, (rightcol-leftcol) - 2, linestyle);
    draw_vert_line(toprow+1, leftcol, (bottomrow-toprow) - 2, linestyle);
    LOCATE(bottomrow, leftcol); putchar(blc);
    draw_horz_line(bottomrow, leftcol+1, (rightcol-leftcol) - 2, linestyle);
    LOCATE(toprow, rightcol); putchar(trc);
    draw_vert_line(toprow + 1, rightcol, (bottomrow-toprow) - 2, linestyle);
    LOCATE(bottomrow, rightcol); putchar(brc);
}

void ansisetup()
{
    CLEARSCREEN;
    WRAPAROUND_OFF;
    draw_box(1,1,23,80,2); /* Z100 wraps around on (24,80) */
    return;
}

```



```

void ansicleanup()
{
    WRAPAROUND_ON;
    CLEARSCREEN;
    return;
}

```

### Listing 3

```

/*
 * ANSI GRAPHICS Demonstration
 *
 * ANSIDEMO.C Stephen A. Jacob, June 2, 1988
 * Uses ANSICHAR.H AND ANSICHAR.C
 *
 * For Z-110/Z-120: GRAPHICS.CHR (renamed ALTCHAR.SYS) contains
 * similar definitions. If you #define Z100 before the #includes,
 * this demo program will look for Z100CHAR.h vice ANSICHAR.h. Only
 * #define Z100 if you have created a Z100CHAR.h file. You should not
 * define Z100 if you want to use ANSICHAR.h.
 *
 */
#include <stdio.h>

#ifdef Z100
#include "ansichar.h"
#else
#include "z100char.h"
#endif

void wait_for_keypress();

main()
{
    int i;
    static char str[] = "VERTICAL LINE";
    char heart = '\x03';
    char flower = '\x05';

    SETNORMAL(BLUE_FG, WHITE_BG);

    ansisetup();
    draw_box(1, 1, 3, 80, 2);
    LOCATE(2, 20);
    printf("DEMONSTRATION OF ANSICHAR.H AND ANSICHAR.C");
    wait_for_keypress();

    draw_box(4, 4, 10, 50, 1);
    draw_horz_line(6, 5, 44, 2);
    LOCATE(5, 5);
    printf("One line box. And a double horizontal line.");
    LOCATE(7, 5);
    printf("This program cycles through a few of the ANSI");
    LOCATE(8, 5);
    printf("codes available with ANSICHAR.h. Of course ");
    LOCATE(9, 5);
    printf("you could print %c's and %c's if you wanted.", heart, flower);
    wait_for_keypress();

    draw_vert_line( 9, 77, 13, 1);
    LOCATE( 9, 78);

```

```

for (i=0; i < strlen(str); ++i)
    printf("%c%s", str[i], CUD, CUB);
wait_for_keypress();

LOCATE(14, 5)
printf("A couple of boxes . . .");
draw_box(15, 5, 22, 20, 3);
draw_box(15, 25, 22, 40, 4);
for (i=1; i <= 6; ++i)
{
    /* print in first box */
    LOCATE(15+i, 7);
    printf("%d %c", i, i);
    LOCATE(15+i, 15);
    printf("%d %c", (i+6), (i+6));
    /* print in second box */
    LOCATE(15+i, 27);
    printf("%d %c", (i+12), (i+12));
    LOCATE(15+i, 35);
    printf("%d %c", (i+18), (i+18));
}
wait_for_keypress();

LOCATE(12, 44);
printf("%s Blinks on the PC", BLINK);
SETNORMAL(BLUE_FG, WHITE_BG); /* turns BLINK off */

LOCATE(13, 44);
printf("%s Underlined on monochrome PC", UNDER); /* <PC> */
SETNORMAL(BLUE_FG, WHITE_BG); /* turns UNDER off */

LOCATE (14, 44);
printf("%s Background is BLUE ", CYAN_BG);

LOCATE(15, 44);
printf("%s Foreground is MAGENTA ", MAGENTA_FG);
SETNORMAL(BLUE_FG, WHITE_BG);
wait_for_keypress();

SETNORMAL(WHITE_FG, BLACK_BG);
ansicleanup();
}

void wait_for_keypress()
{
    LOCATE(24, 24);
    printf("%sPress any key to continue . . .", BLINK);
    SETNORMAL(BLUE_FG, WHITE_BG);
    getch();
    LOCATE(24, 1);
    printf(EL);
}

```

✱

# Greatly Enhanced Wordstar 2000 Plus V3.0

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

## Overview

When MicroPro International released Wordstar 2000 Plus V3 in the Fall of 1987, they not only did a great job in enhancing the features already in V2, but added a number of companion products (discussed later) as well as some major built-in features. I had the pleasure of Beta Testing this product for MicroPro and I was overwhelmed with the power of this release. The purpose of this article is to highlight some of the major features of this new version to give you, the Heath/Zenith User Community, some idea of the power and flexibility of Wordstar 2000 Plus V3.

As I said above, I want to highlight just a few of the features and enhancements since MicroPro International lists almost 400 items in their "What's New" booklet that comes with the product. With this release, MicroPro decided to package the product into two separate packages or editions; the Personal Edition and the Legal Edition, each with most of the Wordstar 2000 Plus V3 product plus a few unique products pertinent to each edition. I will point out the differences later.

During the last year or so, I have noticed, and I suppose you readers have noticed as well, that the new releases of popular software applications as well as some new applications are becoming extremely large. By that I mean, that they have so many features that the code is supplied on 10 or more 360K 5-1/4" floppy disks. Some products, even more diskettes. Before I said that, I was overwhelmed with the Wordstar 2000 Plus V3 release and that includes the number of diskettes to hold all the code. If I counted right, there are 19 diskettes for the Personal Edition and similarly with the Legal Edition.

## Some New Features / Enhancements

**Speed** - The speed of Wordstar 2000 Plus V3 has improved dramatically since the first release, V1.01. For example, to move from the beginning of an 8-page document on a PC-XT machine, to the end of the document, took well over 1 minute with V1.01 and only about 3 seconds with

V3.0; likewise with other tasks. MicroPro did a fantastic job in improving the performance of Wordstar 2000 Plus. There are even certain tasks that Wordstar 2000 Plus V3 can perform better than Wordstar Professional V4.0.

**Speed Writing** - With this feature, the user can go right into entering a document without having to open a file. When finished, the document can be printed and/or saved.

**Cruise Control** - Using a feature called Cruise Control, from Revolution Software, Inc., a user can move the cursor from 10 to 240 characters per second. That's fast. The speed is user controllable.

**FileLocator** - This feature allows the user to locate files easily by author, keywords, words, phrases, or other information. There is no need to really remember file names. Just simply supply some pertinent information in the file you are looking for, and let FileLocator find the file for you. Once supplied with the search information, the FileLocator will search the current directory for all files that contain the search information and display that list on the computer screen.

There is one pre-requisite to using the FileLocator facility and that is all the files to be searched must be "organized."

By this I mean that all the words in the files to be searched must be cataloged or "organized" using the Organize option ("O") from the Opening Menu. This makes it very easy for the FileLocator to find all the files that match a specified string.

**Graphics** - This is a big and very important addition to Wordstar 2000 Plus V3. It allows the user to add graphics to a text document. Users can capture graphic images from other programs (such as a graph from Lotus 1-2-3 or other spreadsheet programs), edit them, and insert them into their documents. The graphics then can be edited, sized, cropped, and rotated. While editing, text automatically flows around the graphic images.

Let me talk a little more on this subject. To show you in a simple example how to add graphic images/pictures to a text document, I have included Figures 1 and 2. Figure 1 shows just a very short document that I want to add a picture to. Included with Wordstar 2000 Plus V3, is a number of files containing clip-art. One file contains a picture of a PC.

Using a memory-resident program that comes with Wordstar 2000 Plus V3, called Inset, one can capture a picture from the screen. This means that while one is using another program or clip-art file, Inset can be activated via a hot-key

The purpose of this document is to demonstrate how one can have a text file and then add a graphics image. This is an image of a PC-type microcomputer. The text in this document was entered first, then the graphics image was inserted. What do you think?

Figure 1



and the entire screen or some portion of it captured. The user can then expand it, shrink it, add some text like I did (Microcomputer), rotate it if desired, and save it.

The next step then is to startup Wordstar 2000 Plus V3, load the text document shown in Figure 1, position the cursor in the document (upper left corner of the picture) where you want the picture/graphic image to go. I placed the cursor on the "T" at the very beginning of the file. Next, one presses CTRL-GM to "merge" in a graphic image. Wordstar asks for the image file, where the image should be placed (left side, center, etc.), whether the image should be superimposed on the text or not, and some margin information. Instead of placing the actual image on the screen at this time, Wordstar 2000 Plus V3 inserts a command tag that provides information to the print program such as the file name, where to place it, etc.

For my example, I told Wordstar 2000 Plus V3 that I wanted the image placed on the right side of the page. What is nice about this feature, is that Wordstar automatically reformats your text to "fit around" the graphic image. After answering the Graphics/Merge questions above, the command tag is inserted, the text is reformatted if necessary, and the word PIX is shown on the screen where the graphic image is to go. The document is then saved and printed using the standard Wordstar 2000 Plus V3 print drivers. Figure 2 is what the document looks like after adding the graphics image. The document was printed on an Epson FX type printer. Over 400 different printers are supported.

**Star Exchange** - This program within Wordstar 2000 Plus V3 converts documents that have been generated on another wordprocessor program to Wordstar 2000 Plus V3 format. It also converts the Wordstar 2000 documents to other formats. Formats supported by Star Exchange are: Wordstar Professional, WordPerfect, Microsoft Word, MultiMate Advantage II, and DisplayWrite.

#### **Just a Quick List of Other Enhancements/Additions**

There are all new function key assignments; in addition there are extended function keys using the ALT key plus 1 through the = key along the top of the keyboard; key macro functions, facility to count the number of words, lines, and pages; new Help facility; a Thesaurus providing 40,000 words with 550,000 synonyms; preview complete pages including pages using column-type text; and LAN support.

#### **PERSONAL EDITION Companion Products**

In addition to the many features and capabilities that are part of Wordstar 2000

The purpose of this document is to demonstrate how one can have a text file and then add a graphics image. This is an image of a PC-type microcomputer. The text in this document was entered first, then the graphics image was inserted. What do you think?



**Microcomputer**

**Figure 2**

Plus V3, there are three Companion Programs, as they are called in the documentation, that add still greater power and flexibility to Wordstar 2000 Plus V3. These Companion Programs (or Products) are: ShowText, PC-Outline, and Fill-A-Form.

**ShowText** - ShowText allows the user to create high-quality presentation visuals and other text charts such as Tables, Signs, and Exhibits. The resulting visuals can either be viewed directly on the computer screen if a graphics board is installed, or hard-copies can be made on a dot-matrix printer, laser printer, or a plotter.

ShowText can be either installed as a stand-alone program or as another add-on product of Wordstar 2000 Plus V3. In this latter case, ShowText can be called at the Opening Menu of Wordstar 2000 Plus V3.

**PC-Outline** - PC-Outline is the easiest and the most comprehensive outlining program available on the market today. It was given the "Editor's Choice" award by PC Magazine. PC-Outline helps the user organize large amounts of information into an easy-to-use, flexible structure. Once in this form, finding information that is needed is extremely easy and fast.

PC-Outline is very flexible and powerful. It supports different file formats, numbering schemes, sorting, editing capabilities, several paragraph formats, multiple windows which allows the user to transfer data between them very easily, etc. Although it is called an outlining program, PC-Outline is much more. It can be used for: a daily scheduler, to-do lists, class notes, a wordprocessor, scratch pad, etc. The uses are unlimited.

Just as with some of the other products, PC-Outline can be installed as either a stand-alone program or in Wordstar 2000 Plus V3 and called via the Opening

Menu. Also, PC-Outline can be installed as a pop-up program (memory-resident or TSR program as it is commonly called). If the available RAM is there, the memory-resident installation makes PC-Outline the most flexible since it can be activated very quickly while executing another program including Wordstar 2000 Plus V3.

**Fill-A-Form.** The last of the three companion products is the Fill-A-Form program that is released with both the Personal and Legal Editions. This program allows the user to enter information very easily in office forms such as invoices, hospital admittance forms, insurance forms, all-types of applications, etc. Just about anything that requires a pre-printed form, can use Fill-A-Form.

With Fill-A-Form, one enters the information in a form on the screen, makes any corrections on the screen, then prints the information on pre-printed forms. This reduces errors and waste of costly forms.

#### **LEGAL EDITION**

The Legal Edition of Wordstar 2000 Plus V3 contains everything that is in the Personal Edition except for a couple of Companion Products. In place of those missing products, the Legal Edition contains a couple of other products, CompareRite and CiteRite, plus a Legal Dictionary that can be used in place of or in addition to the standard dictionary. The products that are missing are Showtext and PC-Outline. The Legal Edition does contain the other Companion Product, Fill-A-Form.

**CompareRite.** The first additional product for the Legal Edition is a product called CompareRite, "The Instant Redliner." This product compares two copies of a document (even source code for programmers) and highlights the differences between them. While comparing

the documents, CompareRite generates a third file that contains the combined documents. The differences are each written on this file in a different highlight mode. For example, one file could use underlining and the other bold printing.

The differences can either be printed or displayed on the screen. When the differences are displayed on the screen for a quick glance, no output file is generated. The user can select how these differences are displayed, such as bold, underline, in brackets, color for screen display, etc. By highlighting the differences, one mode for one file and another mode for the other file, the user can see at a glance where the two files differ.

When comparing files (documents), the only restriction that I can see is that both files must be either Wordstar 2000 Plus files, Wordstar Professional files, Easy files, or ASCII files. One can not mix the formats of the two files. The resulting differences file can be edited with either Wordstar 2000 Plus V3 or Wordstar Professional.

CompareRite can be either installed as part of Wordstar 2000 Plus V3, thus called via the Opening Menu, or installed as a separate product. In either case, the user will be prompted for the two file names plus the name of the third file if the differences are to be written to a file, or simply left blank if the differences are to be displayed on the screen.

**CiteRite.** CiteRite allows the user to check citations to make sure that they conform to the U.S. standard as defined in "A Uniform System of Citation", 14th edition, 1986. This standard is better known as the "Bluebook" which is published by the Harvard Law Review. As the manual states: "Fewer citation errors means more professional, better looking, better respected and, most importantly, more readily accepted documents."

CiteRite is comprehensive; it checks the form of a multitude of citations, including citations to state and federal cases reprinted in official and unofficial case reporters and services; federal and state statutes; law review articles; books; federal legislative history; model codes; and uniform acts.

CiteRite can be used in a number of ways; one, as a stand-alone program; two, as another add-on product to Wordstar 2000 Plus V3 that can be called for the Opening Menu; and three, as a pop-up program (i.e., a memory-resident program). The choice is left to the user. However, the pop-up method is the most useful since it can be called very quickly anytime it is needed.

#### Installation Requirements

Although, Wordstar 2000 Plus V3 works best on a hard disk system, it will work with a dual floppy system provided

at least one of the floppy drives is a "high capacity" drive such as the 3-1/2" 720K drive or the 5-1/4" 1.2MB drive. It is possible after installing Wordstar on a hard disk system to make a 360K floppy disk version. However, many of the features aren't supported with this version such as graphics, indexing, FileLocator, and PostScript if you have a Laser Printer.

During the easy installation process, which is all prompt and menu driven, the user can select what facilities are to be installed. For example, the user does not have to install Graphics or the FileLocator, or any of the Companion Products. It's really up to the user to install what makes sense in his/her environment. Also disk capacity plays an important role in what facilities/products get installed.

The minimum system requirements are the same for both the Personal and Legal Editions: MS-DOS/PC-DOS V2.0 or higher, 384K RAM (512K if the companion products are used as well as CiteRite and CompareRite), hard disk or dual floppy drives with at least one being a high capacity drive, color or monochrome graphics adapter, a printer that will print graphics, and a Hayes (or compatible) modem for the Telecommunications facility.

#### Documentation

The documentation has been completely re-written. Included with this release for the Personal Edition of Wordstar 2000 Plus V3, are the following manuals/booklets: an excellent Reference Guide that covers every command, List/Merge features, Graphics, Indexing, Telecommunications, Converting Files using Star Exchange, Laser Printers, Customization, etc.; a "What's New" booklet listing all the enhancements and new features of this release; Starting and Learning Wordstar 2000 Plus V3 including the installation of V3; the applications booklet which covers how to use Wordstar 2000 Plus V3 for various documents such as Memos, Letters, Newsletters, Video Scripts, Meeting Minutes, Invoices, Phone Directories, Presentation Materials, Contracts, etc.; and the Companion Products Guide covering Fill-A-Form, PC-Outline, and Show-Text (covered elsewhere in this article). The Legal Edition has the same documentation except that it does not have documentation for PC-Outline and Show-Text which are not part of this Edition. It does have separate booklets covering CiteRite and CompareRite which are part of the Legal Edition. MicroPro did an excellent job of providing documentation to help the user.

#### Support

MicroPro has added a new support policy for all registered owners of either the Personal or Legal Editions. Owners will now have unlimited toll-free technical support for the life of Wordstar 2000 Plus

V3 plus 6 months after. The toll-free support number will have someone available 7-days a week. This type of support is rare today but MicroPro wanted to make sure that their users had the support they needed to get their work done. Great idea.

#### References

Wordstar 2000 Plus V3.0, Wordstar Professional, and Easy are products of: MicroPro International Corporation  
33 San Pablo Ave.  
San Rafael, CA 94903

CompareRite and CiteRite are trademarks of:  
JURISoft, Inc.

PC-Outline is a trademark of:  
Telemarketing Resources

FileLocator is a trademark of:  
Richburg Associates

Fill-A-Form is a trademark of:  
Athena Software Corporation

ShowText is a trademark of:  
TimeWare, Inc.

Cruise Control is a trademark of:  
Revolution Software, Inc.

Lotus 1-2-3 is a product from:  
Lotus Development Corporation



---

## CLASSIFIED ADS

**LIKE NEW** Orchid Designer VGA \$250.00. R. Speidel, Box 95E Rt. 1, Emmaus, PA 18049.

**Z-100 POWER SUPPLIES** Reconditioned. \$150.00 exchange. Tom Prescott, 1686 Jennifer Drive, Macon, GA 31206. (912) 788-6077.

**Z-100, PRINTER, MODEM \$\$\$\$ Extras.** Call for list. \$900.00. (808) 878-6096.

**WANTED: HERO JR. PROGRAMMING CARTRIDGE** with/without Jr. (301) 293-7885 EST.

---

In the February 1989 issue of REMark, On the Leading Edge (Page 19) mentioned a company called Lightgate; the author has advised us that this company is no longer in existence.



*We Couldn't Find a Tower that  
was Designed to Our Specs...  
So We Built One Ourselves!*

## The *NEW* First Capitol 386 Tower

### High-Speed 20MHz Operation ...and Lower Cost!

We're not a nationally recognized brand - YET - but we do have one of the *fastest* 80386-based computers on the market! Our CL-386TW line of computers incorporates a 20MHz (no waits) 80386 processor chip fully compatible with Big Blue's finest. *That's 25% faster than 16MHz units!*

Price-wise our CL-386TW units begin selling at less than \$3,000 configured with video, monitor, drive, and 1Meg of RAM! *Compare with prices of \$4,500 and up for comparable units!*

### Designed as a Tower for Space Savings and Convenience

It seems like the faster the system, the more desk space the computer will absorb. Not so with our CL-386TW units! The unit is floor-mounted in a ruggedized case. Only the keyboard and monitor need be located on your desktop. Imagine the joy of having a free desk top once again!

### Incredible Expansion Capability

Included on the motherboard are 2-RS232 standard serial ports, 1-Centronics parallel port, and a real-time clock/calendar with battery backup. Talk about expansion slots - - this baby has 8 slots, with 6 slots open when fully configured!

The chassis contains an over-size 200 watt power supply, plus mounting for *up to 3 floppy devices and 2 hard drives!*

### ...Built the Way YOU Want It!

Unlike the "Big Boys", we don't limit you to the models that we want to build. We build our models the way YOU want them! Every unit is built to your precise order, and burn-in tested to insure quality prior to shipment - all at no extra charge to you.

### We'll Even Sell You a Stripped- Down Unit to Use to Upgrade Your Present System!

If you'd prefer, we'll sell it to you without drives, controllers, or

memory so that you can use it to upgrade an existing system. Not to be confused with our Z-386 upgrades.

### One Year Limited Warranty

First Capitol Computer provides a full one year's limited warranty on parts and labor via module replacement. That is, if your system should prove defective at any time within one full year from shipment, we will replace the defective module with a new one free of charge (other than shipping charges).

### Immediate Availability!

You don't have to wait, they're available NOW! Avoid the Rush! *Call for details today!*

**First  
Capitol  
Computer**

#16 Algana, St. Peters, MO 63376  
Order Line: 1-800-TO-BUY-IT  
Other Calls: 1-314-447-8697



# The Most Important Peripheral

## PRINTERS

### (What The Printer Manual Doesn't Tell You)

William N. Campbell, M.D.

1063 Green Glen Drive

Boothwyn, PA 19061

Copyright (c) 1988 by William N. Campbell, M.D.

#### Background

I began my "sideline career" of "Desktop Publishing" in 1969 by purchasing an IBM MT/ST (Magnetic Tape Selectric Typewriter) and an IBM MT/SC (Magnetic Tape Selectric Composer). The MT/SC was a computer driven instrument that created the finest quality "strike on" typesetting. (It was widely used by experienced commercial printers.) This got me started in the production of typeset text; a Konica camera furnished the photographs to generate halftones; I acquired an A. B. Dick 360 offset printer to make large quantities of brochures and other advertising materials for our 150 acre "Pick-Your-Own" vegetable farm. I wrote the material, prepared the copy, took the photographs, processed the film, and gladly left the preparation of "halftones" and color separations to the "pros"; but I did my share of laying out the "flats", did the offset printing, and one winter actually printed our farm Holiday Greeting cards in 4 color process (we mailed out 4500 cards that year). I also used the equipment to prepare invoices, forms and letters for my medical laboratory practice.

Helen (my good wife) was in charge of planting (meaning she did most of the work) some 20,000 tomato plants yearly, 10,000 cabbage plants, and personally planted several acres of squash, cucumbers, and pumpkins each year. (We used large tractors and automated equipment for the planting). She also did all the bookkeeping for the medical laboratory and the farm. I supplied some ideas, made some decisions, and did the

"Desktop Publishing". As you can see, this was a "Do-It-Yourself" operation. Three years ago we retired from both farming and medicine.

#### Types Of Printers

The IBM Magnetic Tape Selectric Typewriter, and the IBM Magnetic Tape Selectric Composer were both impact (strike on) printers, using the revolving "ball fonts" which were so common in that time period. Each machine printed copy at the speed of 15 CPS (characters per second). I believe these specifications were the last honest printing speeds used in printer specifications to this day, with two possible exceptions - Digital Equipment's printers manufactured between 1975 and 1981 seemed to deliver what they promised, as did the Diablo 630 of the same time period. Today, you will see various dot matrix printers advertised suggesting speeds of 100 to 200 characters per second for "Near Letter Quality" and 300 characters per second for "Draft Mode". Don't believe it.

The average line on an 8-1/2 inch wide sheet of paper contains approximately 70 characters. Three lines, therefore, contain 210 characters. That translates to three lines per second for a 210 CPS printer. Do you see many dot matrix printers printing that fast? You don't need a stop watch to figure out that something is wrong here. What? All printer manufacturers today seem to use the same highly artificial methods for calculating the speeds of their printers. So, you can use these inflated advertised speeds to

compare the relative speeds of one printer to another. Many computer magazines present yearly summaries of the new printers. They usually include a line noting that "we were unable to duplicate the advertised speed" of this printer, or that printer, etc. They add that note as the reviewers are honest people.

I bought my first Heathkit computer in 1977 (an H8), and immediately acquired a Digital Equipment LA36 DECwriter II dot matrix printer which had a "blazing" print speed of 30 characters per second! This speed seemed quite accurate.

I purchased a Digital Equipment Corporation (DEC) LA120 dot matrix printer in 1980. I still own and use this DEC printer, although it is now attached to my H386. This has a "serial" connection to my computer (most printers circa 1980 used serial connections). Its specifications include a speed of 180 characters per second. I believe this. It is the fastest printer I have ever owned. My newest printer is a Hewlett Packard DeskJet, uses a parallel connection to the computer, and it has a rated speed (by the manufacturer) of 240 CPS in draft mode. It is slower than my DEC printer. Draft mode, by the way, usually means that (although fast) it is the poorest quality the printer can produce and still be legible. My DEC printer is **always** in draft mode, but it is legible, and it is fast. My DeskJet produces very good quality output in draft mode, but "I am unable to duplicate the manufacturer's claimed speed of 240 CPS!"

In 1981 I acquired a Diablo 630 "dai-

sy wheel" type of printer, which produced nice quality print, but had a rated speed of 40 CPS. I believe this, also. This machine also had a serial connection to my computer.

Since about 1982, most printers have had parallel connections to their computers, and there have been a proliferation of dot matrix printers, all having artificially elevated speeds and other elevated advertising claims. Hewlett Packard's DeskJet printer is my favorite printer at the moment. It is an "ink jet" type of printer.

The dot matrix printers all use ribbons, and their print heads have increased from 9 pins per head to 24 pins per head and the quality has also honestly dramatically increased. Some of the dot matrix printers are capable of doing some "graphics". I have owned an Epson MX-80 printer, an Epson LQ-850, and a NEC P6 dot matrix printer. They all were worth their price but the fastest I have used is still the DEC LA120, and the best quality unquestionably is printed by the Hewlett Packard DeskJet (ink jet) printer.

The "daisy wheel" printers are on their way to extinction.

The most expensive printers in the PC marketplace today are the LASER printers. You can consider these to be moderately fast, very high quality "copiers". They are capable of producing high quality text and reasonable quality graphics. They still do *not* produce the exceptional quality of a 200 line screened halftone, however. So, photocopiers, halftones, graphic layout persons, and off-set printing are still not outmoded!

What should you purchase for your first printer (or if you were considering updating from your present printer)? I would suggest a medium priced 24 pin dot matrix parallel printer, or the new Hewlett Packard DeskJet (ink jet) printer. Ask an owner of the machine you contemplate purchasing for his/her opinion. Read the most recent computer magazine printer reviews and compare features and prices. Some thought should be given to the following details.

Consider whether you want (or will need) continuous form paper (if so, check the paper feed mechanism on a contemplated printer - you will need a reliable "tractor" feed mechanism and these are often "extras".) How does the continuous form paper enter the machine? From the bottom, or from the rear? How does the paper exit the machine?

Some models of printers that use continuous form paper also offer "extra" accessory sheet feeders. Some printers that use continuous forms allow you to feed a single sheet of paper without removing the continuous forms.

Do you need an extra wide paper feed on your printer? Or, will condensed printing of 132 characters on standard sized paper suit your needs?

Check out several makes and models at your local computer store. Ask about warranties and service available after your purchase. Ensure that replacement ribbons or ink cartridges are readily available. If you have no qualms about purchasing by "mail order", read the advertisements in the various computer magazines and check out the various prices. (If you purchase by mail order, always charge to your credit card if feasible.) Make sure the printer comes with a suitable cable and connectors to fit your computer and the printer (or that the cable is available for a reasonable price.)

As you probably know, I think very highly of Hewlett Packard's DeskJet. The only disadvantage of the DeskJet that I have found is that the ink in the ink cartridges is *not* indelible. I have had no problems with ink smudging on dry paper. As long as you don't spill your coffee on the output copy, there are no problems. I understand that Hewlett Packard is working on indelible ink cartridges for this printer!

#### Setup and Initial Checkouts

After purchasing a new printer, open the carton carefully and locate the Printer User's Manual, remove it, and sit down and look through the Table of Contents and/or the Index for sections on Unpacking, and initial Setup. Read these parts of your manual several times, and then thumb through the entire manual to get a "feel" of its contents. Then unpack the printer and cautiously assemble the machine. Follow any manual instructions with care. Most of the current crop of printers are easily set up and require a minimum of time for this purpose. You will probably find any "dip switches" that are present are already set for the correct settings for your computer. In general, it is wise *not* to change any switch settings at first.

Most printers have a self test mode and the printer itself can be tested *before* attaching it to your computer. Follow the printer manual instructions prudently - see if the printer works all right before you attach it to your computer by cable.

#### Cables

Most Personal Computer printers today use a "parallel" cable to connect the printer to your computer. A few need a "serial" cable. A minority can accept either type of cable. You should be able to purchase (if your printer or computer *did* not come with the correct cable) the necessary cable from your dealer. Since these cables come in different lengths, do take this into consideration. As I noted before, you should have checked out the cable situation *before* purchasing your printer. Remember that you need to have the proper connectors on each end of the cable, as well as procuring the correct cable

itself. Your computer manual and your printer manual usually tell you what kind of connector(s) you need.

In the remote chance that your dealer is unable to supply the correct cable, there are specialty firms that do nothing except make custom cables. Their prices are quite reasonable, their technical knowledge is excellent, their service quick. They can help you via phone if you need technical help over and above what your dealer can furnish. One such firm that I can highly recommend is:

Black Box Corporation  
P.O. Box 12800  
Pittsburgh, PA 15241  
Order Department - (412) 746-5530  
Technical Support - (412) 746-5565

#### Final Setup and Establishing Communication

With both computer and printer disconnected from electrical current, and with both turned off, connect the "computer end" of the cable to the correct connector on the computer (see your computer manual) and the opposite end to the printer (see your printer manual). Now, ensure that electrical power is supplied to both computer and printer. Boot up your computer. Turn on your printer. Set your computer to its parallel mode if your connection is a parallel one. This is most easily done from the monitor prompt by typing MODE LPT1: „P<cr> and there is a space after the word MODE and also there is a space after :. <cr> means you press the Enter key. This "MODE" command can be put in your AUTOEXEC.BAT file so that your computer is automatically set up to communicate with your parallel printer each time you "boot up" your computer at turn on time.

If you have a "serial" connection between computer and printer, then you have 2 options. One option is to run the CONFIGUR command supplied by DOS, and the other is to use two mode commands to establish communication with your printer. I prefer the latter.

You must look up in the printer manual what transmission rate (BAUD) your printer requires, and at what "parity" your printer operates (even, odd, or none) (e, o, or n). If you have a choice of transmission rates, pick the highest. Here is an example of the first MODE command needed for a serial printer operating at 9600 BAUD, and with no PARITY:

```
MODE COM1:96,n,,P<cr> (<cr> means you press the Enter key.)
```

Note that there is a space after MODE, that you only type in the first 2 digits of the BAUD rate, that n stands for "No Parity" and note that there are 3 commas in the command. After you enter this command, the computer may print a short message line on the screen. You can disregard it.

The second mode command that must be entered for a serial connection is:

MODE LPT1:=COM1:<cr> (<cr> means you Press the Enter key.)

Note that there is a space after the word MODE.

As you can see, it is more complicated to connect a serial than a parallel type printer to your computer. With the serial printer connection, the BAUD must be the same for the computer as for the printer.

Also, read your MS-DOS User's Manual section on the MODE command

before attempting to use it to set up a serial printer.

You can enter these two MODE commands, one after the other from the DOS command line or put them both in your AUTOEXEC.BAT file (or other batch file).

You should now be ready to print a

Figure 1

```
10 ' MAKECODE.BAS - Assuming DOS vers 2.xxx or 3.xxx; written in QB;
20 ' Copyright (c) 1988 by William N. Campbell, M.D., 1063 Green Glen Drive,
30 ' Boothwyn, PA 19061 7/28/88
40 '
50 ' (In anything I write, <cr> means to press Enter key)
60 ' (Ø = zero; O = capital o) (1 = one; l = lower case L)
70 '
80 ' This program makes printer code sequence for you. You can send it to
90 ' printer from DOS (monitor) prompt (or from a batch file) using
100 ' 'following line' where <cr> means press Enter (Return) key, and
110 ' filename.DAT means the unique name (maximum of 8 characters)
120 ' you choose for your filename, plus .DAT (Example UNIQUE.DAT).
130 ' (You do not type .DAT as part of file name - program adds .DAT)
140 '
150 ' HERE IS THE 'following line' -> COPY filename.DAT PRN<cr>
160 ' (You do not type <cr> as part of the line, just press Enter)
170 '
180 CLS
190 PRINT " (After you enter your filename, press Enter key.)"
200 LINE INPUT "What is your filename to be (maximum 8 characters).. "; X$
210 X$ = X$ + ".DAT": 'this line appends .DAT onto your chosen filename
220 PRINT
230 PRINT "In following, do NOT enter escape character itself, just the"
240 PRINT "desired sequence of characters your printer manual tells"
250 PRINT "you to enter for whatever function is desired, then <cr>."
260 PRINT "Be sure you use exact upper OR lower case that manual specifies!"
270 PRINT
280 LINE INPUT "Enter character sequence (NO spaces), then <cr>.. "; Y$
290 Z$ = CHR$(27): ' this line puts Escape Code in string variable Z$
300 A$ = Z$ + Y$: ' this line adds character sequence; also see IMP.NOTE below
310 OPEN "O", 1, X$: ' Open THE FILE
320 PRINT #1, A$: ' Create the contents of THE FILE
330 CLOSE : ' Close THE FILE
340 CLS
350 PRINT : PRINT X$ + " now on disk. Exit to DOS & Enter TYPE " + X$ + "<cr>"
360 PRINT "(Above line to satisfy your curiosity as to what is in THE FILE.)"
370 PRINT "Ensure Printer Is Turned On!": ' next line assumes parallel printer
380 PRINT "Next enter MOD LPT1: ,P<cr>": ' (if not previously entered)
390 PRINT "Next enter COPY " + X$ + " PRN<cr>"
400 END
410 ' *** IMPORTANT NOTE ***
420 ' "CTRL Codes" don't use Esc Code; other Printer Codes use "CHR$(#)" in
430 ' "Code Sequence". In such instances, temporarily substitute line 300
440 ' in this program as follows (Also consult an ASCII table). Examples:
450 ' |For CTRL Codes| To do Form Feed (most printers) -> 300 A$ = CHR$(12)
460 ' | ASCII decimal| To turn Epson condensed on -> 300 A$ = CHR$(15)
470 ' | Ø thru 31. | To turn Epson condensed off -> 300 A$ = CHR$(18)
480 ' (In following 2 examples, type desired column number in place of #)
490 ' To set Epson right margin -> 300 A$ = Z$ + CHR$(81) + CHR$(#)
500 ' To set Epson left martin -> 300 A$ = Z$ + CHR$(108) + CHR$(#)
510 '
520 ' When running the temporarily modified program, just <cr> in response
530 ' to line 280's prompt.
```



test document. I suggest you carefully check the following 6 points:

1. Ensure your printer and computer are connected by the proper cable.
2. Make sure your computer and printer are turned on, and that your computer is booted up.
3. Load paper into the printer if necessary.
4. Ensure that your printer is "On Line".
5. Check that your computer is in its root (\) directory and that the root directory contains file AUTOEXEC.BAT.
6. From the monitor prompt type COPY AUTOEXEC.BAT PRN<cr>, noting that there is a space after COPY, there is a space just before PRN and <cr> means you press the Enter key.

The contents of your AUTOEXEC.BAT file should be printed by your printer! If not, carefully review everything that you have done! Review the Computer User's Manual, the Printer User's Manual, and MS-DOS User's Manual. Also, reread this manuscript. If you still can't establish communication between your computer and printer, contact your dealer.

### **Sending Control Sequences From Computer To Printer**

A few printer modes and attributes can usually be altered from a printer's keypad, but *almost all* of a printer's modes and attributes can be changed by sending the printer unique "ESCAPE Sequences". Often, the latter may prove to be a challenging task.

### **How To Create An ESCape Sequence And How To Send It To Printer**

Many printer manuals contain BASIC programs which demonstrate how to change the various modes and attributes. Unfortunately, although these programs are illustrative, writing them and using them are a very impractical way of communicating with your printer. First, you always have to load BASIC, load the program, run it, exit the program and exit Basic. This requires time, and if you compile your BASIC program, it can occupy a horrendous amount of precious storage space. I once wrote a compiled Basic program that changed my printer's mode from normal printing to condensed printing. It worked very nicely but occupied over 90,000 bytes of disk space!

In contrast, I will now show you how to create an ESCape Sequence, put it in a data file that occupies less than 25 bytes, and send it from computer to printer using one very short command line. A neat, quick, and easy way!

You can create any necessary ESCape Sequence by writing one BASIC program and using it to create any necessary Code Sequence, and this program will put the Code Sequence in a tiny data file for you.

When you run this program, named MAKECODE.BAS (Figure 1), you only need to supply an appropriate name for the data file, then type in the unique alphabetical and/or numeric characters (typically 4 to 8 in number) that your printer manual specifies for the desired mode or function. I will show you using actual examples how to do this. (Note that there are certain exceptions to this, and they are explained in the "Important Note" at end of the program in Figure 1.) Then, by using a very short command line (Figure 2), you can send this Data File (containing the desired Code Sequence) to your printer, directly from the "DOS Prompt" or from a small Batch File.

First, you must enter the Basic Program shown in Figure 1. You need only GWBASIC ("Gee-Whiz" BASIC) or QB (QuickBasic) or similar BASIC to do this. I assume you have such a BASIC and know how to enter source code such as shown in Figure 1. Then, carefully check your entry for typographical errors, and SAVE it to disk for future use. Be especially careful when entering a zero or upper case "o"; and also, be careful to distinguish between a lower case "l" and a one. The program itself, when LISTed or printed out will show you which is which (line 60).

Second, you should carefully look up in your Printer Manual the desired Code Sequence. Suppose you have a Hewlett-Packard DeskJet printer, and you desire to send it an ESCape code Sequence to change the printer from its default normal printing style of 10 CPI (Characters Per Inch) to condensed printing of 20 CPI (Characters Per Inch). The HP Manual tells you that the required sequence of characters is Ec ( s # H. ("Ec" throughout Hewlett Packard's DeskJet manual stands for "ESCAPE character".) (The program MAKECODE.BAS will automatically enter the ESCape character for you (line 290).) The only other things you need to know are that when you enter the unique characters of the listed ESCape Sequence, that you do *not* separate the characters by spaces; also, any ESCape Sequence that has either the "#" or "n" character in it, does not mean you should type "#" or "n". These two characters almost always mean you should type a *specific number*! Most of the printer manuals will indicate this, but it won't hurt to emphasize it. In the example we have chosen (instead of typing "#"), we will type 20, since we want a CPI of 20 instead of the default of 10.

After loading your version of BASIC, and loading MAKECODE.BAS, simply run it. The program will only ask you 2 questions. First (line 200) it asks you what to name the data file it is going to create for you. An appropriate name would be DJCOND, so type in DJCOND<cr>. Note that in anything that I write <cr> means you press the Enter key (you do not type

**Figure 2**

COPY filename.DAT PRN<cr>  
(<cr> means Press Enter key.)

in the 4 characters making up <cr> - you press the Enter key).

The program next asks you (line 280) to type in the unique sequence of letters (and/or numbers) that you desire. You type in (s20H and then you press the Enter key. Note that these 4 items that were typed in were all enumerated in the printer manual. The first item was an open parenthesis character, the second item was a lower case s, the next item was 20 (we substituted 20 for "#") and the last item was an upper case H. Note that we typed in exactly what was listed in our DeskJet printer manual, substituting a numerical value for "#" (or for "n" if that had been used). Note also that we did not type in any spaces. Note that if a character was listed in the manual in upper case, I used upper case; if listed in lower case, I used lower case. Last, note that I pressed the Enter key.

If you carried this out and made no typographical errors, there *will* now be on your disk a tiny file named DJCOND.DAT and it contains the precise information that is needed to change the HP DeskJet printer from normal printing to condensed printing.

Watch out for the two most common potential typographical errors - lower case "l" versus a numerical one ("1"); and, upper case "o" versus a numerical zero ("0"). I once spent 2 days trying to make an ESCape Sequence work. It just wouldn't! Finally, I found my error. The sequence contained an "l". I had entered a "1". In other words, I should have entered a lower case "l", but I had entered a numerical "1" (one). Immediately, a comment was added to *that* BASIC program to the effect that the 4th character was supposed to be a lower case "l"! Then, after correcting my error, the sequence worked perfectly.

After you have created DJCOND.DAT on disk (I suggest you carry out this exercise, even if your printer uses a totally different sequence code just to verify that MAKECODE.BAS is working correctly), type SYSTEM<cr> (or whatever you usually do to exit your BASIC to return to DOS), and you will be at the monitor prompt. Type DIR DJCOND.DAT<cr> and you should be rewarded with the appearance of this file name in the directory. We will use the DOS TYPE command to examine the contents of the file DJCOND.DAT. Type TYPE DJCOND.DAT<cr>. You should see on your screen your computer's representation of the ESCape character, and this should immediately be followed by (s20H - different computers use different notations for the

binary representation of the ESCape character. My H-386 computer supplies a tiny left pointing arrow. Other computers I have used have supplied "^E" or "^I" to indicate that there was an ESCape code in that location. Just take my word for it, that if you have carried out the above procedures carefully, something may show up on your screen before the (s20H, and whatever it is indicates the ESCape code is present. Note that you can't reproduce the ESCape code by typing in whatever you may see on your screen - use MAKECODE.BAS (with its CHR\$(27)) to produce the ESCape code.

Here is another example, this time for a Digital Equipment LA120 printer. For condensed printing of 16.5 CPI (characters per inch), this printer manual says ESC [ 4 w - now we already know we don't have to supply the ESC since MAKECODE.BAS does that automatically for us. We run MAKECODE.BAS from BASIC and supply the data file name - DECCOND might be a good choice and press the Enter key. When we are asked to supply the unique sequence of characters, we type [4w and press the Enter key. DONE! DECCOND.DAT is on disk! Note that "[I" is the left opening square bracket (on my keyboard it's just to the right of the "P" key.) Note that there were no spaces before, between, or after [4w when those characters were entered.

It is time to try the procedure on your printer. From the monitor prompt DElete any practice files such as DJCOND.DAT or DECCOND.DAT unless you have a DeskJet or DEC LA120 printer. Then load BASIC and load MAKECODE.BAS, carefully look up in your printer manual the unique ESCape code Sequence for condensed printing for your printer and run MAKECODE.BAS. (Note - if you are using an Epson or similar printer, be sure to read the "Important Note" beginning at line 410 of the program in Figure 1, specifically lines 460 and 470.) When you are asked for a file name, enter 8 or less characters which are meaningful to you (assuming you have an Xyz printer, you might type XYZCOND) and then press the Enter key. When you are asked for the unique sequence of characters, type whatever your manual tells you to, taking into consideration what has been discussed above. Remember MAKECODE.BAS will automatically insert the ESCape code for you, so you do not have to put it in. Press the Enter key when you are finished entering the unique sequence. Exit the program with SYSTEM (or however your BASIC returns to DOS).

Now comes the "moment of truth". How do we send your newly created Code Sequence data file to your printer? I must assume that you have correctly attached the printer to your computer. First, make sure your printer is turned on. Then, if you have not yet configured the com-

puter to your printer, do so using CONFIGUR, or the MODE command(s). For a parallel computer, type (from the monitor prompt) MODE LPT1: „P and then press the Enter key.

Next, refer to Figure 2 and type in the exact short line given, substituting the exact, complete data file name that was created for filename.dat, then press the Enter key. If your chosen file name on disk is XYZCOND.DAT, then you type (from the monitor prompt):

COPY XYZCOND.DAT PRN and press the Enter key. Your printer should now be in condensed printing mode! Print out a short ASCII data file and check it out. Here is an example:

Go to your root directory if necessary and then (assuming your root directory contains AUTOEXEC.BAT) type COPY AUTOEXEC.BAT PRN<cr> where <cr> means "Press the Enter key". The printout should be in condensed printing!

If you do not get condensed printing, check everything very carefully for typographical errors. If your printer has a "serial" connection to your computer serial port, use the two serial MODE commands discussed previously or run CONFIGUR. Your printer must work correctly before you send it the Code Sequence!

You probably have noticed that my program MAKECODE.BAS is very heavily commented! That serves two purposes. First, it is a good didactic method, the redundancy in the comments in the program complementing similar (but differently worded) explanations in this article. Second, it provides me with all the information I will need in the future, to send codes to my printer. Actually, of course, the entire program can be written to occupy just 3 lines! For experienced programmers, my apologies. But experienced programmers have probably written programs to send ESCape Sequences from the printer to the computer!

You can also use MAKECODE.BAS to make data files that will send more than one ESCape code Sequence to your printer. Peruse your printer manual to see if you can combine 2 or more unique sequences after only 1 ESCape code, or whether you will have to send more than one ESCape code and unique sequence to your printer. Try concatenation to see if it will work with your printer. For example, use MAKECODE.BAS to create more than one ESCape code Sequence (lets assume you named the first one A, a second one B, a third one C and so on). Then from the DOS prompt enter:

COPY A.DAT + B.DAT + C.DAT COMBINED.DAT<cr>

The above combines the first 3 listed files and puts them all in a file named COMBINED.DAT. Of course, you can use your own names for each data file. Then, you send COMBINED.DAT to your printer

with the line given in Figure 2. Hopefully all 3 printer parameters will be altered as desired.

One word of caution when dealing with multiple printer codes. Many printers have a "priority listing" that you must follow. One code must precede another which must precede another. Your printer manual will spell this out in detail for you, if there is a priority listing. So, again, carefully read your printer manual!

### Pitfalls

If you have a printer that requires a serial interface cable from computer to printer, ensure that you have the correct cable. The correct cable is called an RS-232 cable and there are two kinds. One is a "straight through" configuration, and the other is a "null modem" configuration (where certain wires are crossed). Most of the cables that connect the serial port of a computer to the serial input of a printer are "null modem" cables. If you have any doubt at all about the "pin out" of the cable, or what kind of connectors are required, tell your dealer or cable manufacturer the exact name and model number of both your printer and computer. As a matter of fact, when purchasing a cable, it is always wise to tell your dealer or cable manufacturer the exact names and model numbers of your printer and computer. They have charts that supply them with the necessary cable and connector information. When connecting parallel devices, all you must make sure of is that the cable is a "parallel" cable and that the connectors are correct. Again, your supplier can determine the connectors needed if you tell him the names and model numbers of printer and computer.

When connecting a printer and computer that have serial interfaces, you must also make sure that both machines are set for the same baud rate (transmission speed). If one is set for 9600 BAUD, and the other for 4800 BAUD, they will not communicate. As I explained previously, you can set the BAUD rate of your computer with one of the MODE commands.

Remember that you must utilize two MODE commands to link your computer to a "serial printer".

If you have a "parallel printer" you do not have to worry about BAUD rates, and there is only one MODE command that must be used to establish a working link between computer and printer.

Once you have your computer and printer working together the only other major obstacle that might confuse you is the nature of the printer's "power on defaults". When you boot up your computer it always sets certain default parameters, and these are always the same each time you boot up (unless and until you purposely change one or more). Both of my printers (Digital Equipment LA120 and Hewlett Packard DeskJet) work similarly.



When either is powered up it *always* defaults to its *same* printing modes, fonts, and functions, unless some are purposely changed by altering switches. This is good and makes it easy for you, the user. Beware of some printers which do *not* work this way. Many printers will *not* power up to the same "factory set" default settings each time. They will "power on" to default settings that are *whatever the modes, fonts and functions were when the machine was last turned off!*

Here is an illustration: This type of printer could have been shipped from the factory with default settings of Courier Font, 10 pitch (10 characters per inch), and Draft mode (fast, but usually poor quality printing). You alter these settings (either by pressing keypad buttons, or by sending ESCape Sequences to the printer or by a combination of both). Let us say you change the Font to "Elite", and change the pitch to 12, and change the mode to Letter Quality. You do some printing, then quit for the day and turn your printer off. The next time you turn your printer on, it will "come up" with the defaults of 12 pitch, Elite, and Letter Quality! This can become very confusing, and disconcerting if you are experimenting with a new printer; and particularly when you are trying out new ESCape code Sequences.

Many Epson and some other printers behave this way, and I do *not* like this. If the manufacturer desires to give the user an option to change the defaults, so be it. That is one thing. But, if you have been using some esoteric settings (for you), finish that job, turn off the printer, and the next time you turn it on, find out you are still dealing with the same esoteric settings - that, to me, is ridiculous! I don't like "power on defaults".

One way to handle this situation, if your printer behaves this way, is to remember to set all the "defaults" back to their original settings (or to whatever default settings you desire), using either the printer keypad or ESCape Sequences, *just before* you turn off the power to your printer. And, you must do this *each* time you turn it off! Otherwise, be prepared to check and possibly alter all the defaults, each time you turn your printer on!

Another solution to this problem is to create a small batch file which contains all the necessary ESCape code Sequences to put your printer in the various modes, settings, and functions that you desire at "power on", and then run this file immediately after you turn your printer on, each time. This way you will always be starting out with known parameters. One thing you must be aware of, however! Printers that behave in the fashion we are discussing frequently have an "Initializing" or "Reset" ESCape code Sequence described in their manual. This code sequence almost always will return all set-

tings to whatever they were when you last turned your printer off. (These printer manuals will frequently use the term "power on default settings" rather than "factory default settings".) So, it is best to leave this particular "Reset" ESCape Sequence code out of your "Initializing" batch code, for it may thwart your purpose and "Reset" your printer to whatever it was set for the last time you turned it off!

### Elementary Troubleshooting

Most printers on the market today come with a one year limited warranty from the manufacturer. Unless you have purchased an extremely high priced printer, you probably won't want a service contract. I have yet to have a personal computer printer develop a mechanical or electronic problem.

The most common problem I have had, is that the print quality becomes poor, the print becoming "gray" rather than a nice "black". There are two easily corrected causes of this problem, assuming your printer has a ribbon. First, check the "lever" that adjusts the relation of the print head to the ribbon. Each printer manufacturer seems to give this "lever" a different name, but the word "lever" seems to be a part of each name. Adjust this "lever" according to the Printer Manual's instructions. If this doesn't help, you probably need another ribbon.

### The Thumb and Finger Test

The most reliable method of determining if your ribbon is at fault is to roll up your sleeves, make sure you have a handy supply of soap and water available, and then carefully run a thumb and forefinger down a portion of the ribbon, squeezing the ribbon gently. If you get ink on your fingers, your ribbon is not at fault. However, you might be surprised that your *fingers remain clean*. If so, you need another ribbon! Just because the current ribbon is "new" does *not* necessarily mean that the ribbon contains enough ink! Ribbons can be quick to dry out.

Consider this *true* story that happened to me. After purchasing a relatively high priced new printer from a well known manufacturer (and taking out a service contract on the printer), I consistently had output that was a light gray on white paper. I adjusted the "lever" according to the Printer Manual without any success. Then I changed the ribbon. Finally, I called for service and the service person promptly arrived the next day. After explaining my problem, the service person applied the "thumb and finger test" described above (that is when I learned the technique), showed me clean finger and thumb and explained that all I needed was a good ribbon. I couldn't believe this and told the service person I had just changed the ribbon the day before.

After asking my permission to use the phone, a call was placed to a large nearby university whose computer department had a number of similar printers. The knowledgeable person who answered the phone supplied the "second source" from whom the university always purchased its ribbons. I promptly contacted the "second source" and put in an order for 6 ribbons. They arrived the next day. I examined one and tried the "thumb and finger test" and then immediately went and washed my hands. The problem was solved. This very well known manufacturer also was well known to supply quite expensive and useless ribbons. I later found out that most of the printer service personnel were well aware of this and thoughtfully steered complaining customers to a "second source". So, if your ribbons fail the "thumb and finger test" when they are new, consider changing your ribbon source!

I can highly recommend the following firm that specializes in new and reloaded ribbons (and also will reload your used "laser cartridges") at very reasonable prices. They provide excellent service. I have dealt with them for many years. If you are interested, either write or call them for more detailed information:

Ben Torres Ribbon Service  
590 E. Industrial Road, Unit 15  
San Bernardino, CA 92408  
Phone (714) 786-5559

### Conclusions

As you can see from this commentary, I began my odyssey to find an affordable, usable solution to Desktop Printing with some very expensive equipment - more expensive than a laser printer, software, and fonts cost today. I never regretted my initial decisions, although I probably didn't save much money by doing it all myself. But Personal Computers were not yet in existence when I began.

In a later article I hope to cover some of the fine points I learned along the way, some practical uses for Desktop Publishing and some money saving ideas for the prospective Desktop Publisher. \*

**EXPLORE  
NEW WORLDS  
WITH  
HUG  
GAME  
SOFTWARE**



**RAYMOND H. MURPHY**  
**P.O. BOX 13242**  
**OMAHA, NE 68113**

# RECOVERING DATA FILES

Normally, I don't go around publicizing my imperfections; however, I learned a few things about recovering files that are worth passing on to anyone who might find themselves in a similar situation. My tale of woe started late one night while working with Condor DBMS on my Zenith. I had selected some records from a dataset for printing. In Condor, this created a temporary "Result" data base with about 15 records in it. After finishing this task, I went on to another data base that contained information on my 1500 volume library. I added a few records, and corrected some others. Neither of these transactions created a temporary "Result" data base. Maybe it was the phase of the moon, or something I ate for dinner, but it seemed like it was a good idea to update the backup copies of the data base at that time. So, I copied the Result data base onto the backup disk. Since this proceeded so rapidly, I decided to copy it onto the master disk. Needless to say, when I called up the Library file, I was shocked to find that it consisted of only the 15 records that I had previously printed. Not only that, but I had done the same thing to the backup disk. On examination, I had not just erased the file — I had copied information over it!!! Needless to say, I was not in the best temper the next day, or for sometime after.

What could I do? Using "HADES", I examined the disk. The directory listed the library file, but upon examining the sector, the only thing it contained was the infamous 15 records from the wrong data base. I have learned the old adage "When all else fails, read the book." But there isn't anything in the DOS book about dumb actions. There was nothing wrong with the commands.

They operated "as advertised". The fact that I told the system to do something dumb, although legal, was a human failure, not a system failure. While in HADES, I decided to look around the disk, and noted several interesting things. First, there was far more data on the disk than the directory indicated. Secondly, some of this "excess" data was portions of my library file intermingled with other data.

For example, there were a few sectors of the library file — about 10 to 15 records per sector, then a sector or two of text from other files, then a sector's worth of some old deleted file, and so on.

Going back to the books, I found plenty of information on disks, files, directories, file allocation tables (FATS), and even how to unerase an erased file, but nothing on how to recover a file that has been written over with the wrong data.

After thinking things over for a while, I noted that DEBUG gives me the ability to change things within memory, and also provides an output capability. This would allow me to extract the desired data from the disk, store it in memory, and then write it to a new disk. I had always been somewhat afraid of DEBUG. It looked too much like assembly language, and that kind of stuff. I had also been told that it was difficult and dangerous. Still, the thought of reentering all 1500 odd records in the data base was even less enticing.

Given the alternatives, use of DEBUG, then, appeared a workable solution. To proceed, the first thing was to make a copy of the damaged disk — in fact, I made more than one — and used them instead of the original disk. DISKCOPY, with the /V flag, copies all sectors to a new disk and verifies the copy. My next step was to determine what was where. I probably could have used DEBUG, but I had HADES, and this provides an easy way to see what is in each sector. Not only that, but it provides me the hexadecimal number of the sector at the same time. Such a task took a few hours and a few sheets of paper. I found that the disk had not only parts of the desired file, but also pieces of previous versions of the file. Since I used this file frequently, there were several versions, and some of the records were in different sort sequences. All the same, I listed these on my disk map, with notes regarding the sort sequence, and other information that I could deduce by viewing the records.

When you load a file into memory when using DEBUG, there are a few things

to bear in mind. First, when I copy data into memory, nothing happens to the original file. The only time things are changed is when the data is copied back from memory to the disk. Secondly, different files are stored in different manners by their parent program. For example, a straight ASCII text file starts at the beginning of memory, and proceeds to an end-of-file marker. Data base files, on the other hand, contain header information that must be provided to the parent program in order for it to handle the data file properly. Figure 1 is an example of a display from DEBUG, covering addresses 0100 to 0180.

Notice that the left-hand column gives the memory address in hex. The first four positions of the address will vary with the system you are using; the last four positions make up the relative address which is of concern to us. In the case of CONDOR, the header takes up from 0100 to 0180. If this were a dBASE file, the header would be different, and contain different information. The best way to determine what is in a particular program's file is to use HADES to view a known good file.

Since I was recovering a CONDOR file, or portions of a file, the header information would be missing or, at best, incorrect. With this in mind, my first action after loading DEBUG was to fill the entire memory (0100 to EFFF hex) with hex 0 by using the fill command, for example, -F CS:0100 L EFFF 0 (The hyphen is the prompt for DEBUG). Next, I wanted to leave space while inputting the data to be recovered. To do this, I started the data at 0180. In this particular example, I loaded six sectors from the disk on A drive starting at record 16h by -L CS:0180 0 0F 6. To insure the correct data was loaded, you can look at the data by using the Display command. Use of -D will show the memory starting at 0180 and increment in 128 byte increments by using the D command. We can look at any desired segment of memory by typing D CS: with the relative address. CS is the abbreviation where your working memory starts, and will vary from system to system. You

could also enter the actual address, but why do more than necessary. It is a good idea to page through the data to insure that what you thought you loaded is there, and you will need to make note of the ending memory location. This is important since we will have to tell DEBUG how much of this data is to be transferred back to disk. To determine the file length, use the Hex (-H) command to do the calculations. For example, -H 0100 0D55 will give the sum 0E55 and the difference 0C55. Since we will want to save the memory starting at 0100 through 0D55, we will want the difference, 0C55, which is the length of file.t. Make note of this figure for later use when we go to modify the registers to save the file. You will find that you will make frequent use of this command if you have significant amounts of data to manipulate.

Figure 2 is a typical header for a CONDOR data file. If you are trying to recover something other than a CONDOR file, it will probably be different; however, the concept is similar. Looking at Figure 2, the left-hand column contains memory addresses in the segment-offset format. Each address is followed by 16 bytes in hexadecimal format, the first is the byte at the offset address followed by the byte at the next address, and so on. To the far right, the content of these addresses is translated from hex to ASCII for those of us who don't prefer to decipher hex.

The 01 hex at relative address 0000-0001 indicates that this is a data file to CONDOR. The next two bytes contain the date the file was created in Julian format. By Julian format, CONDOR means the binary integer number of days since December 31, 1899. Binary integer fields are stored in low order byte first. This means we create the number and then reverse the order of the fields. For example, 1/1/1900 would be 01 00 hex. For our purposes, this isn't critical, so I used a recent date, 15 Jan 88, which is 9C 7D hex. The next byte is used for current drive number. Again, it is not critical, so 02 became my standard since the data files normally reside in drive B.

The title of the file starts at 0004 hex and continues through 000E. Thus, 53 41 4D 50 4C 45 20 20 44 41 54 in hexadecimal is "SAMPLE.DAT". Positions 000F through 0016 are not used and filled with hex 20.

The base number of records is always 1. This is stored in 0017 and 0018 in binary integer format. The next-record-number field, that is, the total number of records + 1, is located in 0019 and 001A hex, again in binary integer format. The length of a particular data record is in 001B and 001C in binary integer format. Since all records are the same length, you will be able to compute this once when recovering different portions of the same data base. The number of fields within a record

is kept in 001D. The number of data fields in the data file is stored in binary integer sequence in locations 001E and 001F. Thus, 0A 00 hex indicates that there are 00 A0 hex or ten records in the file; likewise, 3C 04 would mean there are 043C hex or 1084 records. The date the file was last updated is stored in Julian format as a binary integer in 0020 through 0022. Since we are recovering a file, this field is not of immediate importance, nor worth the effort to calculate and write the correct date. Thus, filling this field with 9C 7D will meet the immediate need. The last field, 0022 hex is the update counter, tracking the number of updates. Fill this with 01 hex. The remainder of the header, from 0023 through 0080 is used by index files, and is filled with hex 0.

Since I was recovering records with the hopes of putting Humpty-Dumpty back together again, I chose to make a standard header and resave, rename, and renumber the records afterwards. I found it easy to save a number of records in a file, check it out, and later concatenate the files into a large one using the DOS COPY command.

The data records start at CS:0180. Each record begins with hex 01 as the first byte, as long as it is an active record. If you find one starting with 00 hex, it indicates that it is an inactive record, and will be deleted the next time you sort. Since I only wanted active records, I looked only for those starting with 01 h. If we were lucky, the sectors of data previously loaded into memory were the records we wanted and nothing else. More likely, there is a combination of desired data and garbage. Our next task is to get the good records into a contiguous series. This is done by moving blocks of memory around with the Move instruction. While it is possible to lose data at this stage, the only loss is that which is stored in RAM. The worst that can happen is that we will have to reload from disk.

The Move command allows us to move a block of data around within memory to a specific starting address. The format for the command is -MCS:100 125 CS:200, which moves the block of memory from CS:100 to CS:125 to CS:200 *starting in reverse sequence*. This means the contents of CS:125 will be moved to CS:225, then CS:124 to CS:224, and so on. In this case, since the addresses CS:100 to CS:125 will not have new data copied over them, they will have the initial data in them! Thus, it becomes important as how we use this command. If, on the other hand, we used -M CS:200 FFFF CS:100, then we would copy the data from CS:200 into CS:100, then CS:201 to CS:101, and so on. In this case, we would write over the data in CS:200 with data from further up in the memory. The actual end of the data would now be at CS:FDFF, but the data from FE01 through FFFF

would now be duplicated, and there would be two file endings, one the true one at FDFF, and the false one at FFFF.

There are two things we can do to keep our sanity while moving data around within memory. The first is to move from the top down, thus overwriting garbage with good data. This will result in a shorter and shorter file the more we use Move. Secondly, to insure I knew where the actual file ended, I put four A's (65 hex) at the end of the original file. After performing a Move, I could then Search for the first occurrence of his string. (-S CS:0100 EFFF AAAA). When I found it, I changed it to BBBB, and so on. In a small recovery, this is probably superfluous, but with a 1500+ record file, it is a life saver, especially if it is necessary to do a lot of rearranging.

To get our now-modified contents of memory containing what we believe is correct data records back to disk as a file, there are several commands we must provide DEBUG so it will translate our desires into action. First, we must name the file using the "N" command. The format is N, drive, file name, and extension. — NB: SAVED1.DAT identifies the file to be saved on A drive is SAVED1.DAT. It does not write the file, nor does it define what to write, which becomes our next task. A word of advice here is to save the recovered data on a new formatted disk with a bogus name. While we could save back to the same disk, it is possible that DOS would write over the file or parts of the file we are trying to recover. My preference was to create a number of SAVED xx.DAT files where xx is the number of the save.

DEBUG requires that we define the amount of memory we want written to disk, and also where to start the task. The easiest way to do this is to identify the end of the file in terms of memory location. Use the Hex computation command (H) to determine the amount of memory between the end and the beginning of the file at 0100.

We then put this value into the CX register. The -R command provides a view of the register contents and flags. The current value of the CX register is probably 0000 since we haven't been doing anything in this area. We then change CX to A3B7 hex by entering -RCX which will give the prompt to enter the desired value for CX, A3B7. To make sure we correctly entered the value, use the -R command again. See Figure 3 for a typical register dump. If you made use of the Move command, it becomes important to identify the true file end, since, as discussed above, moving data results in overwriting data, and results in duplicate end records.

Now we are ready to write the file to disk. This is probably the easiest command since all we enter is -W (Write). The

Continued on Page 60



# Z-100s, Batch Files, Virtual Disks and PeachTree Software

**Rodney E. Cavin**  
**P.O. Box 507**  
**Alamonte Springs, FL 21715-0507**

This started out to be a simple reply to Jim Eilers' letter in the November 1987 REMark asking how to set up a virtual disk in his H-100 to use his PeachText word processor, but one word led to another and here is the final volume.

It was not too many months ago that I sat staring into space wondering how to do the same task when I discovered some articles in REMark and Sextant by some real writers who caught my attention with motivations which catered to basic laziness, and started me on the path to accomplish the same task.

Oh what arrogant ecstasy to discover a letter that I could actually answer! Now it's time for me to return some input to the system to help others of you push back the fog and open the way to using PeachText from a virtual disk.

Those of you who want to skip the basic K - 8, go directly to HERE. Everyone else, put the wagons in a circle and we'll get started.

This is how to set up a virtual disk on an H-100 for using PeachText (or any other program), assuming you have the maximum 768k of RAM, two disk drives and MS-DOS version 3.

You can set up a virtual disk using version 2, but it's under a different file name and I believe there are size limits which do not allow what I describe here. But you may find this interesting for another application. If you're in ZDOS, you might as well take up knitting 'cause you just struck out here.)

Did someone say virtual disk, with a blank stare? Relax, it's nothing but a chunk of RAM memory that we'll tell your machine is another disk drive, I: in this case, and forever more you'll use it just like it was another drive only without all the mechanics to slow things up.

But lesson one: when you re-boot, or shut off the power, all those wonderful

things you just put in your virtual disk are gone forever!

To set up the program mechanics for a virtual disk, you must type in a set of instructions at bootup. And that brings up a preliminary topic, the batch file, so you won't have to type those instructions each time. In fact, I'll show you a method of how not to type more than once.

More blank stares? Batch files are simply lines of plain English (ASCII) instructions which you would otherwise sequentially type on your keyboard to tell DOS what to do. Each time DOS executes a line, it will afterwards return to the next line and execute it. For example, if a batch file consisted of:

```
PT
ZBASIC
```

when executed, it would load PeachText (to non-Peachers, the PT is the load command for PeachText). Then, because you are running a batch file, when you closed PeachText, DOS would return you to the batch file and the next line would be executed, in this case, ZBASIC would be loaded. When you finished with ZBASIC you would be returned to DOS because there are no more lines in the batch file.

(DOS also allows you to cancel the running of a batch file by typing a Cntl-C while the batch file is being read.)

Name a batch file anything you want, the shorter it is the less work to type when you want to run it, but make it something easy to remember. For example, suppose you create a batch file P.BAT to load Peachtext. It's only line would be:

```
PT
```

The .BAT file name extension is reserved by MS-DOS for identifying a batch file and like .COM and .EXE files, .BAT files do not require that the file extension be typed. To execute the above would only require typing a P.

The P.BAT file would be run and DOS would load PeachText. When finished, you would be returned to DOS because there are no other lines to execute in the batch file. Admittedly, this represents the height of laziness as all you would accomplish is typing one letter instead of two to load PeachText. But suppose it were a ZBASIC file called STUFFIT2. Creating a batch file named S.BAT consisting of:

```
ZBASIC STUFFIT2
```

would require only typing the S, saving a lot of keystrokes. A considerable improvement, particularly for a poor typist.

A word of caution though, don't forget that .COM, .EXE and .BAT files are proprietary, and in that descending order. So watch that you don't have a P.BAT file and a P.EXE file on the same disk, as DOS will always run the higher priority file, in this case, P.EXE and you'll never be able to load the batch file even if you type P.BAT, 'cause DOS will ignore the extension and go for the biggie!

I will be using a special batch file in this application, one named AUTOEXEC.BAT. I see a few lights brighten, you've seen that name before. It's just like any other batch file except that this name is reserved by DOS for a special purpose. During the bootup, COMMAND.COM will look for a batch file called AUTOEXEC.BAT, and if it finds one, it will turn the operation over to it.

Ah-ha! With auto-boot, we can go directly from power up to PeachText in a virtual disk without passing GO and entering so much as a single keystroke. Did I say lazy? This is like a government check!

Batch files are great, now how do you write one? Several ways. That cute little line editor included with MS-DOS called EDLIN is one way and I urge you to become familiar with it, particularly to write or edit longer batch files. But for simple files, you can use the copy from the con-



sole functions. To do that type:

```
COPY CON [d:]FILENAME.EXT
```

(The drive name is only necessary if you want the file written to a drive other than the default drive.)

Now, using the P.BAT example you would type:

```
COPY CON P.BAT
```

```
PT CONTROL-Z
```

```
CONTROL-Z
```

On the final Cntl-Z, your one line program will be copied to the disk in the default drive — or another drive if you included a drive [d:] with the file name. Only the last Cntl-Z is necessary to end the instructions, but including that other one on the end of the last program line usually prevents double default drive lines after the program has run, such as:

```
A:
```

```
A:
```

on the screen when you return to DOS.

If you make a typo, Cntl-Z to end it and start over. If a long batch file is involved, such as my examples to follow, I urge you to become familiar with the basics of EDLIN, the line editor which comes with DOS.

Or, use your PeachText. Call up the edit mode, start a file in the usual manner, except be sure to use the .BAT extension so PeachText won't assign a .DOC extension to the file name. Write the lines required, correcting and editing as needed, totally without any formatting instructions. Forget the Cntl-Z's completely, just END the file and it's written.

But get brave, try EDLIN. It's easy.

## HERE

The virtual disk/PeachText programs require two disks, a bootable disk in drive A: (presuming this is your default drive) and a non-bootable disk in drive B:. This is how to put it all together.

Prepare a bootable (system) disk and copy the following files on to it:

```
VDISK.SYS
PT.EXE
PTMENU.PGM
EDIT.PGM
PRINT.PGM
```

The VDISK.SYS file is on Disk II of your MS-DOS program disks.

The other files are from your PeachText which will give you all of the word processor functions in the virtual disk, including the thesaurus and the spelling dictionary (which will come from a non-bootable disk we'll talk about below).

You will need two more files on this disk to make it all work, and you will write them both.

The first is CONFIG.SYS which tells the machine that you are going to configure the system for a virtual disk. Write this file with the copy from the console function by typing:

```
COPY CON CONFIG.SYS
```

```
DEVICE = VDISK.SYS 576 512 CONTROL-Z
```

```
CONTROL-Z
```

If you prefer to try Edlin, or writing with PeachText, the file name is: CONFIG.SYS and its one line is: VDISK.SYS 576 512

The VDISK.SYS will execute the VDISK.SYS program you've already copied onto the disk with the two number groups setting the rules. The first number, 576, is the size of the virtual disk in kilobytes and the second, 512, is the number of bytes per sector.

You may change the 576 kB as you require, but remember to leave enough memory not assigned to your virtual disk to contain the operating system and any other programs that you normally load up. I chose 576 kB here to leave you 192 kB for RAM, plenty of room to run other programs without having to re-boot. My virtual disk is only 488 kB and that allows me enough virtual disk room for PeachText, the thesaurus and the dictionary with plenty of space to create or edit a number of files.

You can also change the bytes per sector, especially if you'll be writing a lot of very short files. See your friendly DOS manual under VDISK.SYS for the options of other sizes and the reasoning behind their choice. In normal usage, you'll find 512 quite acceptable.

The other file you'll have to create is the AUTOEXEC.BAT file that makes it all happen. I suggest you use Edlin or PeachText for this one, but COPY CON if you're a good typist. The file is:

```
DATE
TIME
ECHO OFF
CLS
COPY PT.EXE I: >NULL:
COPY PT.PGM I: >NULL:
COPY PTMENU.PGM I: >NULL:
COPY EDIT.PGM I: >NULL:
COPY PRINT.PGM I: >NULL:
COPY B:PT.THE I: >NULL:
COPY B:SP.PGM I: >NULL:
COPY B:SP86.PTR I: >NULL:
COPY B:SP.DIC I: >NULL:
I:
PT
A:
```

DATE and TIME are needed to allow you to enter the date and time as you normally do at bootup. (This is because COMMAND.COM shifts control to the AUTOEXEC.BAT file before executing the DATE and TIME functions, therefore, omitting them.)

If you have a Smartwatch, enter the load instruction in place of the DATE and TIME lines. Also, make sure you copy that program to this disk so it will operate.

If you use any programs which reside in background in memory, such as PERKS or an on-screen clock display, etc., add their load instructions after the date and time instructions. And, as above, make sure you copy those programs onto this disk. If you do not want to see the loading

responses from those programs on the screen, add the >NULL: to their instruction line as you see later in the AUTOEXEC.BAT file lines.

ECHO OFF and CLS keeps the loading notes from echoing on the screen and clears the screen of everything up to this point.

All of the COPY lines are obvious, loading all of the needed files from the bootable disk (in drive A:) and the non-bootable disk containing the thesaurus and the spell checker (in drive B:), which will be described in a moment, I promise.

If you have any other programs, maybe templates for PeachText letters, articles, etc., make sure you copy them onto the bootable disk and add instruction lines to copy them to drive I:. Again, use >NULL: if you want a clean screen.

Now, everything you'll want to run has been copied into the virtual disk drive, I:.

The I: instruction changes the default drive to the virtual disk, drive I:, where you'll be working for a while.

PT, of course, loads PeachText and you're ready to start.

When you close PeachText with the EN, the last line of the AUTOEXEC.BAT file, A:, will be read and return drive A: as the default. If you do not want that to occur when you close PeachText, then omit this last line and you will remain in the virtual disk.

All of those >NULL:'s at the end of the file lines simply send any backtalk from the program's action to a NULL file (which the program will create for you) instead of to the screen. It's a good clutter killer. If you want to watch all the action, eliminate the >NULL:'s.

Make the second, non-bootable, disk for the thesaurus and dictionary files. Your dictionary will grow, and you may need other files on the bootable disk, so there is no point in crowding. Copy the following PeachText files to this second disk:

```
PT.THE
SP.PGM
SP86.PTR
SP.DIC
```

Copy the SP.DIC last to speed up reading and writing of this file. (This will allow the file to be placed contiguously on the disk and will be quicker to read and write if the drive doesn't have to look all over the barnyard for the sectors of the file.)

You can create files which will reside in I:, or you can work with existing files from their disk in the regular drives. If I am going to edit a file, I will copy it into I:, do the work and copy the final version back to the original disk if I need to keep it.

Again, COPY ANY FILE YOU CREATE OR EDIT IN THE VIRTUAL DISK TO A REAL DISK OR YOU WILL LOSE IT WHEN YOU DESTROY THE VIRTUAL DISK BY

## RE-BOOTING OR POWERING DOWN.

And if you're working on a long one, take some time to copy it to a disk every once in a while. Don't let two hours' work disappear from your virtual disk forever just because some idiot drives his Yuppiemobile into a power pole and puts your lights out for awhile! He may deserve his fate, but you don't need to share it.

Since your dictionary will grow as you add words, and you'll need to keep the SP.DIC file current on the non-bootable disk (as well as on its back up disk — you do keep a backup don't you?) I suggest you write another batch file, CS.BAT (for CopySpeller) and remember to use it before you end your work in the virtual disk.

```
CS.BAT is:
REM TO COPY SP.DIC FROM I: TO A: AND B:
PAUSE Insert SP.DIC and BACKUP in A: and
d B: (RETURN when ready)
COPY I:SP.DIC A:
COPY I:SP.DIC B:
```

This file should be on your bootable disk and you will need to add a line to your AUTOEXEC.BAT file: COPY CS.BAT I: >NULL: to copy the file into virtual disk drive I:.

If you'd like to program the copying of the speller at the end of each session with PeachText, add the instruction: CS to the AUTOEXEC.BAT file between the PT and the A:. Then when you close PeachText, the AUTOEXEC.BAT file will run CS.BAT. (You can always skip it with Cntl-C.)

Now if you've waded through all of this, you should be a little familiar with how the objectives were reached and be able to build on or simplify to your own needs.

(You pros who have plowed through this just to test me have no doubt found some areas not quite always and totally true. Right, but this is just to introduce the general operation and give a specific sample. The readers can and are encouraged to explore further to their hearts content. Besides, if I did try to go deeper, I'd really get the mail.)

You'll learn that it takes a minute to load everything when you bootup using your virtual disk, and the wait can be a royal pain if you only need to write a short note or notes and don't plan to need your spell checker or thesaurus.

If so, you may prefer to split up the batch file instructions as I do. I bootup to use the virtual disk, but before everything is loaded, the AUTOEXEC.BAT file ends and I choose using the virtual disk with or without either or both the spell checker or the thesaurus with a second file, V.BAT, which finishes the loading with the choices I've made. To do that, make your AUTOEXEC.BAT file:

```
DATE
TIME
ECHO OFF
CLS
COPY V.BAT I: >NULL:
```

```
ECHO For Virtual Disk enter: V -
with Spell Dictionary: V S
ECHO - with Thesaurus: V T - with both:
V S T
ECHO (SP-Thesaurus disk must be in B:)
```

And write the V.BAT batch file which will contain the rest of the instructions:

```
ECHO OFF
COPY PT.EXE I: >NULL:
COPY PT.PGM I: >NULL:
COPY PTMENU.PGM I: >NULL:
COPY EDIT.PGM I: >NULL:
COPY PRINT.PGM I: >NULL:
IF "%1P.DIC"=="SP.DIC" GOTO D
IF "%1P.DIC"=="SP.DIC" GOTO D
IF "%2P.DIC"=="SP.DIC" GOTO D
IF "%2P.DIC"=="SP.DIC" GOTO D
GOTO T
:D
ECHO Spell Dictionary being loaded.
COPY B:CS.BAT I: >NULL:
COPY B:SP.PGM I: >NULL:
COPY B:SP86.PTR I: >NULL:
COPY B:SP.DIC I: >NULL:
:T
IF "PT.%1HE"=="PT.THE" GOTO TH
IF "PT.%1HE"=="PT.THE" GOTO TH
IF "PT.%2HE"=="PT.THE" GOTO TH
IF "PT.%2HE"=="PT.THE" GOTO TH
GOTO E
:TH
ECHO Thesaurus being loaded.
COPY B:PT.THE I: >NULL:
:E
I:
PT
A:
```

All of the previously mentioned extras you may need will have to be added to these batch files. The hidden calculators, clocks, etc. in the AUTOEXEC.BAT file and other Peachtext files, templates, etc., will go in the V.BAT file.

You may enter the T and/or S in any order after the V (space), upper or lower case and the batch file will function. The only critical entry is to use the V first (the file name) and include the spaces between the entries as shown.

(If you want to read more about this in the batch file part of the DOS manual, you're using those additional conditions %1, %2, %3, etc., to insert things in the running of the batch file.)

In the AUTOEXEC.BAT file, the V.BAT file is copied into the virtual disk so that if, later in your session, you find you need the speller or the thesaurus, you can do the V S T and the batch file will function without having to leave I: for A:.

The ECHOs print the information on screen when the file is running to remind you of your choices.

In case you noticed some BASIC-like phrases in the V.BAT file, you're right. Batch files allow some of this decision making and I used it.

The V.BAT file loads the various files as before, then looks to see if an S or s was entered at the beginning. If so, the program skips to :D and copies the dictionary files to I:, then goes to :T. If not, the program skips directly to :T.

At :T a check is made for a T or t and if either is found, the thesaurus is copied; if not, the program skips to :E. :E finishes the file, changing the default drive to I: and loading PeachText.

You will note I added on screen echos to let you know that the speller and/or the thesaurus are being loaded. They are just to remind you what you selected and what the program is doing when you wake up after the wait while these programs are being loaded.

If you use a hard disk, you may find PeachText operates fast enough in reading and writing not to warrant setting up a virtual disk, but it can easily be done. Because you will probably not always want to bootup with a virtual disk taking up much of your RAM memory, I suggest you use a bootable floppy as I have described. That way, if you auto boot from the hard disk, you'll have your normal memory; but if you want to set up the virtual disk, re-boot from the floppy.

If you change often between the edit and the print modes, while finalizing a document for printing, you'll discover that PeachText is much quicker without having to take the time to open and close files on a physical disk.

I tried to overcome this problem by buying one of the new and popular WYSIWYG word processors only to discover a couple of important functions which I used regularly in some of my writing could not be done with super Whizzer word processor.

Fortunately, about that time I discovered the virtual disk and set up my program. Now, moving from print to edit to print, et al, is quick enough to be acceptable in my application.

And bless its heart, PeachText still works even under the duress of your friends laughing at your antique word processor and its cluttered screen. Cluttered? Those clutters are my security blanket, reminding me of the details of formatting at all times while normally not even seeing the extra little funny words and slashes while I'm deep in thought of my text.

(Try to create your own variable counter, print it anytime, anywhere, as often as needed, then change it however you wish as necessary. I do it regularly to count and print scene numbers in a script writing template I use in PeachText.)

(Try that with the new and very popular Whizzer! Especially after explaining your needs to a knowledgeable salesman who assures you that Whizzer will do it all — then, \$500 lighter and weeks later when you're talking to the Whizzer company the phone nicely sez, "Gee . . . you can't do that with Whizzer. Anything else I can help you with?" "Wanna send my 500 bucks back?" Then it gets very quiet, save for the "click", and in some exchanges the dial tone . . .)

I hope this has been of help, and per-

haps has motivated and enlightened others to jump in, read through the MS-DOS book and try a few batch files and create your own special virtual disk. After all, what can really happen?

WILD INTERRUPT . . . WILD INTERRUPT . . . \*

Continued from Page 56

disk drive should whirl, click, and chatter, and our file is written to disk. -Q will let us quit DEBUG.

I found it useful to immediately go into CONDOR to see what, if any problems were created. If I was satisfied, then, I would sort, select all records (SELECT WHERE ALL), and save the resultant file under a different name. This save action solves the bother with trying to straighten out the header by creating an altogether new file with its own name, header, and date.

Some of the problems you may encounter are:

**A. Partial Records.** We know that each record starts with 01. If we have a partial record at the end of a sector, CONDOR will get confused if we try to run it. The result is usually only an unreadable record if you are lucky. If you can't find the rest of the record, it is best to delete it. Records can extend over a record boundary, but must be contiguous.

**B. Records Split Over a Sector Boundary** with other data in between. This is a fairly common occurrence since files are not necessarily stored contiguously, but crammed into available sectors. The best way to handle this is to make the record contiguous by deleting the intervening material. An easy way to do this is to use the -M (Move) command.

**C. When CONDOR starts doing funny things**, such as displaying garbage or

listing a partial record, the most likely problem is a partial or off-length record. This can happen if you pick the wrong 01 byte as record start point. Records can legitimately have 01 entries as data. In these cases, I found it necessary to actually count the record length. Usually, I found the length was wrong and had to insert zeros to fill out the field, or delete the record.

In summary, it would have been easier not to mess up the data file to begin with. Still, it gave me the opportunity (?) and motivation to learn how DEBUG operates. And, yes, I was able to fully recover all, but one, of the 1508 records originally lost.

My thanks go to Jeffrey Joll of Condor Computer Corporation, Technical Support for his assistance in identifying the Condor file characteristics. \*



## IBM XT in an H/Z-100

Scottie Board W/ZPC \$149.

### OPTIONS AVAILABLE

- PC Compatible Serial Port W/Cable \$50
- 2nd Port W/Cable \$45. • Clock/Calendar \$45.
- No Solder H/Z-100 MOD Kit \$5.

Requires H/Z-100 MS-DOS 768 of RAM, H/Z-100 Modification.

### FACTORY NEW...THE REAL THING

- Z-217-1 W/DOCS, HDWE, CABLES, INSTRS
- With 20 Meg ST-225 Drive \$599.
- With 40 Meg MS3650 Drive \$729.

### \*\*NEW\*\*

- Pair of AT Drives - External, Self Contained, Complete
- Uses 1.2 Meg AT Floppy Drives • W/Power, Cables,
- 34 pin to 50 pin adapter, Instructions - \$299.
- 34 pin to 50 pin adapter alone - \$23.95

IBM XT is a registered trademark of IBM Corp. • ZPC is a product of HEATH USER GROUP.

### PC/XT/AT and WORKALIKES

- LOGITECH Scanman 4" x 10" Scanner - \$199.
- C7 Serial Mouse - \$ 79.
- HI RES BUS Mouse - \$ 89.
- Complete PC Hand Scanner - \$169.
- GRAVIS Joystick - \$ 39.

### HARD DRIVES - All Makes/Models at BEST PRICES

ADD ON CARDS, VIDEO, Controllers, Adapters

Tape Backup 40/60 MEG, 2 MEGS/minute.

Colorado Memory Systems DJ-10 \$299.

VCR Tape Backup "IMAGER" \$199.

### BEST PRICES ON:

TURBO XT, 80286, 80386 Computers -

BUYING SERVICE - any item you wish us to purchase COST + 10%

GIVE US AN OPPORTUNITY TO BID ON YOUR NEEDS

### Scottie Systems, Inc.

1609 S. Main St.

Milpitas, CA 95035

(408) 262-5021

VISA & MC ACCEPTED, ALL PAYMENTS SUBJECT TO APPROVAL

Reader Service #121

# Z-100 Graphics Software

DOODLER-V Graphics Package **\$99.00**  
with D-MOUSE Mouse Driver **\$109.00**  
with Logitech C7 Mouse & Driver **\$189.00**

Font Library Disk for DOODLER-V **\$29.95**  
*44 Additional Ready-to-use fonts*

Texture & Symbol Library Disk **\$29.95**  
*Hundreds of pictures, symbols, textures*

ScreenPro Utility **\$59.00**  
*The ultimate screen print/capture utility*

**FREE** Catalog available on request

*From the Leader in Z-100 Graphics...*



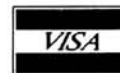
**3620 Amazon Drive**  
**New Port Richey, FL 34655**



Order toll-free...

**800-346-2152**

Or in Florida... 813-376-9347  
Unconditional money-back guarantee  
if your not satisfied!



Reader Service #107





## 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.

**TURBOZ** is a library of machine-dependent routines for Borland's Turbo Pascal Versions 4.0 and 5.0, which enables the Zenith Z-100 to compile and run Turbo programs in native Z-100 mode. To use this product, you will need Borland's command line version of the compiler, TPC.EXE, and the SYSTEM and DOS units in their TURBO.TPL file. You may write Turbo programs in your favorite editor, and compile them using TPC.EXE, the command line version of the Turbo Pascal compiler. The resulting programs will run on the Z-100 without assistance from a PC emulator. TURBOZ does NOT enable you to use TURBO.EXE, the integrated environment in native mode. TURBOZ's main aim is to provide portability for Turbo Pascal source code to and from the Z-100 and PC compatible computers.

**Why you need TURBOZ.** Borland's TURBO.TPL library file contains code which is very machine-dependent. Any code which must control the screen display, cursor, keyboard, and printer is usually unique to the particular computer for which it is designed. In the case of the keyboard and printer, the code CAN be designed to be MS-DOS generic. In their quest for speed, Borland designed even this code to be PC-dependent. They totally bypassed MS-DOS for nearly every function except in the DOS and System units. For you advanced programmers, there is one other benefit to avoiding DOS's built-in services besides speed. You can use many of Turbo's I/O (Input/Output) procedures in interrupt service routines. The TURBOZ units contain Z-100 unique code which performs nearly all the same functions as Borland's code for the PC.

**Requirements.** A Z-100 computer equipped with version 2.xx of the monitor ROM, Turbo Pascal Version 4 or 5 from Borland International, MS-DOS vers. 2 or above, at least 256K of memory for the command line compiler.

### Disk 1

**TURBOZ.DOC** — Full documentation  
**Z100CRT.TPU** — Console interface unit with windowing capabilities  
**MINICRT.TPU** — Alternate console unit

with less overhead/capabilities  
**ZGRAPH.TPU** — Graphics unit with ViewPorts and clipping  
**ZGRAPH2.TPU** — Auxiliary Graphics unit with Doodler support  
**ZLST.TPU** — Printer unit  
**CRT2.TPU** — Supplies features found in TURBOZ to Borland's CRT unit  
**GRAPH2.TPU** — Doodler support for the PC  
**README.DOC** — Latest changes  
**\DEMO** — Source and executable code demonstrating the implementation of graphics in TURBOZ  
**\XYPLOT** — A fully functional XY data plotter with source demonstrating windows

### Disk 2

**\*.ASM, \*.OBJ, \*.PAS** — Complete source code to everything except the DOS and System units.

**Setting Up.** To get started, simply copy TPC.EXE and TURBO.TPL from the Borland distribution disks, and the contents of TURBOZ disk #1 onto a working disk or into your Turbo directory on your hard disk.

**Capability.** TURBOZ provides nearly complete compatibility with Borland's IBM version to the extent that the Z-100 hardware will allow it. This allows software to be developed on the Z-100, and then recompiled using IBM units with little need to change the source code. Conversely, if Turbo source code is available, it could be recompiled using TURBOZ for use on the Z-100.

Z100Crt features built-in line editing capability for any Read or ReadLn, and a super-fast screen-output routine for Write and WriteLn. Every attempt has been made to duplicate the behavior of the routines in Borland's CRT unit wherever the Z-100 will allow it, even Windows are implemented! This unit makes writing programs with a professional touch easy. Sophisticated user interfaces are simple to implement using these basic routines. A new feature as of version 3.1 is the capability to Write or WriteLn the entire IBM™ character set, including the rich graphics character set. I should emphasize that this does NOT require ALTCHAR.SYS! The IBM set is a standalone capability in Z100Crt. This makes back-porting PC source to the Z-100 nearly trivial.

MiniCrt is a scaled-down version of Z100Crt which offers most of the standard Crt functions except windowing and input editing. If you just want to knock out a "quick and dirty" program without a lot of overhead, this is your unit.

The ZGraph unit likewise implements all the pertinent features of Borland's Graph unit except line types, pattern filling, and stroked fonts. Even ViewPorts are implemented complete with clipping! Solid-fill (paint) is provided, and the GetImage and PutImage routines provide support for the BitBlt boolean operators allowing special effects, such as animation and negative images. All the fundamental graphics routines are written in carefully crafted assembly language for speed and compactness. If you are using Turbo Pascal 5.0, Arcs and Sectors are now drawn using the 8087 if you have one. If not, the real number math is emulated. All other graphics require only longint math, at most. ZGraph for Turbo 5.0 features some new procedures: FillEllipse, Sector, SetAspectRatio, and SetWriteMode. These are documented in the Borland manual.

The ZGraph2 unit supplements ZGraph by providing your choice of two bit image routines which support Paul Herman's Doodler V ".PIC" format. Get50Pic/Put50Pic works with Doodler 5.0x, and GetPic/PutPic works with Doodler 5.1x. The Graph2 unit provides these same routines for the PC. The Z-100 routines support the boolean operators, but the BitBlt parameter is currently ignored by the PC routines. You can exchange screen images both ways between your programs and Doodler V!

ZLst provides BIOS output to the PRN device via the familiar Lst device. Your programs will not hang if the printer is not available; this unit will time out after a few seconds of trying to access the printer, and return to your program. The boolean function LstReady returns True if the printer is online and may be called prior to attempting printer output.

**About the Documentation.** The brief documentation is provided to explain several details unique to these units as implemented on the Z-100. It is not intended as a replacement for the Turbo Pascal manual or the interface listings of each of the units for more details, calling syntax, and parameters.

TURBOZ (2 disks) \$25.00  
 1 disk w/.TPU files, documentation  
 and interface listings \$15.00  
 Keith Greer  
 1405 Bills Drive  
 Beavercreek, OH 45385  
 (513) 429-1432 (Voice)  
 (513) 429-5818 (Data)

In three years, **Microsoft Windows** has emerged as the industry graphics-based environment for 80286-based PCs. As a result, there has literally been an explosion of graphic applications for PCs. Windows' consistent, easy-to-use interface lets you take advantage of existing DOS and any new Windows and MS OS/2 Presentation Manager applications. Once you learn its standard user interface, you'll be well on your way to knowing how to use all the new graphics-based applications.

ZDS is pleased to announce Microsoft Windows 2.1.

This new version has a new easy-to-follow installation process that automatically identifies your hardware configuration for you — no more guessing about your memory or hardware configuration.

Windows 2.1 also supports Microsoft's Extended Memory Specification (EMS). This adds 64K to the 640K that DOS programs could previously address. This new "high-memory" feature provides up to 87% more performance for Windows applications than did the previous versions.

Microsoft Windows comes in two versions: Microsoft Windows/386 and Microsoft Windows/286. Both provide powerful capabilities and features that will make your personal computing easier and more productive, and both provide exactly the same graphical user interface.

### Configuration Control at Last!

Many major application programs "want" all of your computer's resources to run. This often means that one CONFIG.SYS file won't work for all programs. In fact, for best results, each major program needs its own CONFIG.SYS and AUTOEXEC.BAT files.

This requirement usually leads to a flock of short batch files which are used to copy renamed CONFIG and AUTOEXEC files into the "real" CONFIG.SYS and AUTOEXEC.BAT files. To reconfigure and run a program, you execute the appropriate batch file and reboot. This technique leads to "short-file clutter" and wastes disk space. In addition, you will often bootup just to reconfigure and reboot.

The **FBE Configuration Control System** allows simple menu selection of one of several previously defined configurations during bootup. You boot from power up directly into your application program with the proper CONFIG.SYS and AUTOEXEC.BAT files in place.

The clutter of "configuration control" files is eliminated because all of the configuration information is contained in standard CONFIG.SYS and AUTOEXEC.BAT files. No modifications of any sort are made to DOS or to any files except the CONFIG.SYS and AUTOEXEC.BAT files. No spurious error-messages-to-be-ignored are generated during operation.

The FBE Configuration Control System is priced at \$29.95 postpaid and is available from FBE Research Company Inc., P.O. Box 68234, Seattle, WA 98168 or telephone (206) 246-9815 M-F 9-5 Pacific Time.

The **Staunch 8/89'er**, the quality quarterly newsletter for H-8 and H/Z-89/90 computer users, is extremely pleased to announce that it begins *bimonthly* publication in January. Because of the increasing frequency of appearance, the annual subscription rate rises to \$12 per year (overseas please add \$4 for air mail delivery). But see below for a renewal special!

Becoming bimonthly was directly tied to an increase in circulation to 500. Projections to the end of this year suggest that this goal will, indeed, be achieved. Staunch would like to thank those individuals and organizations who contributed assistance in the promotional campaign. These are Lee Hart of TMSI, the national HUG, Peter Shkabara of Anapro Corp., and David Powers of Generic Computer Products, Inc. Without their support, increased publishing frequency could not have occurred.

Staunch's public domain and royalty software library, initially prompted by a generous donation by Peter Shkabara, continues to grow. We expect to issue a catalog in late winter or early spring. Recent acquisitions include more releases from Ray Massa of Studio Computers for both HDOS and CP/M and over 35 printer device drivers for HDOS, written by John Smith of THUG. Moreover, Staunch has commissioned keying in of the now-public-domain HDOS 2.0 manual! Dan Jerome and John Toscano, both members of SMUGH, are supplementing and updating this material. At the moment, the first two chapters (system configuration and the operating system reference) are available in either hardcopy or disk form.

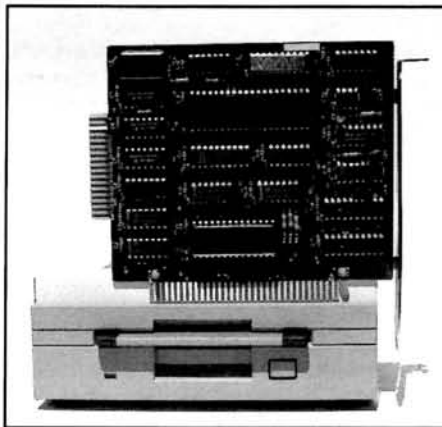
Further, the staff of Staunch remains the same for the new year. Kirk L. Thompson continues as editor/publisher and Hank Lotz, the newsletter's creator, as contributing editor. Staunch is also pleased that Richard Streeter, author of system enhancements and printer device drivers for HDOS 3.0 is writing a quarterly column on the new HDOS. His third such will appear in the first bimonthly issue in mid-January.

As Staunch shifts to bimonthly publication, it also celebrates its second birthday. We would like to thank advertisers

SigmaSoft and Systems, TMSI, and Generic Computer Products for their support during the past year. We also thank the increasing number of authors who have contributed so much to Staunch's pages. Of course, we continue to pay for articles; if interested, please write for our author's guide.

Finally, to celebrate the start of its third year, as well as increased publishing frequency, Staunch is including a premium with all renewals for 1989 received before 1 January. Subscribers may select either a terminal escape code/graphics symbol reference chart or any item of public domain software in Staunch's library. Back issues are also still available. These can be had for \$5 for the 1987 set (edited by Hank Lotz) and \$8 for the 1988 set (edited by Kirk Thompson). Please address all orders, subscriptions, and inquiries to:

The Staunch 8/89'er  
 Kirk L. Thompson, Editor  
 #6 West Branch Mobile Home Village  
 West Branch, IA 52358  
 (319) 643-7136



**Practical Computer Technologies (PCT)** announced today the release of its newly developed PRACTIDISK line of unique auxiliary micro and floppy drive subsystems as a sensible upgrade for all PC-XT and AT (286/386) microcomputers. PRACTIDISK provides vital capabilities with straightforward compatibility in data-media interchange. This advanced add-on subsystem offers a breakthrough with the most convenient and economical solution for migrating software between current and newer generation PC diskettes — of all types.

For more information, contact:  
 Practical Office Systems, Inc.  
 Mindy Schwarcz  
 3972 Walnut Street  
 Fairfax VA 22030  
 (703) 385-3332





## Version Change for PC-Write

Dear HUG:

Thanks for the favorable comments about our word processor, PC-Write, in your December 1988 article, "A Really Fast Word Processor for the Zenith 171." We appreciate the work the writer, Daveed Dov Shachar, went to in telling his readers how to set PC-Write up for the Zenith Z-171 screen and keyboard and for a RAM disk.

However, there was one small misstatement in the article. Quicksoft is no longer selling Version 2.6 or 2.71 of PC-Write; these have been superseded by our latest version, PC-Write 3.0. Version 3.0, among other improvements, breaks the 60K file limit and allows users to edit multiple columns of text on screen and in the file.

Sincerely,  
Miriam Harline  
Public Relations/Communications  
Quicksoft  
219 First Avenue N #224  
Seattle, WA 98109

## Too Many Keyboards

Dear Editor:

I got my November issue of REMark yesterday and I just had to tell you how pleased I was with it, especially the two articles on computer hardware modifications. My son and I both have Z-148s which we have modified. He put the Premier Technologies board in his and has added an Orchid Designer video card with which he runs a Thompson Ultra Sync color monitor. He uses the other slot for a 20 Meg Plus Development Hard Card. He has retained the two 5.25" floppy drives, while replacing the 8088 with an NEC V-20 and has added two additional banks of 256k RAM chips.

My unit has the NEC V-20 and one additional bank of 256k RAM. I installed the ZA-141-1 and a Seagate ST-225 drive. The drive controller is, I believe, a Taiwanese copy of a Western Digital design. The hard drive replaced one of the floppy drives which then became drive B: in my ZW-241. The '148 now follows the usual pattern of timing out the floppy drive before booting from the hard drive. I suppose this is very much like the final ZW-148 configuration, though I've not seen one. My monitor is a ZVM-1220A. The '148 (which I own personally) serves as my office computer and the ZW-241 is my home computer.

We are to get new workstations at the office, so the '148 will soon be dismantled for parts for a project I have going at home. I have found that Brooklyn

Bridge has a unique utility that permits one to run a program from a remote computer. I have been interested for some time in trying to learn what I could do to simulate parallel processing with MS-DOS machines. About a year ago, I acquired a discontinued product from Alloy Computer Products called a BiTurbo board. This is essentially a second computer (NEC V-20) with a megabyte of memory and a COM2 port that resides on the ZW-241 buss and is activated with a device driver that can be invoked at boot time or later as desired. The communications/networking program consumes about 160 K of RAM on the host machine. On my ZW-241, at least, the host must be running MS-DOS 3.1. A special version of DOS 3.1 supplied by Alloy is loaded on the BiTurbo board when it is invoked. The BiTurbo board uses about 671 K of its RAM for programs and 256K as a cache. I don't know what it does with the rest of its memory.

The Alloy scheme requires a hard disk assigned the letter C: and temporarily partitions the free space on the disk equally between the two processors. Once the slave processor has been activated, the display and keyboard are switched between the host and slave by the "hot key" combination <alt .>. This has worked very well, but there are two problems with it that I have not been able to resolve as yet: (1) I have not been able to come up with a way to invoke the slave processor from a batch file. For example, I would like to have the host process write a file and then give a signal to the slave that the file has been written. When it receives this signal, the slave would begin its processing of the recently-written host file. I now do this by having the program which is executing write a message to the screen when the file has been written and I then invoke the slave manually. (2) The BiTurbo board can only work with CGA or Hercules (or MDA) adapters. It uses host memory in the region 0A040 to 0AFFH to communicate with the slave. An Everex VGA board running as primary video controller — even with a 1240 monitor attached to its digital port and the card configured to run a monochrome monitor — causes the whole system to lock up (requiring a cold reboot) when one tries to set the date and time of the slave system after it's been invoked. If the VGA board is designated as the secondary video controller, however, and a Z-329 connected to the 1240 monitor is designated the primary video controller, everything works fine. Video control is then never switched to the VGA monitor, a 1490, during the BiTurbo session. When the BiTurbo session is terminated, video can be switched by software to the secondary controller.

The BiTurbo board is equipped with an 8087 and actually has a higher processing speed in the PC Magazine floating

point benchmark than the 80287 in the ZW-241.

I have installed a Seagate ST-251 as drive D: in the system and partitioned it as one bootable drive of 11 MB and a non-bootable drive of 32 MB. The bootable partition is invoked by the Ctrl Alt Ins BW1 sequence and becomes drive C: when the system comes up. The other partition is not reachable and the original drive C: becomes drive D:. During normal boot (with DOS 3.21), the smaller partition on the ST-251 is not reachable and the larger one is drive D:. OS/2, however, recognizes both partitions of the ST-251. I have version 2.3B of the system ROM installed.

The machine also has a fully populated Z-405 memory card; a controller card for a pair of external 8" floppy disk drives; a Compti-Card II controller for an external 3.5" 1.44 Meg floppy; an additional I/O card with two serial ports, one parallel port, and a game port; and a fan from a Z-386. Since the power supply fan in my ZW-241 blows in, I made the other fan blow in also and dispensed with the Z-386 baffle plate. Pat Swayne's suggestion to change the direction of the power supply fan rotation for better cooling tells me that the heat from the power supply, PLUS the heat from the CPU and I/O boards has little place to go. In my system, I left off one card bracket to permit the third serial port, the game port, and the 3.5" drive cables to exit the box. In addition, I have placed the system box on a stand which has a keyboard drawer underneath so that the cooling holes in the front of the machine are not blocked by the keyboard. I believe the additional exhaust area thus available will save me the trouble of reversing the power supply fan and installing the auxiliary fan baffle plate. This plate's design seriously offends my professorial sensibilities. (Fluid Mechanics is among the courses I teach.)

If you count the number of cards installed, you'll find I have no room for additional XT-compatible cards. I had access to a 6-slot expansion chassis whose transmitter-receiver cards were beginning to go. It has a nice size power supply and mounts the cards horizontally in a vertical stack. I plan to turn it into an XT clone and to use the keyboard, disk drives and hard drive controller from my '148 with it. The drives will go in the drive chassis that came with the Z-217 upgrade kit for my Z-100. The two together make a nice, compact, vertical package. Perhaps I'll use some other parts from the '148, as well. I will then use Brooklyn Bridge as the communications medium between the two machines. It will be interesting to see if I can get the baud rate at which the two machines will talk to each other to go over 19,200. I had to use that rate between the '148 and an eaZy pc.

There is also a possibility that I can



## Zenith Lap Top Enhancements

- **One Megabyte EMS memory card**  
Available for Supersport 286 and Turbosport 386
- **Four Megabyte EMS memory card**  
For Supersport 286
- **SCSI small computer system interface card**  
Supersport 286
- **ARC net LAN card**  
Supersport 286
- **Two Megabyte CMOS RAM Disk**  
For Z181

All cards fit internally.  
One year warranty and 30 day return.

### For Information:

**CALL 714-540-1174**  
**FAX 714-540-1023**  
**or Write**

Zenith is a trademark of  
Zenith Data Systems

**Ai** AMERICAN  
CRYPTRONICS INC.

1580 Corporate Dr., Suite 123  
Costa Mesa, California 92626

Reader Service #101

lay hands upon an early Z-386. I would stack that one on top of the '241 system box. My problem is what to do about all the extraneous keyboards. I can conceive of running multiple jobs (parallel processing), but I never developed the hand-eye and eye-foot coordination required to play a multi-manual organ with foot pedals. In the present case, I just don't have the desktop space available for three keyboards. Thus, I wonder if there is not some way to operate two machines from one keyboard? Currently, on bootup, the BIOS checks to see if a keyboard is connected and stops the machine if it isn't. I don't know whether this checking is continuous. If not, one might use a switch to boot first one machine and then switch the leads to boot the other. Perhaps the switching transient on the signal line might cause the systems to crash? Anyway, I'd be interested if any of your readers have a suggestion about how to implement such a feat. I realize that the XT class uses a two byte key code, while the AT class uses a three byte key code, but some keyboards autosense which system they are connected to. (You have to manually switch the Zenith 101-key keyboard, I've learned).

Let me pass along one final tidbit. We have a pair of Z-151s and a pair of Z-158s in our departmental offices which we wanted to upgrade by adding additional

memory and 30 meg hard drives. As many of your readers know, when one works for a state institution, one is constrained by rather rigid purchasing policies. Our 30 Meg hard drives turned out to be Seagate ST-238s in controller/drive kits supplied by Everex. We found that we could not get the machines to boot from the hard drive with Zenith MS-DOS 3.21. Reformatting the hard drives under PC-DOS 3.2 allowed everything to work properly.

When I called Zenith customer service in Chicago, I was told that Zenith had never supported any RLL drives so they had no suggestion to offer other than what we already suspected we would have to do: Backup the drive to floppies and reformat using PC-DOS 3.2.

Keep up the good work!

Frederick O. Smetana, Professor  
Mechanical and Aerospace Engineering  
North Carolina State University  
Raleigh, NC 27695-7910

\*

Continued from Page 38

III Plus, though I haven't fully explored that capability. I do know, however, that according to the programs' specs, XyWrite III Plus supports more printers than the others.

I also know that writers can be a quirky bunch whose work is influenced, in varying degrees, by the efficacy of their tools. With the SuperSport, XyWrite and my other peripherals, I feel faster, better-prepared and more comfortable. If I can only get my jump shot to work as smoothly, I'll really mean business during my next journey through the NBA.

### Products Discussed

SuperSport Portable Computer  
(Model 20; list: \$3,599)  
2400/1200/300 bps internal modem (list: \$549)  
Zenith Data Systems  
St. Joseph, MI 49085

XyWrite III Plus Version 3.53 (list: \$445)  
XyQuest  
44 Manning Road  
Billerica, MA 01821  
(617) 671-0888

Falcon (list: \$45)  
Spectrum HoloByte  
2061 Challenger Drive  
Alameda, CA 94501  
(415) 522-3584

\*



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



# 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!

The HUG-386 and HUG-386-C are upgrade kits that let you upgrade your H-241 or H-248 series computers up to a full H-386. Now, how do you determine which upgrade kit to buy? The H-386-C includes a dual winchester/floppy controller, while the H-386 does not include any disk controller. Since the old H-241 controller is not '386 compatible, you'll probably want the "C" model if you're upgrading a '241. If you're upgrading a '248, your decision will depend on whether you need a new dual controller or not.

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.

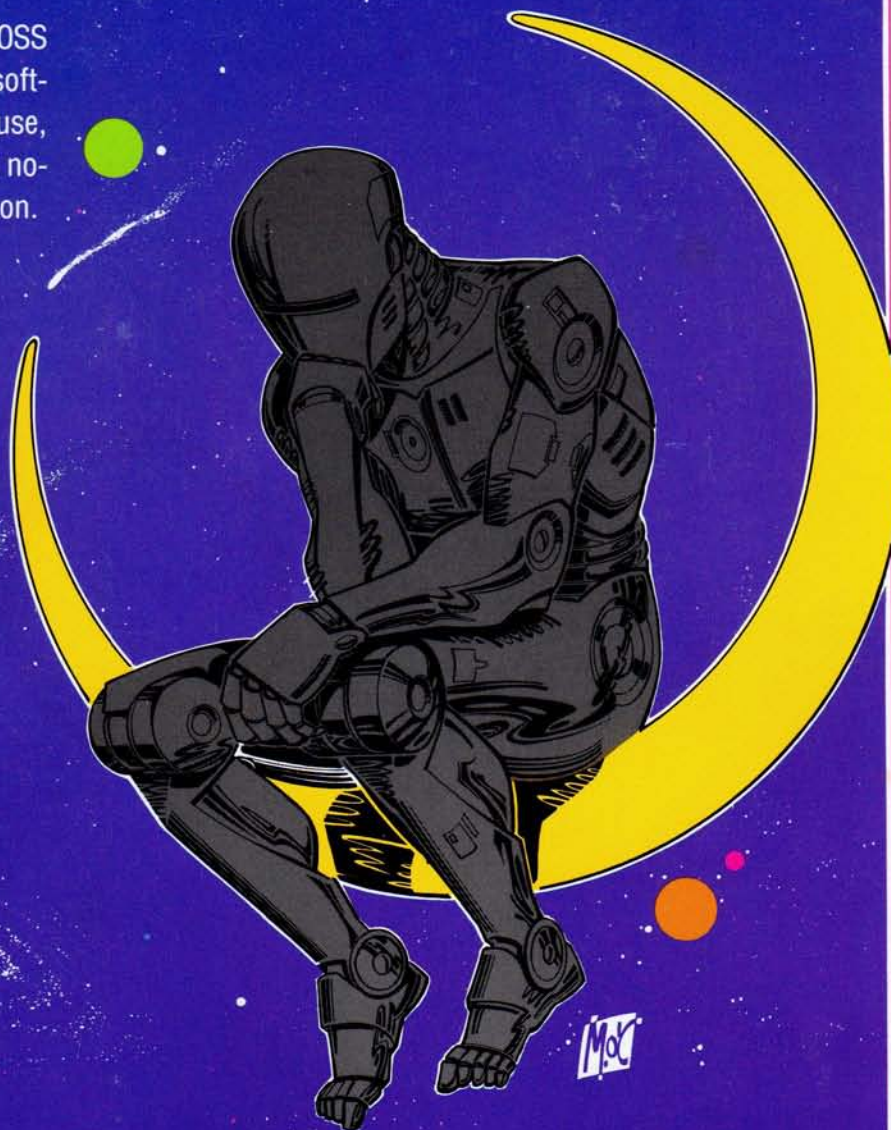
### Heath/Zenith Computer Store Sales

- 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 soft-  
ware is too complex to use,  
get *HUGMCP* for the no-  
hassle modem connection.



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

BULK RATE  
U.S. Postage  
PAID  
Heath Users' Group