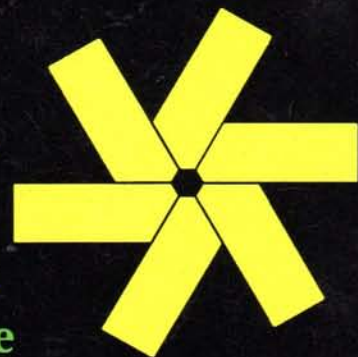


REMark

March 1991



The Official Zenith Data Systems Users Magazine

*Getting to Know Zenith Data Systems'
Bulletin Board System*

Page 4

*Breaking the Churlish
Bounds of DOS*

Page 35

*Getting Started With ...
Mice and Pens*

Page 43





HADES II

It's HOTTER than ever! Jam-packed with new features, HADES II still remains the easiest-to-use disk editor ever! Just look at some of the features:

- Sector Display/Editing
- Sector HEX/ASCII String Search
- File Display/Editing
- Physical and Logical Cluster Display
- File HEX/ASCII String Search
- Drive Parameter Display
- 512 MegaByte Drive Size Limit
- File Attribute Display/Edit
- Automatic Erased File Recovery
- Manual Rebuild File Recovery
- Works with Headerless MS-DOS Disks
- PC-Compatible or H/Z-100

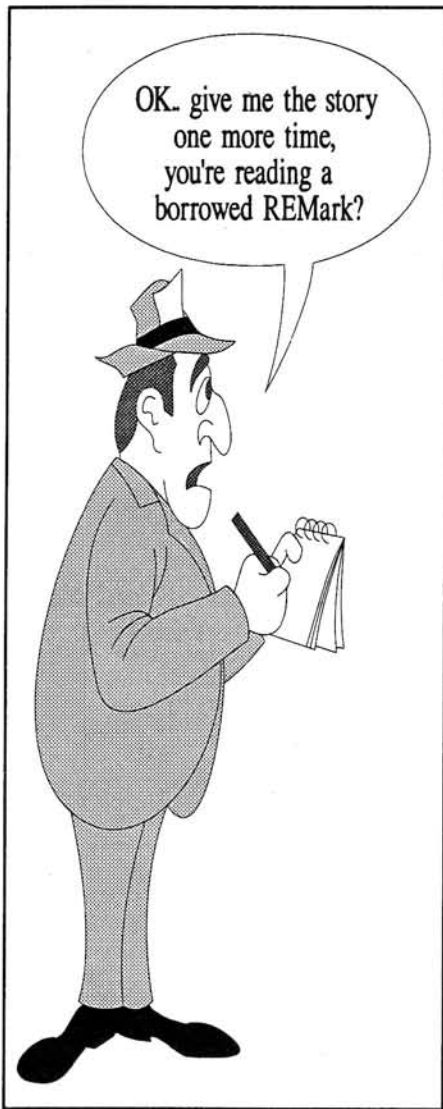
HADES II is still only \$40, and original HADES owners can upgrade their distribution disk for only \$15. Call HUG today at: (616) 982-3463.

REMark

March 1991



The Official Zenith Data Systems Computer Users Magazine



PC Compatibles

Impressive New Features <i>Edwin C. Wiggins</i>	5
The World of WP50 and Its Wonders <i>Salli Bracket</i>	7
Database Publishing with dBASE IV and a LaserJet Printer <i>Alan R. Neibauer</i>	17
Breaking the Churlish Bounds of DOS - Part 1 <i>David W. Lind</i>	35
Getting Started With . . . Mice and Pens <i>Alan R. Neibauer</i>	43
Minimum Effort Hard Drive Backup <i>Ron Siebers</i>	47

H/Z-100 & PC Compatibles

On the Leading Edge <i>William M. Adney</i>	11
ENABLE Revisited - Part 5 <i>George P. Elwood</i>	21

General

Introduction to C++ <i>Lynwood H. Wilson</i>	25
dBASE III - Part 11 <i>D. R. Cool</i>	29
Port-O-Call: COM1 <i>Laura White</i>	33

Resources

Software Price List	2
ZLink-COM1 BBS	4
Classified Ads	10

Advertising Placement

Reader Service No.		Page No.
250	B.U.D. Unlimited	32
104	FBE Research Co., Inc.	31
117	Payload	24
193	QuikData, Inc.	42
149	W.S. Electronics	34

Software

Managing Editor
Jim Buszkiewicz
(616) 982-3837

Software Engineer
Pat Swayne
(616) 982-3463

Production Coordinator
Lori Lerch
(616) 982-3794

Secretary
Lisa Cobb
(616) 982-3463

COM1 Bulletin Board
(616) 982-3956
(Modem Only)

ZUC
Software Orders
(616) 982-3463

Contributing Editor
William M. Adney

Printer
Imperial Printing
St. Joseph, MI

Contributing Editor
Robert C. Brenner

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

	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 ZUG Product List for availability of bound volumes of past issues. Requests for magazines mailed to foreign countries should specify mailing method and include appropriate, additional cost.

Send Payment to: Zenith Users' Group
P.O. Box 217
Benton Harbor, MI 49023-0217
(616) 982-3463

Although it is a policy to check material placed in REMark for accuracy, ZUG 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 Zenith Data Systems Computer Centers.

ZUG is provided as a service to its members for the purpose of fostering the exchange of ideas to enhance their usage of Zenith Data Systems equipment. As such, little or no evaluation of the programs or products advertised in REMark, the Software Catalog, or other ZUG publications is performed by Zenith Data Systems, in general, and Zenith Users' Group, in particular. The prospective user is hereby put on notice that the programs may contain faults, the consequence of which Zenith Data Systems, in general, and ZUG, in particular, can not 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 Zenith Users' Group, St. Joseph, Michigan.

Copyright (c) 1991, Zenith Users' Group

PRODUCT NAME	PART NUMBER	OPERATING SYSTEM	DESCRIPTION	PRICE
H8 - H/Z-89/90				
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
AUTOFIL (Z80 ONLY)	885-1110	HDOS	DBMS	30.00
BHBASIC SUPPORT PKG	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	CHKBK 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 PKG	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 SECT SUPPORT PKG	885-1121	HDOS	UTILITY	30.00
HDOS PROG. HELPER	885-8017	HDOS	UTILITY	16.00
HOME FINANCE	885-1070	HDOS	BUSINESS	18.00
HUG DISK DUP UTILITY	885-1217-[37]	CPM	UTILITY	20.00
HUG SOFTWARE CATALOG	885-4500	VARIOUS	PROD TO 1982	9.75
HUGMAN & MOVIE ANIM	885-1124	HDOS	ENTERTAINMENT	20.00
INFO SYS AND TEL. & MAIL SYS	885-1108-[37]	HDOS	DBMS	30.00
LOGBOOK	885-1107-[37]	HDOS	AMATEUR RADIO	30.00
MAGBASE	885-1249-[37]	CPM	MAGAZINE DB	25.00
MISCELLANEOUS UTILITIES	885-1089-[37]	HDOS	UTILITY	20.00
MORSE CODE TRANSCEIVER	885-8016	HDOS	AMATEUR RADIO	20.00
MORSE CODE TRANSCEIVER	885-8031-[37]	CPM	AMATEUR RADIO	20.00
PAGE EDITOR	885-1079-[37]	HDOS	UTILITY	25.00
PROGRAMS FOR PRINTERS	885-1082	HDOS	UTILITY	20.00
REMARK VOL 1 ISSUES 1-13	885-4001	N/A	1978 TO DEC '80	20.00
RUNOFF	885-1025	HDOS	TEXT PROC	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 PKG	885-1127-[37]	HDOS	UTILITY	20.00
STUDENT'S STATISTICS PKG	885-8021	HDOS	EDUCATION	20.00
SUBMIT (Z80 ONLY)	885-8006	HDOS	UTILITY	20.00
TERM & HTOC	885-1207-[37]	CPM	COMMUN & UTIL	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	25.00
Z80 ASSEMBLER	885-1078-[37]	HDOS	UTILITY	20.00
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	CHKBK SOFTWARE	25.00
COPYDOS	885-1235-[37]	CPM	UTILITY	20.00
DISK DUMP & EDIT UTILITY	885-1225-[37]	CPM	UTILITY	30.00
DUNGEONS & DRAGONS	885-1209-[37]	CPM	GAMES	20.00
FAST ACTION GAMES	885-1228-[37]	CPM	GAME	20.00
FUN DISK I	885-1236-[37]	CPM	GAMES	20.00
FUN DISK II	885-1248-[37]	CPM	GAMES	35.00
GAMES DISK	885-1206-[37]	CPM	GAMES	20.00
GRADE	885-8036-[37]	CPM	GRADE BOOK	20.00
HRUN	885-1223-[37]	CPM	HDOS EMULATOR	40.00
HUG FILE MANAGER & UTILITIES	885-1246-[37]	CPM	UTILITY	20.00
HUG SOFTWARE CAT UPDT #1	885-4501	VARIOUS	PROD 1983 TO 1985	9.75
KEYMAP CPM-80	885-1230-[37]	CPM	UTILITY	20.00
MBASIC PAYROLL	885-1218-[37]	CPM	BUSINESS	60.00
NAVPROGSEVEN	885-1219-[37]	CPM	FLIGHT UTILITY	20.00
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
X-REFERENCE UTIL FOR MBASIC	885-1231-[37]	CPM	UTILITY	20.00
ZTERM	885-3003-[37]	CPM	COMMUNICATIONS	20.00

Price List

PRODUCT NAME	PART NUMBER	OPERATING SYSTEM	DESCRIPTION	PRICE
H/Z-100 (Not PC) Only				
CARDCAT	885-3021-37	MSDOS	BUSINESS	20.00
CHEAPCALC	885-3006-37	MSDOS	UTILITY	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
DUNGN & DRAGONS (ZBASIC)	885-3009-37	MSDOS	GAME	20.00
ETCHDUMP	885-3005-37	MSDOS	UTILITY	20.00
EZPLOT II	885-3049-37	MSDOS	PRINTER PLOT UTIL	25.00
GAMES (ZBASIC)	885-3011-37	MSDOS	GAMES	20.00
GAMES CONTEST PACKAGE	885-3017-37	MSDOS	GAMES	25.00
GAMES PACKAGE II	885-3044-37	MSDOS	GAMES	25.00
GRAPHIC GAMES (ZBASIC)	885-3004-37	MSDOS	GAMES	20.00
GRAPHICS	885-3031-37	MSDOS	UTILITY	20.00
HELPSCREEN	885-3039-37	MSDOS	UTILITY	20.00
HUG BKGRD 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
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	ATRONOMY UTILITY	20.00
SMALL-C COMPILER	885-3026-37	MSDOS	LANGUAGE	30.00
SPELL5	885-3035-37	MSDOS	SPELLING CHECKER	20.00
SPREADSHEET CONTEST PKG	885-3018-37	MSDOS	VARIOUS SPRDST	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
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
BACKGRD PRINT SPOOLER	885-3029	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 II	885-3040	MSDOS	UTILITY	40.00
HEPCAT	885-3045	MSDOS	UTILITY	35.00
HUG EDITOR	885-3012	MSDOS	TEXT PROCESSOR	20.00
HUG MENU SYSTEM	885-3020	MSDOS	UTILITY	20.00
HUG SOFTWARE CAT UPD #1	885-4501	MSDOS	PROD 1983 - 1985	9.75
HUGMCP	885-3033	MSDOS	COMMUNICATION	40.00
ICT 8080 - 8088 TRANSLATOR	885-3024	MSDOS	UTILITY	20.00
MAGBASE	885-3050	VARIOUS	MAG DATABASE	25.00
MATT	885-8045	MSDOS	MATRIX UTILITY	20.00
MISCELLANEOUS UTILITIES	885-3025	MSDOS	UTILITIES	20.00
PS' PC & Z100 UTILITIES	885-3052	MSDOS	UTILITIES	20.00
REMARK VOL 8 ISSUES 84-95	885-4008	N/A	1987	25.00
REMARK VOL 9 ISSUES 96-107	885-4009	N/A	1988	25.00
REMARK VOL 10 ISSUES 108-119	885-4010	N/A	1989	25.00
REMARK VOL 11 ISSUES 120-131	885-4011	N/A	1990	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				
CARDCAT	885-6006	MSDOS	CAT SYSTEM	20.00
CHEAPCALC	885-6004	MSDOS	SPREADSHEET	20.00
CLAVIER	885-6016	MSDOS	ENTERTAINMENT	20.00
CP/EMULATOR II & ZEMULATOR	885-6002	MSDOS	CPM & Z100 EMUL	20.00
DUNGEONS & DRAGONS	885-6007	MSDOS	GAME	20.00
EZPLOT II	885-6013	MSDOS	PRINTER PLOT UTIL	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
LAPTOP UTILITIES	885-6014	MSDOS	UTILITIES	20.00
PS' PC UTILITIES	885-6011	MSDOS	UTILITIES	20.00
POWERING UP	885-4604	N/A	GUIDE TO USING PCs	12.00
SCREEN SAVER PLUS	885-6009	MSDOS	UTILITIES	20.00
SKYVIEWS	885-6005	MSDOS	ASTRONOMY UTIL	20.00
TCSPELL	885-8044	MSDOS	SPELLING CHECKER	20.00
ULTRA RTTY	885-6012	MSDOS	AMATEUR RADIO	20.00
YAUD (YET ANOTHER UTIL DSK)	885-6015	MSDOS	UTILITIES	20.00

This Software Price List contains all products available for sale. For a detailed abstract of these products, refer to the Software Catalog, Software Catalog Update #1, or previous issues of REMark.

Now Available!

ZUG software is now available on 2" disks. Just put a "90" at the end of the part number (i.e., 885-6014-90). Also add \$3.00 to the purchase price of the software (i.e., \$20.00 + \$3.00 = \$23.00).

LAPTOP OWNERS . . . don't feel left out! All of ZUG's MS-DOS software is available on 3-1/2" micro-floppies too! When ordering, just add a "-80" to the 7-digit ZUG 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. They say it's so easy to use, they didn't even need to look at the manual. "It's the only modem software that I use, and I'm in charge of the HUG bulletin board!" says Jim Buszkiewicz. HUGMCP runs on ANY Heath/Zenith computer that's capable of running MS-DOS!

ORDERING INFORMATION

For VISA, MasterCard, and American Express phone orders, telephone the Zenith Users' Group directly at (616) 982-3463. Have the part number(s), description(s), 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: Zenith Users' Group, P.O. Box 217, Benton Harbor, MI 49023-0217. VISA, MasterCard and American Express require minimum \$10.00 order. No C.O.D.s accepted.

Questions regarding your subscription? Call Lisa Cobb at (616) 982-3463.

Zenith Data Systems ZLink-COM1 Bulletin Board System (BBS)

WHAT IS ZLink-COM1?

ZLink-COM1 is a computer-accessed electronic message system. You get automatic access to the board when you subscribe to REMark. It lets you "post" messages to other users, browse through and download megabytes of programs and files, and scan For Sale items, much like bulletin boards at your work place or school.

EASY TO LEARN

ZLink-COM1 is designed for the novice computer user. It has full-featured, step-by-step menus to guide you through its many separate divisions.

Log-in on ZLink-COM1 is automatic. All you do is enter your first name, last name, and ID number when asked. First-time users will also be asked some simple questions about their monitors; number of lines per screen and characters per line, etc. The board will compare your entries with its validated member database, check for messages addressed to you, and then display the Top Menu.

From the Top Menu you can elect to review (I)nformation and Bulletins pertaining to the board, move to the (D)atabase to download/upload files, or enter the (K)eypboard Shopping Club. Other selections let you check for (N)etwork mail or messages in the (M)essage system, or start a live discussion with others using the board at that time in the (C)onference area.

Once you log on, we suggest you select the (I)nformation area to learn about the board. This also lets you see any special bulletins pertaining to the board or group. If you get stumped while looking around, (H)elp is only a keystroke away from most areas. And you can always CTRL-C to return to the last menu or jump to the (T)op Menu.

ACCESS TO USED EQUIPMENT

One of the main features of ZLink-COM1 is the Key-board Shopping Club. This menu gives you access to the (B)argain Centre and (S)oftware and Subscription Sales areas. The Bargain Centre provides you with access to new and/or used Zenith Data Systems equipment at extremely economical prices. A list of available units is periodically posted, and members have a first-come, first-served chance at all items. Ordering something is as easy as entering the item number, Visa, MasterCard, or AmEx card information, and membership data. Once completed, your selection is reserved and confirmed on the spot.

Software is ordered in the same way from the (S)oftware Sales menu. Here you can order programs written to make computing easier for you. There are selections for PC compatibles as well as earlier Heath/Zenith 8- and

16-bit computer systems, including the H/Z-100 series non-PC computers.

SOFTWARE TO GO

The ZLink-COM1 (D)atabase selection takes you to Shareware by the Meg. Shareware, or "try before you buy" software, occupies a major amount of hard disk space on the board. There are, at any give time, hundreds of disks worth of software available. The files are divided into topics such as Amateur Radio, Utilities, CAD, Games, and others. Each topic has a separate directory listing that tells you what the various programs do. If you see a program that appeals, you can download it to your system using one of the popular protocols supported by the board: XMODEM, YMODEM/YMODEM-g (Batch), KERMIT, or SEALink.

ZENITH DATA SYSTEMS SOLUTIONS

One entire section of the Database is devoted to information about Zenith Data Systems equipment. This area features suggestions on hardware/software compatibility, tips on using your computer, and files that relate the experiences of other users. Technical Notes, updates on Zenith Data System software, and configuration data for older models can also be found here. And the best part is that all of these files are downloadable by anyone; you don't have to subscribe to REMark to get these files.

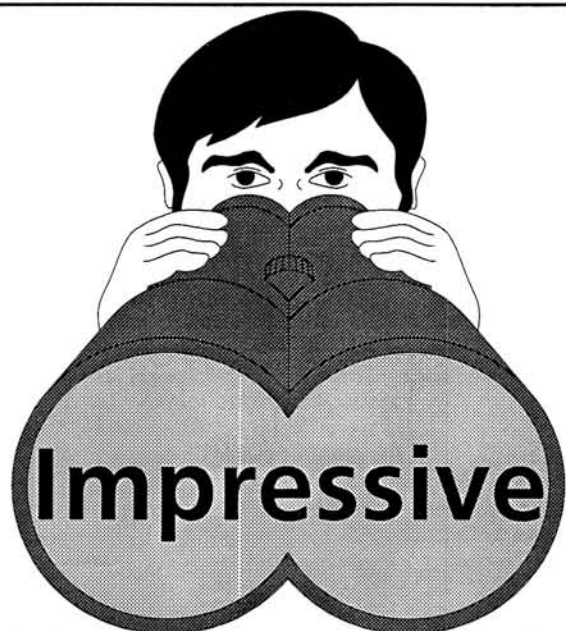
Have a question about your model ZXX-NNNN computer, or a bug in a BASIC program? You will likely get help from one of the thousands of users who regularly monitor the Message board. Here you can ask questions of other users, post items for sale or wanted, correspond with other members, and leave a private message for someone, including the sysop.

HARDWARE AND SOFTWARE REQUIREMENTS

To access ZLink-COM1, you need a PC compatible computer, a modem, and some type of communications or modem control program. The software should be configured for 8-bits, no parity, one stop bit (8N1), and capable of 300, 1200, or 2400 baud.

Try us; you'll like us! You can, and without subscribing first. Just set your modem, dial (616) 982-3956 and log on as a "New User." You'll get a chance to look over all board areas and download those Zenith Data Systems specific files mentioned earlier. You cannot, however, download any other software or order something from the Bargain Centre until you become a subscriber.

If you have any questions and want to talk to a human, call (616) 982-3463 Monday through Friday, 8:00 AM to 4:30 PM, Eastern Time.



WordPerfect Version 5.1

New Features

Edwin G. Wiggins
13 Clare Drive
East Northport, NY 11731

Introduction

Early, WordPerfect Corp. began shipping version 5.1 of its flagship product. New features include: context sensitive help, right and center justification, long document names, FOR and WHILE macro commands, relative (to the margin) tabs, spreadsheet linking, more flexible outlining and page numbering, mouse support and the equation builder. Three of these features: mouse support, spreadsheet hot links and the equation builder, are discussed in detail below. They represent a major enhancement of the product.

Alas "there ain't no free lunch." The price of the enhancements is slower operation. Version 5.1 takes more time to load than 5.0 did. When you issue the print command, there's a longer delay before printing actually begins. The long filenames feature is so slow on my machine (an ancient Z-148) that I can't bear to use it. Perhaps on a fast machine it's usable.

Despite the slower performance, version 5.1 is well worth the price of the upgrade.

Mouse Support

One of the few serious criticisms of WordPerfect has been the lack of an intuitive user interface. Although I have been using version 5.0 for nearly two years, I still have to stare at my function key template for most menu activities. Now users have an alternative. Press the right mouse button (on a Microsoft Mouse) and a menu bar appears across the top line of the screen. The choices are:

File Edit Search Layout Mark Tools Font Graphics Help

Note that features are not arranged as they are on the function keys.

Click on one of these menu choices with the left mouse button (on a Microsoft Mouse) and you get a drop down menu of

features. If you click on Search, you get this menu of choices:

Forward
Backward
Next
Previous

Replace

Extended

Goto

Click on Forward and you get the familiar (to WordPerfect users)

-> Srch:

prompt at the bottom of the screen.

The mouse can also be used to move about the screen and to quickly mark blocks of text to be moved, deleted, etc. As soon as you move the mouse, the usual small rectangle mouse cursor appears. Move this cursor to any point on the screen and click the left button to move the typing cursor to that point. Unfortunately there are no scroll bars, so you can't use the mouse to scroll the screen.

Blocks can be marked easily with the mouse. Move the mouse cursor to the beginning of the block, press and hold the left button and drag the mouse cursor to the end of the block. The block is highlighted in reverse video as you go. On my computer the reverse video lags the movement of the mouse cursor annoyingly. When marking blocks, the screen does scroll when you push the mouse cursor against the top or bottom of the display. This means that blocks that extend past the top or bottom of the screen can be marked with the mouse.

As soon as you press a key on the keyboard, the mouse cursor disappears. This prevents confusion on the screen display. If you move the mouse, its cursor reappears instantly.

I've been using version 5.1 for over six

months, and I seem to stay with the function keys for menu activities. Other users, particularly those new to WordPerfect, may prefer the mouse menus.

Spreadsheet Hot Links

WordPerfect itself has some spreadsheet-like capability. In version 5.0 this was called "Math/Columns." This capability has been improved and simplified in version 5.1, and it has been renamed "Tables." Even the enhanced capability is not nearly in the same league with a real spreadsheet.

Version 5.0 could import a spreadsheet print file; it could not import an active spreadsheet file. The process of creating and importing the print file was tedious at best. Version 5.1 can hot link to a spreadsheet file on disk. Links can be permanent or one time only. If the link is permanent, the spreadsheet in the WordPerfect document is updated each time the document is loaded. If the spreadsheet file has been modified since the last loading of the document, the document is updated.

Here's how the process works. First use your favorite spreadsheet program to create the spreadsheet file. It must be saved to disk in WKS (Lotus release 1A), WK1 (Lotus release 2), WKQ (Quattro), PlanPerfect or Microsoft Excell format. Switch to WordPerfect and move the cursor to the desired location in your document. Initiate the link by pressing <CTRL F5> (or click on the File choice in the mouse menu).

Select Spreadsheet from the menu at the bottom of the screen (or click on Text In and then Spreadsheet in the mouse menu). Select Create link (same on mouse menu). At this point you get the same full screen menu whether you were using the mouse or the keyboard. Select Filename from this full screen menu and type the name of the spreadsheet file. Next select Perform link from the menu, and WordPerfect does the

rest. The table below is a spreadsheet that is hot linked into this document.

WIDGET SALES	
JAN	\$100.00
FEB	\$200.00
MAR	\$300.00
TOTAL	\$600.00

The table above can be edited just like anything else in the document. I can change JAN to JANUARY. I can also change the January sales figure to \$250, but the total will NOT change. Be careful when editing!

Link options can be set to import the latest version of the spreadsheet only on command or automatically each time the document is retrieved. Regardless of how this option is set, all spreadsheet links in the document can be updated at any time. This can be helpful if you have edited the table and don't like what you have done.

The native files of my favorite spreadsheet, Quattro Pro, are WQ1 files. WordPerfect can not link to WQ1 files, so I have to instruct Quattro Pro to save the spreadsheet as a WK1 file if I want to import it into WordPerfect.

Equation Builder

My favorite new feature in version 5.1 is the equation builder. Since I'm an engineering professor, I have frequent need to include Greek letters and mathematical symbols in my documents. I tried a scientific word processing program, but it was too cumbersome to use for correspondence. So I used two different word processors: one for scientific work and the other for routine documents. The equation builder feature in WordPerfect 5.1 enables me to use one program for everything.

The equation builder is accessed by typing <ALT F9> and selecting 6 or E from the menu at the bottom of the screen. From the next menu, select 1 or C to create a new equation. Select 2 or E to edit an existing one. Since I want to create a new equation I press C. The following full screen menu appears:

Definition: Equation

- 1 - Filename
- 2 - Contents Equation
- 3 - Caption
- 4 - Anchor Type Paragraph
- 5 - Vertical Position 0"
- 6 - Horizontal Position Full
- 7 - Size 6.5" wide x 0.333" (high)
- 8 - Wrap Text Around Box Yes
- 9 - Edit

Selection 1 makes it possible to load a previously saved equation from disk. If you use the same equation frequently, you'll only have to type it once. The default settings in items 2 through 8 are usually satisfactory. If so, you simply press 9 or E.

Note that Edit on this menu includes the creation of new equations.

At last the actual equation builder screen appears. This screen is divided into three windows: the editing window, the display window and the equation palette window. Equations are created by typing in the editing window. Once an equation is created, it can be displayed in the display window exactly as it will appear in print. Commands and symbols used in creating equations appear in the equation palette window. The examples below show how to create a variety of equations ranging from simple to complex.

Example 1 The Quadratic Formula

Remember the quadratic formula from high school algebra? It's almost impossible to write that formula in readable fashion with an ordinary word processor. To create the equation with the equation builder proceed as follows: Activate the equation builder as described above. The cursor will be in the editing window when the equation builder screen appears. Simply type the following:

`x={-b plusminus sqrt{b^2-4ac}} over {2a}.`

Press <F9> and the following appears in the display window:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Although the syntax used to create the equation is fairly natural, several things deserve further explanation. Notice the three keywords: plusminus, sqrt and over. "Plusminus" produces the plus-or-minus sign. "Sqrt" generates the square root symbol, and "over" divides the formula into numerator and denominator.

Braces, "{" and "}", are used to group things together. For instance `sqrt{b^2-4ac}` causes the square root symbol to include all the necessary symbols. Everything between the equals sign and the keyword "over" is enclosed in braces to designate the numerator, and everything following "over" is in braces to designate the denominator.

If the equation in the display window is correct, just press F7 (exit) twice to return to the normal document screen. You will see a box marking the location of the equation, but the equation itself will not be visible. To display the document and the equation, select "View Document" from the Print menu.

Spaces typed in the editing window are ignored when the actual equation is created. That is, "a over b" and "aoverb" produce the same result. The former is much easier to read, however, so use spaces to make your work easier to read later. Although typed spaces are ignored, you can introduce spaces in your equations. A tilde character (~) makes a full size space

and a backward accent (') makes a narrow space.

The first term inside the square root, b^2 , was created by typing "b^2" in the editing window. There are several other ways this could have been done. Typing "b sup 2" will do it. The command "sup" is short for superscript. Another way is to press <F5> to move the cursor to the equation palette, use the arrow keys to move the highlight down to the second selection (SUP or ^) and press <Enter>. This places "SUP" in the editing window and returns the cursor there.

The Command menu appears in the equation palette initially, but there are 7 other menus, including Greek, that can be brought up simply by repeatedly pressing the <PgDn> key when the cursor is in the equation palette window.

Example 2 Trigonometry

Remember this identity from high school trig:

$$\sin^2 \alpha + \cos^2 \alpha = 1$$

That one was produced by typing

$$\sin^2 \alpha + \cos^2 \alpha = 1$$

in the editing window. Note the use of tildes to create spaces in the equation.

Example 3 Calculus

Here's a really wild one. The average energy of a neutron in a nuclear reactor is given by:

$$\bar{E} = \frac{1}{N} \int_0^\infty \frac{2\pi N}{(\pi kT)^{3/2}} E^{3/2} e^{-E/kT} dE$$

I created that by typing the following in the editing window:

$$\overline{E} = \frac{1}{N} \int_0^{\infty} \frac{2\pi N}{(\pi k T)^{3/2}} E^{3/2} e^{-E/kT} dE$$

The keyword "overline" causes the line to appear over the E to the left of the equals sign. The integral with limits is created by "int sub 0 sup inf." "Int" creates the integral sign, sub 0 creates the lower limit, and sup inf creates the upper limit. "Inf" is the keyword for infinity.

There seems to be no limit to the complexity of equations that can be created with the equation editor. The only thing it won't do is solve the darn things.

Symbols in a Line of Text

The equation builder is wonderful for producing whole equations, but sometimes I just need a single Greek letter in the middle of a line of normal text. In the statement of a test question I may want to say "Calculate the value of ..." The Greek letter π was produced by the following sequence of commands:

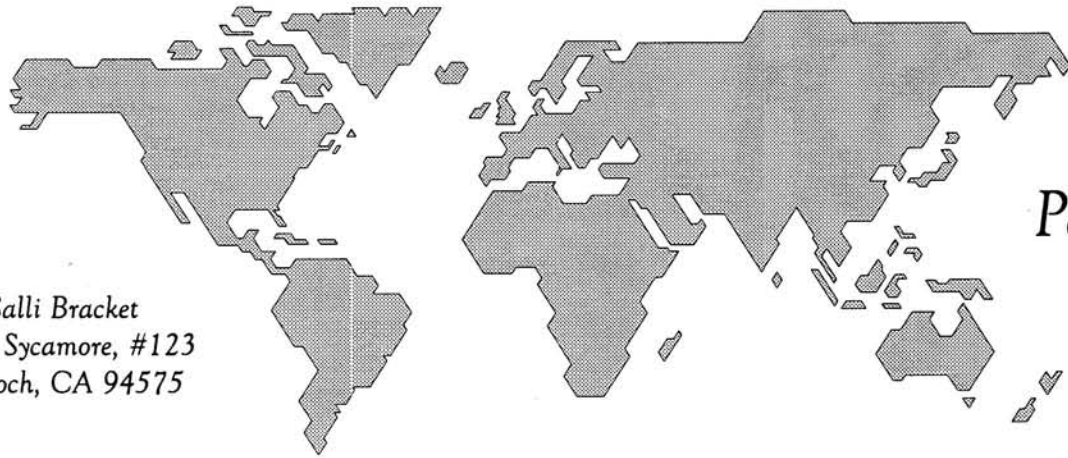
<CTRL V> 8.17 <Enter>

When you press <CTRL V>, the following prompt appears at the bottom of the screen:

Continued on Page 34

The World of WP50 and Its Wonders

(4.2 and 5.1 will be considered)



Part 3

Salli Bracket
2201 Sycamore, #123
Antioch, CA 94575

We now continue the fascinating world of Wordperfect. In my first article (September 1990), we learned basic editing and cursor movements. In the second article (November 1990), we studied merges.

Now let's move to the world of STYLES. WONDER #4: WordPerfect gives us a chance to collect our common format changes in one place - a style document.

TIP 1: Before starting styles, I suggest creating a directory for styles. Go to DOS (Ctrl F1). Type md \wp50\styles. To return to WordPerfect, type EXIT.

TIP 2: I have found it less confusing to have the left and right margins and tab menus show the position by columns (4.2 does this by default). This will become clearer in further discussions. For now just follow the instructions below. You will change the default of inches to 4.2 units or columns.

Press Ctrl F1 (Setup)
Press U (in WP51 press E then U)
Press D(isplay), U
Press S(tatus line), U
Press F7

You now have your original screen back.

Styles are Wordperfect's method of creating specific format changes in your document. Many of you may have already discovered the concept of 'format files'. Format files being a specific file that has special format codes it. For example, suppose you use letterhead a great deal or want to create you own. There are many format changes you need: you need to start on line 9 and end on line 62, you want the letter in a 12 pitch (10 point) font, and today's date to appear automatically. You could have a specific file, called 'letter.fmt' with all those codes. Every time you use letterhead, you would retrieve 'letter.fmt', put in your information, and save it under

a different name.

This works well, but styles give you much more variety of formatting as you will see with the examples below. The advantages to styles are 1) you can turn them on and off, 2) several styles can be put in one document, called a style document, thus taking up less space on your disk and by pressing the style key (ALT F8) (4.2 does not have this feature), you have an immediate list of all your styles, and 3) editing or making additions is easier.

To make this clear, follow me down the rabbit hole with Alice, and we will learn by experimenting with the following examples. First, you will learn how to create individual styles, using paired and open styles. Then you will put them all together into a 'style document.' A 'style document' is a group of individual styles. When you clear the screen (F7,n,n), the style document is cleared from the style menu. When you retrieve the 'style document' (ONLY IN THE STYLE MENU) it lists the individual styles in that style document. You then choose (or turn on) the style you wish to use.

1. Letterhead - The pitch is 12, the first typing line is 9 (therefore the top margin is 8), the left/right margins are 1" (therefore 12), the last typing line is 62 (therefore the bottom margin is 3), and the date is always today's date. OPEN
2. Indented Quotes - The margins are indented 5 spaces (one tab set) left and right and the line spacing is single space. There should be a blank space below and above the quote. PAIRED
3. Envelope - Paper Size is envelope, the top/bottom margin is 12/2, the left/right margins are 65/2. These depend on how your printer feeds envelopes. Try these margins, then adjust accord-

ingly. Creating an envelope form will be discussed in further articles.

4. Columns - Two parallel columns definitions with default margins and turned on. More details of columns will be discussed in further articles.

Before starting, make sure you have a blank screen. Press F7,n,n.

To create #1, press ALT F8; the following screen will appear.

Styles Name	Type	Description
-------------	------	-------------

1 On; 2 Off; 3 Create; 4 Edit; 5 Delete; 6 Save; 7 Retrieve; 8 Update 1

Press 'c' for create; the following screen will appear.

Styles: Edit		
1 - Name		
2 - Type		Paired
3 - Description		
4 - Codes		
5 - Enter		HR+

Selection: 0

Explanations of each item on the above screen:

1. This is the actual name of the style, i.e., letterhead.
2. There are two options here. A paired style can be turned on and off by pressing ENTER. An open style is one that is turned off by pressing 'Off' in the Menu. It is on from the point you insert it throughout the rest of the document.
3. This is detailed explanation of your style.
4. This is the actual codes and/or text you need to create your style.
5. This refers to how you want the ENTER y to work in a paired style. You can have it turn the style off and on, just turn the style off, or turn the style on.

Type n(name)
Type letterhead

Press ENTER
 Type t(ype)
 (The options will show at the bottom of your screen.)
 Type o(pen)
 Type d(ocument)
 Type letterhead with automatic date
 Type c(ode)
 Now you have the screen below:

stead of 15.)
Put in Automatic Date:
 Press Alt F6 (Flush Right)
 Press Shift F5, c (date code)
 (Today's date will appear from the system date. If you have not set up the system date, this will be incorrect. To set system date, check the last page.)

Type e(nter)
 Type f(off)
 (This means when you press ENTER the style will go off.)
 Type c(ode)
 Now you have the screen in Figure 1.
 Notice the screen is split into two screens. The top screen shows the cursor above the comment line. The bottom screen shows the [comment] highlighted (this is where the cursor is).

Style: Press EXIT when done

Doc 1 Pg 1 Ln 1 Pos 10

To create the proper codes based on the description above for letterhead.

Choose a 12 Pitch Font:

Press Ctrl F8, f (base font)
 Cursor to your choice and press ENTER.
Note 1: Always put your font change first. WordPerfect will automatically change the left/right margins, etc. to fit the font change.

Change the margins:

Press Shift F8, p(age), m(argin)
 Remember an 8-1/2" X 11" paper has 66 lines.
 The first typing line is 9,
 Top margin of 8
 The last typing line is 62,
 Bottom margin is 4 (66 - 4 = 62)

Therefore:

Type 8
 Press ENTER
 Type 4
 Press ENTER Twice
 Press l(ine), m(argin).
 You are using a 12 pitch font.
 12 pitch means 12 characters (columns) per inch.
 You want a 1 inch left/right margin)

Therefore:

The margins should be 12/12. If not:
 Type 12
 Press ENTER
 Type 12
 Press ENTER
Note 2: WordPerfect's default font is 10 pitch and the tabs are set at every 5 starting at 0 (0, 5, 10, 15...). Using 12 pitch, your margins start at 12. This means your first tab would be 15. To have tabs every 5, you need to change the tab settings. (In WP51 this problem has been corrected; you don't need to change tabs.)
 Press t(ab)
 Use home, left arrow to cursor to 0 column on tab line.
MAKE SURE THE CURSOR IS ON THE 0 COLUMN.
 Press EOL (Ctrl End)
 Type 12, 5 (the first number is the beginning column and the second number is the spacing you wish)
 Press ENTER
 Press F7 twice
 (Now you have tabs every 5 starting at 12 which means your first tab will be 17 in-

Press ENTER 4 times.
 (Your cursor is now set at the right point to start an inside address.)
 You now have all the codes you need for your letterhead style.
 Press Exit (F7) twice. You are back to the original style screen. Your screen should look like this:

Styles		
Name	Type	Description
letterhead	open	letterhead with automatic date

1 On; 2 Off; 3 Create; 4 Edit; 5 Delete; 6 Save; 7 Retrieve; 8 Update 1

Before testing your style you need to save it in a 'style document.'
 Press s(ave)
 Type \wp50\styles\master.sty
 Press ENTER.
 (Notice the extension is 'sty'. This will distinguish all your style documents from other files.)
 Press F7
RETURN TO THE STYLE MENU, (ALT F8)
 Press o(n)
 You will see the date and the cursor 3 lines below. If you type a couple of lines and print this out, you will see all your margins in the proper place. Now you don't have to go through all those changes every time you need to use letterhead. You just **TURNE**D THE STYLE ON. Clear the screen (F7, n, n).
TIP 3: If you have a short letter and want the date down further on the page, you can create two styles, shortletter and longletter. The shortletter style would be identical to the one just created, EXCEPT your top margin would be 12.
 To create #2, indented paragraphs, go to style menu (Alt F8)
 Type c(reate)
 Type n(ame)
 Type indent
 Press ENTER
 Type t(ype)
 (The options will show at the bottom of your screen.)
 Type p(aired)
 Type d(ocument)
 Type indent quotes w/ single space
 Press ENTER
 With a paired style, you have an extra option:

in front of the [comment] (in the bottom screen).
 Cursor one column to the right. Your cursor will now be below the comment line (in the top screen) and be to the right of the [comment] (in the bottom screen). Whatever we place here will happen when the style is turned off (or when you press ENTER in your document).
 Press ENTER twice
 (This will give you one blank line after the indented quote.)
 Press Shift F8, l(ine), s(line spacing)
 Type 2
 Press ENTER
 Press Exit (F7) three times
 You are now back to the style menu. It should look like Figure 2:
 You want to add the second style to your style document (master.sty).
 Press s(ave)
 Type master.sty
 When asked to replace, press y
 Press F7
DO NOT CLEAR SCREEN!
To Test the Indent Style:
 Make sure there are no codes in the edit screen.
 Reveal Codes (ALT F3 OR F11).
 Double Spacing (Shift F8), l(ine), s (line spacing)
 Type 2
 ENTER, Exit (F7)
 Type two sentences
 At the end of the second sentence, turn your indent style On (Alt F8, o). Type two sentences for your indented paragraph and press ENTER. Type two more sentences. You can save your document or just clear

|Place style on codes above, and style off codes below|

```
{ . . . . . }  
[comment]
```

Press EXIT when done

Doc 1 Pg 1 Ln 1 Pos 10

Figure 1

Styles

Name	Type	Description
indent	paired	indent quotes w/single space
letterhead	open	letterhead with automatic date

1 On; 2 Off; 3 Create; 4 Edit; 5 Delete; 6 Save; 7 Retrieve; 8 Update 1

Figure 2

the screen. Below is a sample:

This is a sample to show how an indented quotes would look in a document. I have a document that is double spaced, and I will put my indented quote right after this sentence.

A large rose-tree stood near the entrance of the garden: the roses growing on it were white, but there were three gardeners at it, busily painting them red.

Alice in Wonderland

Now I go back to double spacing and finish the rest of my document. You can see how and indented paragraph looks.

#3 Envelope (This will only work if you have already created an envelope form).

Go to style menu (Alt F8):

(If you cleared the screen, there will be nothing listed on your menu.

Press r(etrieve)

Type \wp50\styles\master.sty

Press ENTER

Press c(reate)

Type n(ame)

Type envelope

Press ENTER

Type t(ype)

(The options will show at the bottom of your screen.)

Type o(pen)

Type d(ocument)

Type envelope format

Press ENTER

Type c(ode)

Check the description above for envelope style.

Press shift F8, p(age), s(paper size), e(nvelope), e(nvelope)

Press m(argin)

Type 12

Press ENTER

Type 2

Press ENTER twice

Press 1(ine), m(argins)

Type 65

Press ENTER

Type 2

Press ENTER

Press F7 three times

To Add To a Style Document:

Type s(ave)

Type master.sty

When asked to replace, press y

Press F7

To Test Your Envelope Style:

Press Alt F8, o(n)

The cursor is on Column 65.

Type in a name and address and print.

If the placement on the envelope is not to your liking, change the:

T/B margin and/or L/R margins.

Clear the screen (F7, n, n)

To create #4, Columns, go to style menu (Alt F8)

Retrieve master.sty

Type c(reate)

Type n(ame)

Type columns

Press ENTER

Type t(ype)

(The options will show at the bottom of your screen.)

Type o(pen)

Type d(ocument)

Type 2 parallel columns with default margins

Press ENTER

To Create Columns:

Press Alt F7, d (WP5.1 - Alt F7, c, d)

Now you have the screen in Figure 3.

Press t(ype)

The choices will appear at the bottom of your screen.

Press p(arallel)

You don't have to adjust the margins, because you are using the defaults.

Press F7

Press c(olumn on/off) (This turns column on.) (In WP5.1, o(n))

Press F7 twice

You are now back to the style menu. It should look like Figure 4.

Add the fourth style to your style document (master.sty).

Press s(ave)

Type master.sty

When asked to replace, press y

Press F7

To test the column style, go to Alt F8, Press o(n).

Check the status line. Notice it now says:

Col 1 Doc 1 Pg 1 Ln 1 Pos 10

Type a sentence.

Press Ctrl ENTER (Hard page break creates your next column.)

Check the status line. Notice it now says:

Col 2 Doc 1 Pg 1 Ln 1 Pos 10

Type a sentence.

To go between the columns, press the Goto key (Ctrl Home) and the arrow keys.

The column mode will stay on until you turn it off. Turn off the columns at the end of the last column. (Alt F7, c) (WP5.1 - Alt

F7, c, f)

Now you have a style document (master.sty) that has four styles in it. You can create more styles to fit your needs and add them to master.sty or you can have several style documents. For example, since I do a great deal of writing, I have one style document, called book.sty.

(The styles in it are: chapter, title page, table of contents, index). I have another style document called master.sty. (The styles in it are: doc, envelope, letterhead, indent).

When you leave WordPerfect, the style menu will be empty again. Or if you clear your screen (F7, n, n), the style menu will be empty. There are three ways to retrieve a

Text Column Definition

1 - Type	Newspaper					
2 - Number of Columns	2					
3 - Distance Between Columns						
4 - Margins						
Column	Left	Right	Column	Left	Right	
1.	10	40	13.			
2.	45	75	14.			
3.			15.			
4.			16.			
5.			17.			
6.			18.			
7.			19.			
8.			20.			
9.			21.			
10.			22.			
11.			23.			
12.			24.			

Selection: 0

Figure 3

Styles

Name	Type	Description
columns	open	2 parallel columns
indent	paired	indent quotes w/single space
letterhead	open	letterhead with automatic date
envelope	open	envelope form

1 On; 2 Off; 3 Create; 4 Edit; 5 Delete; 6 Save; 7 Retrieve; 8 Update 1

Figure 4

style document you have created.

- If you have used a style in a document, when you save that document, the total style document is saved with it. Therefore, by retrieving that document, the style menu will list all the styles in that style document.
- You can retrieve a style document in the style menu.
Press ALT F8
Type r
Type <name of style document, including directory, i.e., \wp50\styles\

master.sty>

- You can have the style document come up in the style menu automatically by placing it in the LOCATION FILES in the SETUP MENU.
Press Shift F1
Press L (location of files)
Press L (Style Library filename)
Type <name of directory and style document>, i.e., c:\wp50\styles\master.sty
Press ENTER
Press Exit (F7)

To Edit a Style:

Press ALT F8
Cursor to style or Press N, type <name of style>, press ENTER)
Press e(dit), c(ode)
Make changes. If you need to change margins, etc., remember to delete the old codes first.

Press F7 twice
Press s(ave), master.sty, ENTER
Press y to replace
Press F7

If you wish the change to be permanent, be sure and save the style document. If you only wish the change for the one file, don't save the style document. The change in the style will be saved with your document.

To Change Your System Date:

Go to DOS (CTRL F1).
Type date
Type <today's date> in the following format: mm/dd/yy, i.e., 03/12/90

Press Enter
This date will remain correct until you turn off the computer.

If your computer has a system clock, the date will stay even when the computer is turned off.

If you have a system that gives you the DOS prompt (C: or C>>) when you turn on your machine, but the date is wrong, type the date command before you go to WordPerfect.

Update

Suppose you make a change in a style, then you decide you want the original style. Press u(pdate). The list in the style menu will return to the original style document.

Some Ideas for Styles

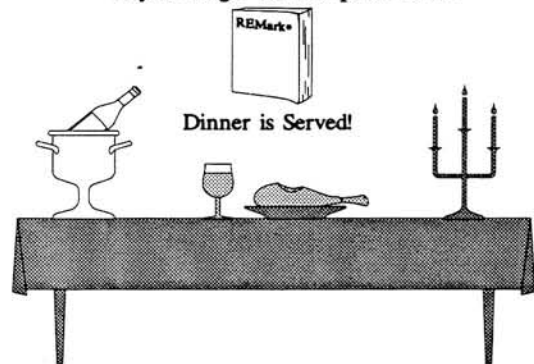
- Specialized signatures with job titles.
 - A special document format such as one with double spacing.
 - A format with numbered paragraphs.
 - A format for bibliographies
 - A format for an index page
- Happy styling. Our next trip will be lessons on creating and using columns. ✨

Classified Ads

HEATHKIT H14 PRINTER, \$59.00. Add 10% for UPS. Ronald P. Hoff, 722 Frederick Street, Cumberland, MD 21502-1736, (301) 724-1617.

GEMINI PC EMULATOR FOR Z-100, \$150.00. Konan hard disk controller for Z-100, \$100.00. (619) 375-5331 after 5 PM, PDT.

If you hunger for Computer news...



On the Leading Edge

William M. Adney

P.O. Box 531655

Grand Prairie, TX 75053-1655

Copyright (C) 1991 by William M. Adney. All rights reserved.

LaserJet III Printing with Windows 3.0, Imager, Backup Alternatives, Backup Strategies

In the last column, I mentioned that I had some problems printing from the Windows 3 Notepad on my LaserJet III printer. I also mentioned that I had called Hewlett Packard about a new driver for the printer, and here are the results.

Windows 3.0 and the LaserJet III

The LaserJet III driver I received from Hewlett Packard works fine, and it corrects the problem I mentioned last time. In case you missed my last article on the subject, I first noticed the printing problem when I attempted to print the Windows "readme" information (from the Notepad) on my LaserJet III. When I attempted to print the readme information, the top half of the last line on the first page was printed, and similar problems occurred on subsequent pages. Sometimes the bottom half of the characters in the first line on the page was printed. For those of you unfamiliar with the LaserJet, there is a "reserved" margin of about a half inch on all four edges of the paper that is "unprintable." It was quite obvious that the Windows 3 LaserJet III printer driver did not seem to know about the unprintable areas on the page, and it attempted to print there.

From what I've read, all versions of Windows have apparently had a similar problem printing on all models of LaserJet printers as I mentioned in my last article. For whatever reasons, Microsoft has either been unable or unwilling to correct the problem, which is kind of interesting considering how long the LaserJet has been around in one form or another. I mention this specifically because I have received a couple of letters from several members that pointed out the same problem. Although my experience was specifically with the LaserJet III, I understand that Hewlett Packard has drivers for other LaserJet

models too, and these new drivers correct the same kind of printing problem.

Installation of the driver is a snap, and there is a README file on disk that you need to review. First, you copy the driver and help file to the same directory where your existing LaserJet III driver is. Mine were in the \WINDOWS\SYSTEM directory. Then, start Windows, fire up the Control Panel, and make sure that the LaserJet III is the active printer. According to the README file on the disk, it is necessary to do this to make sure the WIN.INI file is updated with the correct information. That's all there is to the installation, assuming you have already installed the LaserJet III; otherwise, you will need to go through the usual printer installation process.

As I was doing the installation, I did notice some interesting technical details. The disk I received from Hewlett Packard was dated 9/5/90 and contained three files: README, HPPCL5A.HLP, and HPPCL5A.DRV. Those files had a size of 8,850 bytes (dated 8-10-90), 13,871 bytes (dated 8-3-90), and 416,115 bytes (dated 8-6-90) respectively. In contrast, the corresponding LaserJet III driver file included on the Windows 3 distribution disk was much smaller: it was only around 216,000 bytes. The most obvious point is that the driver from Hewlett Packard was nearly twice the size of the one from Microsoft. From that, I conclude that there must be quite a few features that Microsoft left out of their driver.

Regardless of what happened, you can obtain a LaserJet driver for Windows from Hewlett Packard by calling the distribution warehouse at (303) 353-7650. Be *SURE* that you know what LaserJet model you need the driver for, and be prepared to supply that model to the HP representative. Model identification on the LaserJets

is easy because it is identified on the top or front of the unit. For example, if your printer just is identified as *LaserJet*, you probably have the original unit. You might also have the LaserJet Plus, LaserJet II, LaserJet IIP or IID, LaserJet III or LaserJet IIID. Again, be sure you know what model you have so you can request the appropriate driver for it. Different models have different features.

The Imager

I have mentioned the Imager hardware and software a number of times in the last several years, the most recent of which was in the December 1990 column. Unfortunately, I found out that the company which made the Imager (AutoFax) has apparently gone out of business, although I had checked with them as I was writing the December article and there was no mention of any problem like that. I found out about that because I had tried to call AutoFax about a technical problem and heard a recording that the "number was disconnected and there was no new number". I called the Light Pen Company and learned that they have not been able to reach AutoFax either. One of the support representatives at Light Pen Company also told me that they only had Imagers set up with IRQ 2, so that is not particularly helpful if you were interested in a board with a different IRQ as suggested in my December article.

That is also not particularly helpful to those of us who already have and use an Imager because it is not likely that additional software enhancements will be available to add some features I had hoped for. And, as I learned the hard way, there may be some other problems that occur which require technical support, which is one of the reasons I tried to call AutoFax in

the first place. Although I have seen a number of strange problems in my years of fooling around with computers, this is one of the most elusive I have ever found because it involves a conflict between the Imager hardware/software and a video board.

Interrupt Overlap

The Z-449 video board, which is an autoswitch unit, came with my Z-386/16 and I have used it a lot. The Z-449 is a dual-port board which supports the "old" digital monitors (with a 9-pin connector) as well as analog monitors (with a 15-pin connector), such as the FTM monitor (ZCM-1490 series). The Z-449 provides video support for CGA through EGA resolution, and it has a "pseudo-VGA" mode as well. I also have a Heath HVB-550 video board that supports true VGA, and it is also a dual port board. Depending on what I am doing, I sometimes use the HVB-550 video board, and that's how I found this problem.

I was attempting to restore a subdirectory with the Imager (and its software) when the software displayed an *Interrupt [sic] overlap* error message and terminated the program without restoring the files. Fortunately, I remembered that I had changed the video card since I had last used the Imager, so I changed it back, and the restore worked perfectly. Since that error message is not documented in the Imager manual, I attempted to call AutoFax and received the recording about the disconnected number. I still do not know exactly what an "interrupt overlap" is, but I do know it is quite unlikely that I will find a technical trick that will allow me to use the Imager with the HVB-550. For now, I have found a way to work around the problem by simply using a different video board with the Imager. But I think this situation illustrates a couple of very important points and provides the opportunity to smash a popular myth about computers.

The Myth About Compatibility

Many people seem to believe the myth that a "compatible" computer is always 100% compatible with everything, both hardware and software. That is simply not true and never has been. NEVER! It does not matter what brand of computer you are using (including IBM), although you will generally find that something which is compatible with a Zenith Data Systems computer is usually, but not always, compatible with other brands as well. And to carry that same idea one step further, I find that many people unreasonably expect that today's hardware and software will be compatible with that developed in the future. Not a chance . . . not even a little one! There are literally hundreds of examples of this kind of "incompatibility", and I'll mention just a few of them.

If you are using or considering using

Windows 3.0, for example, consider that you MUST buy new (or upgraded) software to take full advantage of Windows' new features in this version. Windows 3.0 also does not like the 84-key keyboard on Z-241 and Z-248 computers, and you may need to get a new ROM (at least version 2.1) and a 101-key keyboard for that to work.

For IBM computers, the PC/XT/AT hardware was not even remotely compatible with that used on the PC Jr., and some software would not work either because of the PC Jr.'s strange keyboard. You also cannot use an IBM AT to read a 3.5-inch floppy disk that was formatted with PC-DOS 4.0 on an IBM PS/2 computer unless you use the DASDDRVR.SYS driver that was included on a patch disk for that DOS version. At this point, I am planning some additional testing to determine exactly what the problem is, and I expect to include that in a future column. See my December 1990 column for more information on that problem.

And of course the Zenith Data System eaZy PC cannot be expanded using "standard" hardware, which makes it similar in many respects to the PC Jr. and some of the Tandy 1000 series models. Even though the eaZy PC is not quite the orphan that the PC Jr. is, the eaZy PC has still been a disappointment to a lot of owners. Like most of these less expensive systems, the eaZy PC was not designed to accommodate standard video, disk drive, and memory additions. All of these kinds of systems—the eaZy PC, the PC Jr., and many of the Tandy 1000 series—were designed as low-cost systems without the expansion features of the more expensive computers. In general, what you have is all you can have. With an eaZy PC for example, you cannot add a VGA monitor, a larger hard drive (or another one) or megabytes of expanded or extended memory.

For Tandy (Radio Shack) computers, not all IBM compatible software will run on some of the 1000 series computers because they have a unique keyboard. Many of the usual hardware upgrades for "compatible" computers will also not work on some of the 1000 series computers because of lack of space or standard "connectors".

That is just a part of the problem. As of right now, there are four hardware "standards" for a computer's bus configuration: ISA, "proprietary" ISA, EISA, and MCA. The bus used in the old IBM PC, XT, and AT is generally referred to as an Industry Standard Architecture, or ISA, bus. IBM introduced the Micro Channel Architecture (MCA) bus in the PS/2 series of computers, and the MCA bus is completely incompatible with the ISA bus. From a user perspective, that means you cannot use any of the "old" boards designed for the ISA bus in the MCA bus, and to add insult to injury,

the MCA boards cost more than the ISA boards.

Perhaps the most important reason for a new bus design was dictated by the fact that an 80386-based computer requires a 32-bit bus that was not supported by the original ISA (16-bit) bus. And many of the original 80386-based computers, such as the Z-386/16 and Compaq DeskPro (20 MHz), were designed before some of the details of the MCA bus were available. Because a 32-bit bus was required for these systems, manufacturers had to "enhance" the existing ISA bus to handle this more powerful CPU. For that reason, hardware support for 32-bit boards (such as memory) was generally limited because each manufacturer developed a unique (sometimes called proprietary) ISA bus that had been expanded to support the required 32-bit architecture. That's why 32-bit memory cards for computers like the Z-386/16 and DeskPro were only available from the manufacturer. And that caused an obvious "compatibility" problem. As a result, a group of manufacturers, including Zenith Data Systems and Compaq, worked together to develop a new bus standard which is now known as the Extended Industry Standard Architecture, or EISA, bus.

The whole point of this is that there is NO hardware compatibility between the EISA and the MCA bus. In fact, you cannot even insert an EISA board into an MCA bus because the connectors are different. And of course both bus configurations are not remotely compatible with the ISA bus, except that most EISA computers do provide at least one ISA slot for "older" boards.

Although there is generally some level of hardware compatibility among all of the so-called "compatibles", the bottom line is there is NO absolute guarantee of compatibility in all possible configurations, even within the same brand of computers. And when one considers the wide variety of hardware types and manufacturers, I am amazed that the general level of compatibility is as high as it is. But enough of the compatibility issue. Let's consider some ideas about backing up a hard drive.

Hard Drive Backups

Perhaps the first question is: Why should you back up a hard drive? I think that the most important reason is because it is an absolutely guaranteed that your hard drive will fail at some point. And it is possible to lose thousands of hours worth of work, especially if one considers the incredible capacity of today's hard drives. You may have to cope with restoring hundreds of megabytes of programs and data, and that will take more time than most users can afford. Even restoring programs from the distribution disks can be difficult and time consuming, particularly if you have developed a "custom" configuration that most

of today's software supports.

Although I have mentioned the general subject and importance of hard drive backups frequently in this column, there are really three important issues to consider. First, you must consider the question of WHEN to back up a hard drive. Second, you must consider what ALTERNATIVES (e.g., tape or floppy disks) are available to do the backup. And third, you need to determine a backup STRATEGY that works for you.

WHEN to Do a Backup

How often you need to do a backup frequently depends on the volatility of the data. That is, how often do you change the contents of existing files or create new ones? For example, one can make a good case for backing up at least some files every day if it is used for order entry in a business, especially if a data base is used. A business could lose considerable money if orders are "lost" because of a hard drive failure, not to mention problems with irate customers. But there are a number of other times that you should back up at least some of the files on hard drive, and I have listed the most common ones in Figure 1.

The list in Figure 1 is quite specific, but there are some good practices that should be followed to minimize data loss when a hard drive fails. For example, it is good practice to back up files when any software is changed or added to a hard drive. If you have used some of the installation programs, you will know that, in addition to the fact that they take a lot of time, you may have customized or added features that are a little different than the normal ones. Many people, including me, do that customization over a period of days or weeks, so you really don't want to lose all that work.

It is also good practice to back up files before you make any hardware changes, such as adding a new printer, more memory, or anything else. After you add or change hardware, you may also find that you need to make changes to your software also, which is included in the previous paragraph. If you add a new printer for example, you will frequently find that you must also install that new printer as described in the previous paragraph. And some software is kind of funny about printers. For example, if you install a new printer

in Word Perfect and make it your default printer, Word Perfect will automatically format any data file you edit for that newly installed printer. Moreover, it sometimes happens that a hardware change results in some kind of a problem that could cause a loss of data, such as accidentally moving a switch or bumping a connector so that it fails to make proper contact.

The last category of backups includes making special copies of data files that are the result of a lot of work and/or time. Complicated spreadsheets, data bases, and program files are particularly good examples of this concept.

WHY to Make a Backup

Perhaps the most important reason to make a backup is that a hard drive is guaranteed to fail because it is a mechanical unit. Electronics can fail too, but you probably know that electronic components generally fail less often than mechanical ones. The major question about hard drive failure is not *if* it will fail, but *when* it will fail.

More importantly, my experience is that a hard drive most often fails with no warning whatsoever. I've had a half dozen hard drive failures in nearly nine years of working with microcomputers, and five of them failed when I powered up the system first thing in the morning. For the other one, I noticed that the hard drive had an increasing number of bad sectors identified by the old DETECT command, so I backed it up and got rid of it before it finally failed for good.

Even if you only use floppy disks, it's good practice to make periodic backups of your critical data files, especially those in which you have invested a lot of time in creating. Again, spreadsheet files are a particularly good example of relatively small files that may have a considerable investment of time in their creation. The DISKCOPY is a very easy way to make relatively quick and painless backups of floppy disks.

Backup Alternatives

At the present time, there are two viable hard drive backup alternatives. The first is to use a software program to back up files to a floppy disk set. The second alternative is to use a tape backup system. There is also the possibility of using a CD-ROM (Compact Disk-Read Only Memory) for backup,

but the prices for those units are generally in the \$1,000 range, which makes them out of reach for many users. These units are sometimes called WORM drives, and the acronym means Write Once, Read Mostly (or Many). While the CD-ROM drives have a number of advantages, including speed, I will probably not think much about getting one until the "erasable" units become generally available at a reasonable price.

There are many number of software programs that one can use to back up and restore files, and I have tried many of them. Of course MS-DOS includes the BACKUP and RESTORE programs, but I have found them much slower and more difficult to use than other software.

FastBack still seems to be very popular, but I have found it to be more expensive than other utilities, and I have never used it on my own systems because of its cost. I have tried the backup and restore programs included in the Mace Utilities, but I had problems with their displays on my laptop. During a restore the program would occasionally generate garbage on my SupersPort 386SX laptop (and the SupersPort 286 as well), but there did not seem to be any problem with file integrity.

I have also tried the Norton backup/restore programs, but they are much slower than the other programs I have tried on my systems. All in all, I prefer and use the backup and restore programs that are included in PC Tools. These programs are fast and easy to use. The programs also sense when a disk is changed, so you do not have to "press any key" to continue with a backup or restore (Mace also has that feature). The PC Tools backup/restore programs also seem to be the most flexible for my needs because they have implemented a number of options that provide an extremely flexible use of backup strategies. The PC Tools backup/restore programs are apparently so popular that they are available as a standalone package, but I recommend spending the extra few dollars for the complete PC Tools package. I have found it highly useful in a variety of different applications.

Types of Backups

My observations indicate that most users' backup strategy is not to back up a hard drive. That's too bad because I get occasional letters about what to do when a hard drive fails and there is no backup. Although it is somewhat understandable why an individual user does not make a periodic backup (because of time), I have always thought it was pretty stupid for a company to play "You Bet Your Business" by not making a periodic backup of critical information. Unfortunately, many people do not recognize that there are many ways to make backups, and you can develop a backup strategy that is relatively painless

1. Change or upgrade to new DOS version
2. Change or upgrade to an application (e.g., Word Perfect, Quattro, etc.)
3. Addition of a new application
4. Any change to the memory configuration, either boards or SIMMs
5. Change or upgrade to the video card/monitor
6. Change or upgrade to a peripheral (e.g., printer, modem, etc.)
7. Any major change to existing files, especially large files
8. Any new files that require lots of time/work to create.

Figure 1. When to Back Up Files on a Hard Drive.

and will protect your time and financial investment. To understand how to develop an effective backup strategy, it is necessary to review the four kinds of backups that are commonly available in backup software.

The full backup. The full backup includes making a copy of all files in each hard drive partition. Some kind of tape backup system is particularly good for this kind of backup because one can begin the process and simply wait for all files to be copied to a tape. Although that may take some time depending on what tape backup system you use, it is easy because one does not have to sit at the system and play musical floppies. In most cases, a full backup will reset the DOS Archive Bit to OFF (indicating that a file has been backed up). But it is very important to have at least one full backup for each partition to implement any effective backup strategy as you will see.

The incremental backup. An incremental backup is used to back up only those files that have been created or changed since the last full backup. What happens is that the backup software checks the DOS Archive Bit, and if that is set to ON, then the file is backed up and the Archive Bit is reset to OFF. The Archive Bit is reset so that the software can tell if a given file has been created or changed since the last backup.

The differential backup. Like the incremental backup, the differential backup also checks the DOS Archive Bit, and if it is set to ON, the file is backed up. Unlike the incremental backup, the differential backup does NOT reset the archive bit to OFF.

The specific subdirectory/file backup. This kind of backup is particularly useful during hard disk maintenance when you need to have more space on a hard drive or you are upgrading software. For example, if you are upgrading to a new version of Word Perfect, you may want to backup the subdirectory containing the old version to a set of disks just in case you need that old version again. Or you may find a group of files that you don't need very often and you can back them up in order to have more free space on a hard drive. Regardless of the reason, this is an effective way to reduce your hard drive requirements so that you do not need to get a larger hard drive every few years. I suggest doing this, particularly with older software versions, because you don't have to worry about the difficulty of reinstalling the software. My experience is that it is much easier to restore a program than it is to reinstall it. And of course you do not really need a special backup program for this because you can always use COPY or XCOPY.

With those kinds of backups in mind, let's take a look at several strategies that may help improve your recovery capability.

Defining Partitions

Although I have mentioned this before, it is important to consider how your partitions are defined in order to implement an effective backup strategy. You will find that it is much easier and faster to always have at least two partitions (or single partitions on two hard drives). Here's how I have set it up on my system.

Drive C is always the "system drive." That is, it contains only software (not data), including DOS and all my application programs for word processing, spreadsheets, graphics, and utilities. The important point about this is that the files on this drive do not change very often, which also means that the partition does not have to be backed up very often. As you will see, this kind of configuration makes it simple to take a full backup once every six months or so (depending on how much change or additional software is added). In the interim, it is also quite easy and fast to take a periodic differential backup every month or when there is a major change to software. A periodic backup is important because a number of small files, such as configuration files, may change during that period. Unless I add a large program, a periodic differential backup (e.g., monthly) rarely exceeds the space required on a single 3.5-inch high density floppy disk, even over a six-month timeframe.

Drive D is always the "data drive." It contains only data that is specific to my current needs, and the files on this drive may be created and/or changed on a daily basis. Because the files on this drive change frequently, I take an incremental backup frequently, usually at least once a week. And I keep at least three sets of incremental backups so that I always can find a previous version of a file. If I'm working on a major project that results in a major change to one or more files on a daily basis, I usually copy those files to a floppy disk at the end of the day, just in case. Because I use my computer for nearly all of my business, I do not have the time to recreate work that has already been completed. Not to mention the fact that I have a lot of deadlines that don't usually have much slack in them, so time is also an important factor.

The whole point is that you can make the backup process relatively painless if you define two hard drive partitions like this. Files on the system drive are usually not too volatile, meaning that they are usually not created or changed all that often, at least not as frequently as data files. That means you do not have to back up the system partition nearly as frequently as the data partition. On the other hand, data files tend to be VERY volatile, and they ARE frequently created or changed. That partition must be backed up on a frequent and regular basis. How often that partition needs to be backed up depends on how fre-

quently you create or add data files. In order to understand how to decide what kind of backup strategy to use on your system, let's take a quick look at what the Archive Bit is.

The Archive Bit

If you take a careful look at how a directory (or subdirectory) is constructed, you will find that the first 11 characters are reserved for the file name and file type (sometimes called the extension). The next character in a directory is called the Attribute Byte, which consists of eight bits. An attribute may be turned OFF, in which case the appropriate bit is zero (0). Or an attribute may be turned ON, in which case the appropriate bit is set to one (1).

Because each and every file has a directory (or subdirectory entry), each and every file also has an Attribute Byte. In the case of most data files, most of the Attribute Byte consists of zeros, except the Archive Bit which is set to one when a file is created or changed. Other attributes include the Directory Bit (which is used to indicate a subdirectory "file"), the Volume Label Bit, the System File Bit, the Hidden File Bit, and the Read Attribute Bit. A file may have more than one bit set to ON, depending on its use. For example, the BIOS and the DOS Kernel files usually are created with the System, Hidden, and Read-Only bits turned ON.

The Archive Bit is "maintained" by DOS, and it is turned on whenever a file is created or changed. Because the Archive Bit is set to 1 by DOS whenever a file is created or changed by any software, such as a word processor or spreadsheet, it is an easy way to identify files that have changed. Good backup software checks the Archive Bit and may reset it depending on the type of backup. You can also use the ATTRIB command in current DOS versions to change either the Archive Bit or the Read Attribute Bit. With that quick introduction to the Archive Bit, let's take a look at how you can use this idea to implement an effective backup strategy.

The Full Backup Strategy

All effective backup strategies begin with a full backup of each partition on the hard drive, regardless of what it contains. Most commercial backup software, including all of the products I mentioned earlier, turns the Archive Bit OFF to indicate that a file has been backed up. This establishes a *BASELINE* so that future backups can be defined to back up only those files that have been created or changed since the last full backup.

There seems to be an assumption that a full backup is always required, and because it takes a lot of time and effort, few users do it. Actually, you may find that you only need to take a full backup as little as once a year if you set up your system to

make it easy to recover any file. The whole idea is that taking a full backup every day, week or month is really not worth the time and effort unless you have nothing better to do. And most of us can find more entertaining things to do rather than swapping floppy disks or baby sitting a tape backup.

The Full/Differential Backup Strategy

After you have taken a full backup, a differential backup can be taken periodically to ensure that all files on that partition can be recovered quickly. Remember that a differential backup will copy all files that have the Archive Bit turned ON, but the process DOES NOT reset the Archive Bit to OFF. As I said before, this approach is especially good for non-volatile information, such as application programs on the system drive (drive C). My practice is to keep two "sets" of a differential backup, just in case one is not readable when I really need it. And because both sets only consist of a maximum of three disks each, it is quick and easy to make those backups on a regular basis. For my system drive, I make a differential backup at least once a month or whenever I add new software to my system drive. But the beauty of using the differential backup procedure is not really obvious until one considers how it is used to restore files.

Consider the extreme case where you lose all files in a partition. First, you must restore files from the last FULL backup. And second, you take the *most recent* differential backup and restore those files. That's all there is to it. And if a critical file that is included in the differential backup gets wiped out or corrupted on the hard drive, you only have to restore that file from that latest backup. If the file is not on the latest differential backup, then it is obvious that the most current version of the file is part of the full backup. As you can see, the differential backup makes it easy to locate the most current version of any file. Since that looks easy, let's take a look at how one can use the incremental backup strategy.

Full/Incremental Backup Strategy

This strategy also requires that you make a full backup first. An incremental backup will copy all files that have the Archive Bit turned ON, and the Archive Bit is reset to OFF as part of the process. I have found this kind of backup most useful for data files because I may want to recover difference "versions" of a file, which makes it especially good for data files (i.e., volatile files) on my system. My practice is to keep at least four "sets" of an incremental backup so that different versions of an active file will most likely be available. If I am working on a really time-critical project, I may simply copy a critical file to a floppy disk at the end of a day so that I always have the latest

version. For my data drive (drive D), I make an incremental backup at least once a week. By the way, I also make it a practice to make a backup of my data drive once a month so I do not need to have more than four sets of incremental backup disks. Now let's see how the incremental backup strategy is used in the extreme case of a total partition (or drive) failure at the end of a month (just before the next full backup).

The first step is to restore all files from the latest full backup. Then, files are restored from the first backup set. Files from the second backup set are restored next. Then, files are restored from the third backup set. And finally, the files are restored from the fourth and last backup set. If you have been keeping track of things, you will see that each incremental backup set may or may not contain files from the previous backup. More importantly, each incremental backup set must be restored in its original order to ensure that the latest version of a file is finally restored. For example, consider a file that was changed just before the first and fourth backup. When the first backup set was restored, that file was current as of the date of that backup. But it was not current as of when the fourth backup was taken, so it must be overwritten with the most current version.

Backup Strategies Compared

Although a full backup must always be available, it takes too much time and effort to frequently perform a full backup on all hard drive partitions. The only advantage of taking a full backup is that it contains all files as of the date of the backup, but you must have a supplemental strategy to back up files that are added or changed between the full backups. One way around this is to back up specific subdirectories/files to floppies on an as-needed basis.

The differential backup is particularly effective when the contents of a partition do not change frequently or when you want to quickly restore files that have been created or changed. The only disadvantage of using a differential backup on partitions that contain frequently created or changed files is that you will have an increasing number of floppy disks as you work with more files. That also means that it will take more and more time to perform a differential backup, which is the main reason I prefer an incremental backup for partitions that contain volatile data.

The incremental backup can help minimize the time required to perform a periodic backup because only files with the Archive Bit ON are backed up. The biggest advantage of using the incremental backup is that it requires the least amount of time (and fewer floppies), but it is more difficult to work with if you have to restore all files because of a hard drive failure. If you consider how often a hard drive failure is likely to occur however, the incremental

backup provides the necessary recovery capability and still minimizes the time required to make periodic backups. The only time an incremental backup really becomes painful is when a complete partition must be restored, but that is rare enough that the time savings on an ongoing basis are worth the risk.

The incremental backup also has the advantage that, by its nature, it will back up various "versions" of a file, which makes it especially useful for finding various spreadsheets, programs, and other items you might change during a development process.

And if you need to really be sure that you don't lose critical files, don't forget about making specific backups using the COPY or XCOPY command. That is one of my most frequently backup strategies, and it works. Despite the number of hard drive failures I've had, I have only lost a total of about two hours worth of work.

Powering Down

Regardless of what kind of backup strategy you use, remember to keep it simple and straightforward so that it is useful. No backup strategy is worthwhile if it is not performed on a regular basis. And one other point is worth mentioning.

It is good practice to have a "off-site" backup off all partitions. Don't try to get fancy with this idea because it is not necessary. If you have a full backup for the hard drive on your home computer, take a copy of it to the office and keep it in a desk drawer. Some people use a bank's safe deposit box as off-site storage. Off-site storage will protect you and your data in the extreme case of a building fire which destroys your computer.

For help in solving specific computer problems, be sure to include the exact model number of your system (from the back of the unit or series from the Owner's Manual), the ROM version you are using (use CTRL-ALT-INS to find it, except for the eaZy PC), 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., modem, bus mouse or video card), 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 parallel) 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 ZDS or Heath 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 Discussed

Software

PC-Tools \$149.00
PC Backup 99.00

Central Point Software
15220 NW Greenbriar Parkway, #200
Beaverton, OR 97006
(800) 888-8199 (Automated order line)

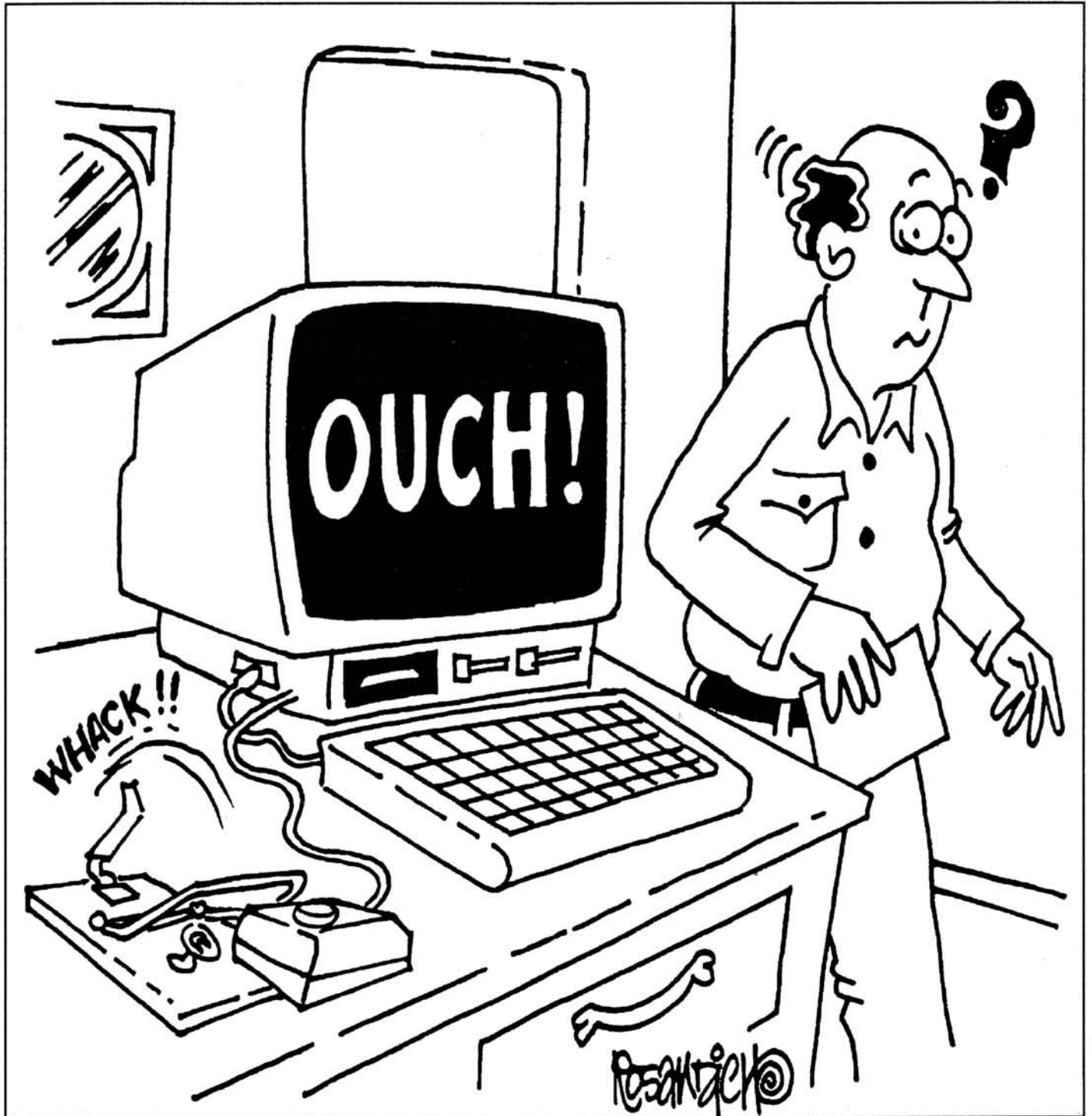
Mace Utilities \$149.00
FastBack 189.00
Fifth Generation Systems, Inc.
11200 Industriplex Blvd.
Baton Rouge, LA 70809
(800) 873-4384 (Orders only)

Norton Backup \$149.00
Symantec Corporation

10201 Torre Avenue
Cupertino, CA 95014-2132
(800) 343-4714

Hardware

HP LaserJet III
Payload Computers
15718 Sylvan Lake
Houston, TX 77062
(713) 486-0687 (Call for current price)*



Database Publishing With dBASE IV and A LaserJet Printer

Alan R. Neibauer
11138 Hendrix Street
Philadelphia, PA 19116

Database publishing does for database information what desktop publishing does for text. Its objective is to produce professional looking reports from the data in your database. Yet traditionally, database programs such as dBASE, deal with printers generically, transmitting only ASCII characters without formatting. Without individual printer drivers for specific printers, these programs have no built-in way of selecting fonts, printing graphics, or changing orientation.

This results in rather plain looking reports in portrait orientation — serviceable and informative but not very effective. With LaserJet and compatible laser printers, reports print in the internal Courier font. There are no graphics, and only one font and point size.

Of course, you can purchase add-ons that give plain vanilla database programs the ability to create attractive and imaginative reports. You can also save your data as an ASCII file, load it into a word processor, and format it using that program's capabilities.

With most programs, you could only take advantage of your printer's special capabilities by writing programs. Using control codes in program lines, you could print in landscape, select fonts and paper sizes, even print envelopes and labels. But you'd have to be an experienced programmer to create more complex command files that produce very professional looking reports.

Fortunately, dBASE IV comes with installable printer drivers that can print in portrait or landscape, as well as boldface, underline, subscript, superscript, and italics. You can even build in five fonts using the dBASE configuration file.

In this article, we'll focus on using dBASE IV's built in printer control powers.

Printer Drivers

dBASE IV comes with four LaserJet printer

drivers: HPLAS100.PR2 for portrait orientation and 100 dpi graphics; HPLASL.PR2 for landscape orientation and no graphics; HPLAS2I.PR2 for portrait orientation with 300 dpi graphics; and HPLAS2ID.PR2 for portrait orientation, 300 dpi graphics, and duplex mode printing on the LaserJet IID.

The major differences between the drivers are orientation, the character formats available, and the default setup commands transmitted to the printer. However, all of the drivers can be used with any model LaserJet, although you won't be able to access all features. For example, you could use the HPLAS2I and HPLAS2ID drivers to print italics on a LaserJet Plus with the A cartridge installed. But with the HPLAS100 driver, characters formatted as italic print underlined instead. So the main purpose of selecting the correct printer driver is to use your printers special features — not for

printing basic ASCII-text reports.

Installing dBASE IV

You add support for your printer when you install dBASE on your system. At one point in the installation process you'll see a printer selection screen (Figure 1).

Press Shift-F1 to display a list of available printers, scroll the list until Hewlett-Packard is highlighted, then press Enter to show alternative models and drivers (Figure 2).

You'll notice that the model names are not very distinctive, two just say LaserJet. But when you highlight the model name, the corresponding driver is shown in the right-hand list box. So highlight one of the models then press Enter. Type the port (such as LPT1 or COM1) then press Shift-F1 to select another driver. Repeat the process until all four LaserJet drivers are included.

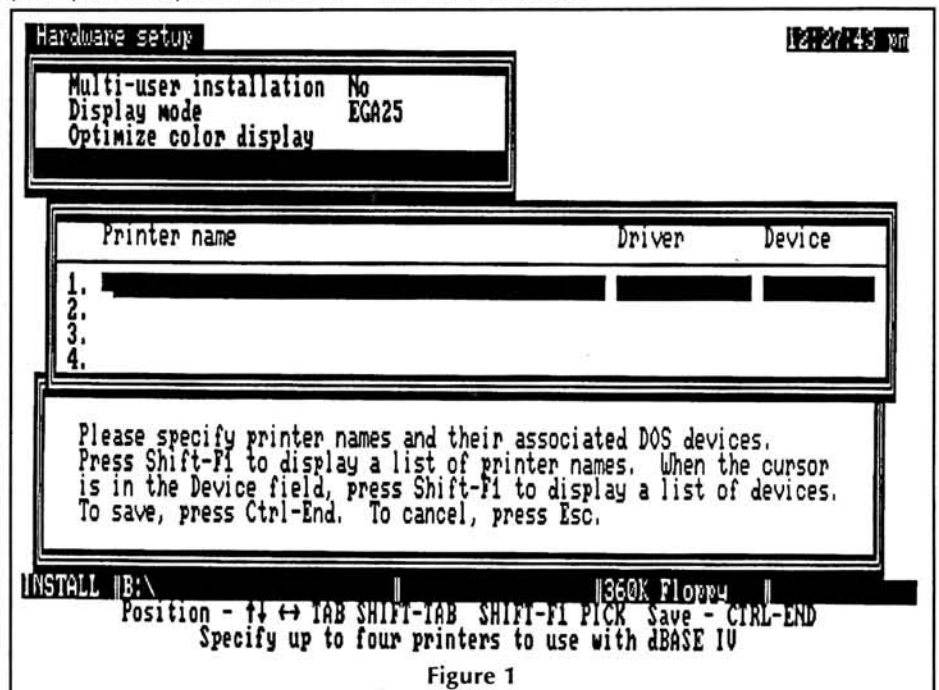


Figure 1

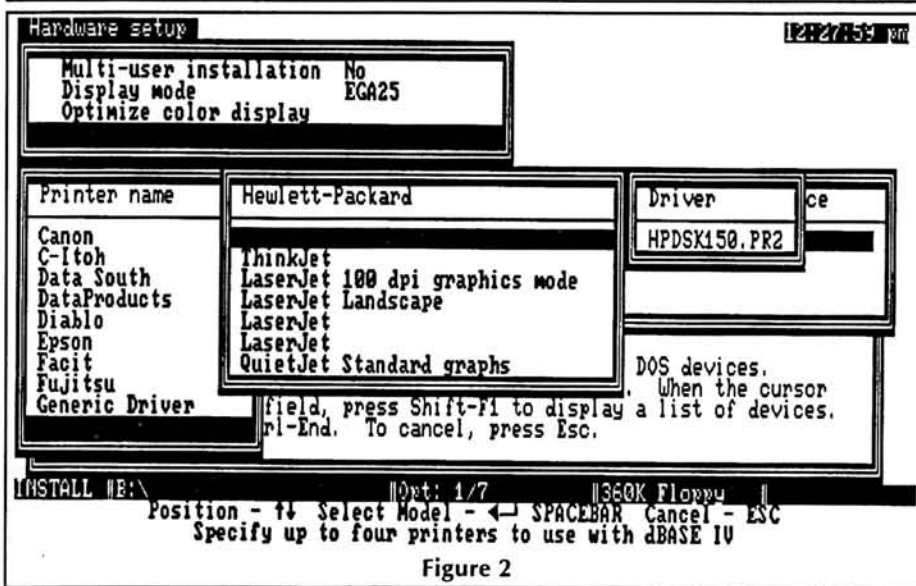


Figure 2

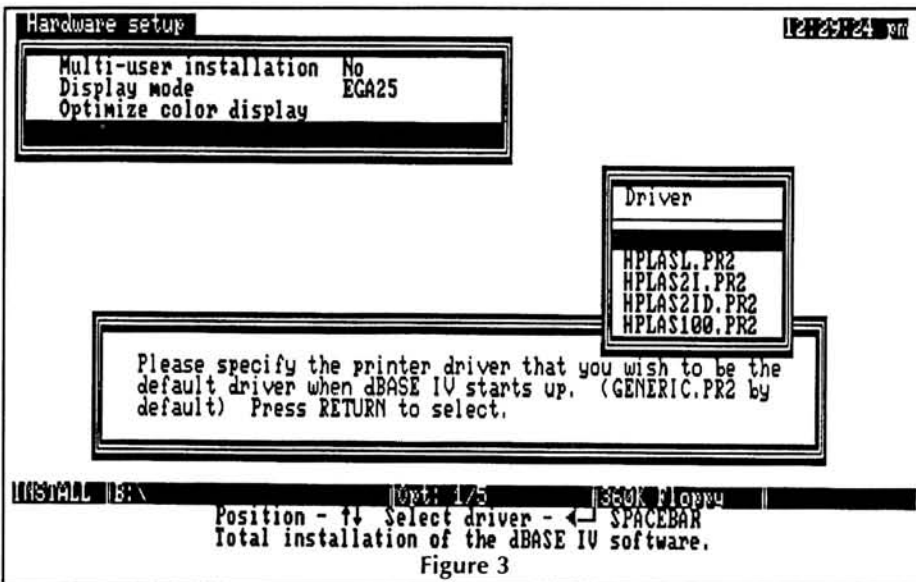


Figure 3

Press Ctrl-End to save the configuration and display the default printer screen (Figure 3). Select the driver you want to be the default, then press Enter. Follow the instruction on the screen to complete the installation process. When you're done, dBASE IV and the selected drivers will be on your hard disk. Although you selected one driver as the default to be used automatically by dBASE, you can manually select other drivers once in the program.

Also on your hard disk will be the file CONFIG.DB which contains other default settings and a list of your installed printers. You can modify this file to change drivers or add fonts by using the DBSETUP program in the Installation disk, or by editing the file with a word processor.

Interacting With dBASE

dBASE IV has an extensive menu system for creating and manipulating databases. For simplicity, however, we'll use instructions from the dot prompt. You're at this prompt when you see a single dot at the

left edge of the screen. To display the dot prompt from the menu system, press Esc then Y.

You can access the available pitch settings in a driver when you design report and label forms. The other features, such as boldface and underlining, must be included in a dBASE program, or a print command from the dot prompt. To use all of your printer's features you have to be quite familiar with most aspects of dBASE.

First, let's take a look at how to select a driver other than the default. Then we'll take a look at designing report forms.

Changing Printer Drivers

Any of the drivers you installed during the setup procedure can be selected from the dot prompt. dBASE provides a special variable, `_pdriver`, that contains the name of the printer driver. If you enter `?_pdriver` at the dot prompt, you'll see the name of the current driver.

To change drivers, just assign the variable a new driver name such as the command

```
_pdriver="HPLASL.PR2".
```

dBASE will respond with:

```
Printer driver installed  
HPLASL.PR2
```

In this case, all print jobs will now be produced in the landscape mode.

Creating Report Forms

One way to use the features made available in the printer driver is to create a report form. From the dot prompt, type something like create report items, or some other report name. The main report form screen will appear (Figure 4).

The easiest way to create a report is to press F10 to enter the command bar at the top of the screen, then select Quick Layout from the Layout pull-down menu. The field names will become column headings, and each field another column in the report. However, if you print a report in the portrait mode you'll get a plain looking printout. And if your records are over 80 characters long, some of the data won't appear at all.

If you have long records, you could use your printer's alternate pitches for either the entire report or individual bands, or sections. To print the entire report in the Line Printer font, for example, pull down the print menu, then select Control of printer to display the menu shown in Figure 5.

Toggle options by highlighting them with the arrow key then pressing Enter. Press Enter on the Text Pitch option, for example, to cycle between default, elite, pica, and condensed printing. Press ESC to leave the menu, then save the report using Exit.

If you want to change just one section of the report, such as the column headings, place the cursor on the section, press F10, then pull down the Bands menu. Set the text pitch or any other options for that particular band, then save and exit the report. Remember, report forms are printed from the dot prompt with the command report form items to print.

Formatting in dBASE IV Programs

Character styles such as boldface and italic must be activated in dBASE programs. By entering dBASE IV Style commands, or LaserJet PCL instructions, you can format individual characters, use downloaded fonts, or select any LaserJet feature.

In this article, we'll assume you know how to enter a dBASE program using the modify command program-name command. For instance, if you want to write a dBASE program to switch the LaserJet into landscape, you might enter this command at the dot prompt: modify command landscp. Write your program, then press Ctrl-W to save it and return to the dot prompt, or press Ctrl-Q to quit without saving the file.

dBASE provides a number of ways to

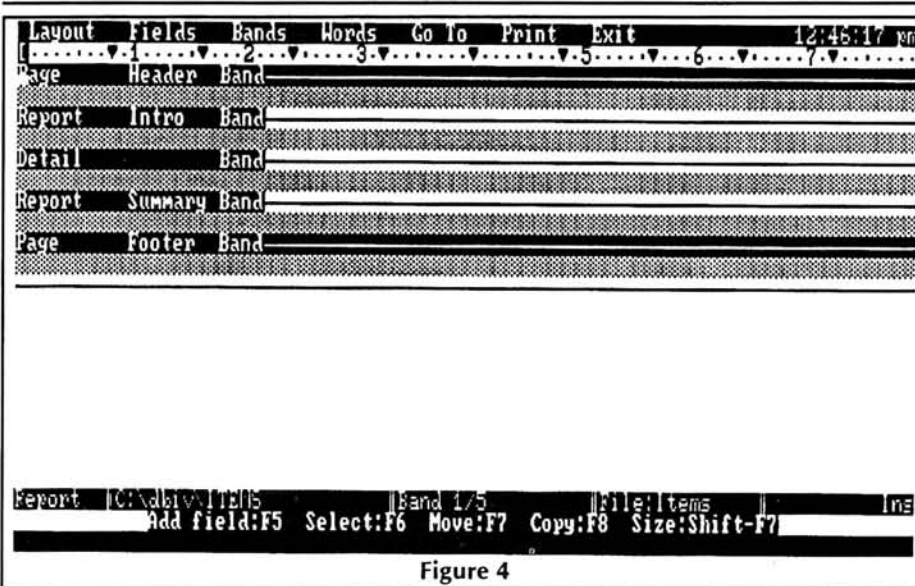


Figure 4

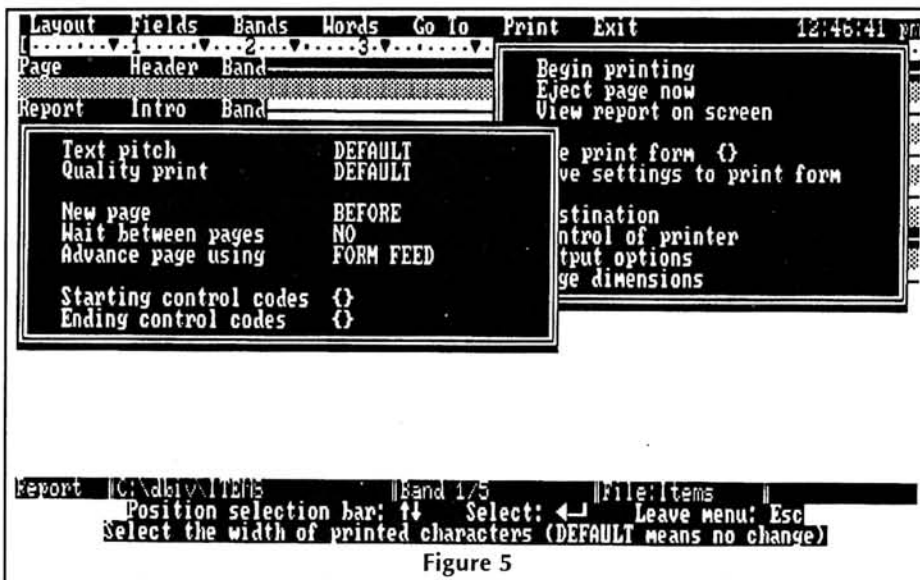


Figure 5

output data to the screen and printer, but the most fundamental are ? and ?? . This program, for example, prints one record from the database ITEMS:

```
use items
set print on
?
?"Stock Report"
?
?
?"Number of Order      ":"
?? ORDER_NUM
?"Part number          ":"
??PART_NUM
?"Item                 ":"
??PART_NAME
?"Unit cost            ":"
??COST
set print off
```

The use items command tells dBASE to get its information from the database called ITEMS.DBF, then set print on directs output to the printer as well as the screen. Each line starting with ? is an output line. They are either a blank line, print a string in quotation marks, or print the contents of a field. Finally, set print off stops further output from being printed.

The report generated by the program isn't very exciting. So let's see how we can use the Styles command in dBASE programs. dBASE has five character formats, each associated with a style code: B for bold, I for italic, U for underlining, R for superscripts (raised), and L for subscripts (lowered).

Using the codes, we can enhance our sample program this way:

```
use items
set print on
?
?"Stock Report" STYLE "I"
?
?
?"Number of Order      ":" STYLE "B"
?? ORDER_NUM
?"Part number          ":" STYLE "B"
??PART_NUM
?"Item                 ":" STYLE "B"
??PART_NAME
?"Unit cost            ":" STYLE "B"
??COST STYLE "U"
set print off
```

This program prints the heading in italic, the field names in bold, and the cost underlined. To print in italics, you must have a

cartridge or internal italic font. In addition, using the HPLAS100 and HPLASL drivers, text formatted as italic will be underlined instead.

If you want to footnote an entry or create formulas, use the R or L commands as in:
 ? "The sign for water is H"
 ?? "2" STYLE "L"
 ?? "O"

The ?? commands print on the current line without performing a line feed/carriage return. So in this case, the character 2 prints subscripted between the H and O.

Using Memory Variables

dBASE provides two special variables — _pscode and _pecode — that you can use to format your entire report rather than individual lines or characters.

Start by defining the codes you want each command to transmit. For example, the commands

```
_pscode="{27}{38}{108}{52}{72}"
_pecode="{27}{38}{108}{49}{72}"
```

are the decimal equivalents for the escape codes that activate the lower tray in a LaserJet IID, and that returns feeding to the upper tray. Use these memory variables to transmit any types of codes to the printer, even to select fonts or paper size.

Then add the line PRINTJOB at the start of your program and ENDPRINTJOB at the end, as in

```
PRINTJOB
.
.
rest of program
.
.
ENDPRINTJOB
```

The contents of _pscode will be sent to the printer when the job begins; _pecode when it is done. The codes will also be sent when a report or label form is printed.

Changing dBASE Configuration

Each time you start dBASE, the program checks the file CONFIG.DB. This file contains the names of your default and available printers, as well as other specifics about the dBASE environment.

For example, you'd find these lines in CONFIG.DB for a system with all HP drivers installed:

```
PDRIVER = HPLAS100.PR2
PRINTER 1 = HPLAS100.PR2 NAME "Hewlett-Packard LaserJet 100 dpi graphics mod" DEVICE LPT1
PRINTER 2 = HPLASL.PR2 NAME "Hewlett-Packard LaserJet Landscape" DEVICE LPT1
PRINTER 3 = HPLAS2I.PR2 NAME "Hewlett-Packard LaserJet" DEVICE LPT1
PRINTER 4 = HPLAS2ID.PR2 NAME "Hewlett-Packard LaserJet" DEVICE LPT1
```

The PDRIVER command shows the default driver to be used, while all of the selected drivers are listed in PRINTER commands. The commands include the driver filename, printer name, and port.

If you didn't install all of the drivers when you set up dBASE, you can either edit this file with a word processor, the dBASE

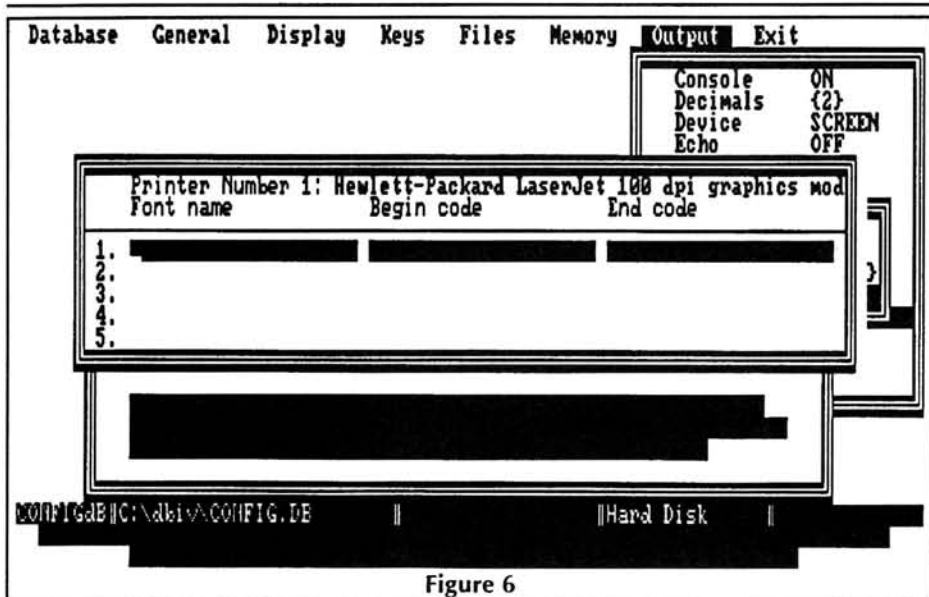


Figure 6

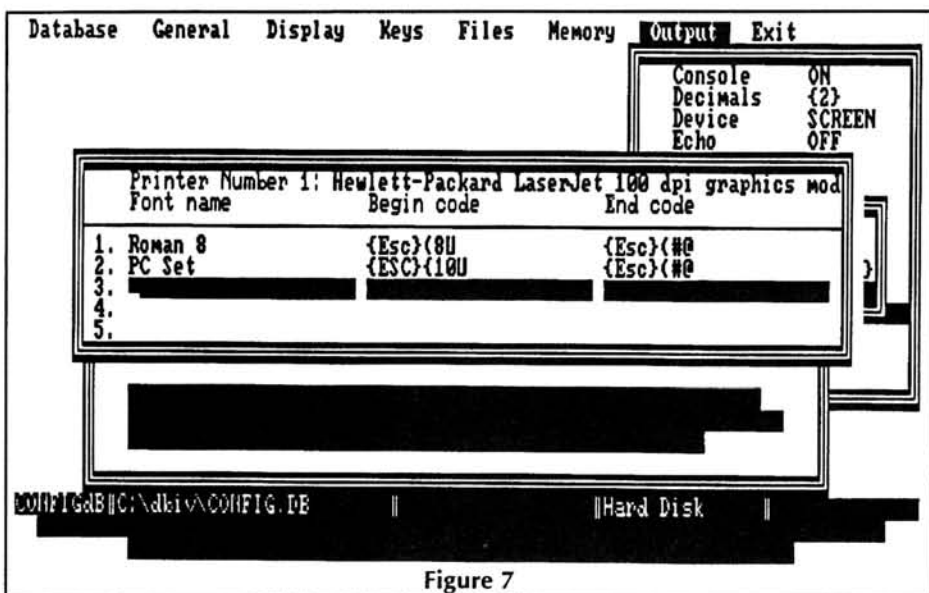


Figure 7

modify command function, or by using the DBSETUP program on the Installation disk.

LaserJet users should be most interested in the Printer Font command. Look at this section of another configuration file:

```
PRINTER 3 = HPLAS2I.PR2 NAME "Hewlett-Packard LaserJet" DEVICE LPT1
```

```
PRINTER 3 FONT 1 = {Esc}(8U, {Esc)(#@ NAME "Roman 8"
PRINTER 3 FONT 2 = {Esc}(10U, {Esc)(#@ NAME "PC Set"
```

The two Print Font commands include starting and ending codes to switch between the Roman 8 and PC symbol sets.

Use these codes with the style command, but include the font you want to change to, such as:

```
? "TEST" STYLE "1"
```

This command line selects the Roman 8 symbol set with the codes {Esc}(8U, then switches back to the original primary set at the end of the line — codes {Esc)(#@. Including Style "2" in the command would select the PC symbol set.

You can type these lines directly into the configuration file, or run DBSETUP. To use this program, insert the Installation disk in drive A, type DBSETUP, then press Enter. Select Config.db from the command line, then Modify existing CONFIG.DB from the pull down menu. Type the drive and path where you installed dBASE IV, then press Enter.

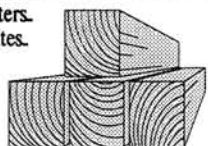
Pull down the Output menu then select Printer to display options. These let you set the default status of the printer, change or add drivers, set the default driver, and add fonts. Select fonts to display the menu shown in Figure 6.

Press PgDn or PgUp until the driver you want to add fonts to is shown at the top of the box. Then type the name for the font, and the beginning and ending codes in the appropriate columns, as in Figure 7. This shows the two symbol sets installed as Font 1 and Font 2. When you're done, press Ctrl-End to save the file, then Exit the program.

While dBASE doesn't provide a utility for modifying the driver files, you can change them using a sector editor such as PCTools. For instance, you could edit the driver to add italic support in the landscape mode, which is not available in the driver as it is supplied with dBASE. However, this is a complicated process that requires thorough knowledge of the PCL LaserJet language and your dBASE printer drivers.


But using the style command and selecting options from dBASE menus, you can produce interesting and professional-looking reports. If you have an earlier version of dBASE, you'll have to use LaserJet PCL commands directly in program lines. We'll look at these techniques in a future article.

Splinters.
termites.



With this board
all you have is
a stick in the mud!


Messages.
Orders.



With this board,
you're really talking!

Which is more useful?

Zenith Data Systems



Check out the
ZUG Software listings!

ENABLE Revisited

Advanced Graphics

Part 5

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

In the last article, I briefly covered the basic graph capability available with Enable. This function is available for both the spreadsheet and database. This article will cover the advanced graphics capability of Enable called Perspective.

Perspective was an add on software package that was available with Enable V2.0. I addressed this package briefly in my first set of articles. Enable has added Perspective II as an integral part of OA enhancements. This package was created by Three D Graphics and now supports color. The Three D package was also included with a spreadsheet package called Boeing Calc several years ago. This is the same package that is used to generate many of the graphic displays in pictures of computers in magazines. Look at the pictures of monitor in the current HeathKit catalog and you will see some of these Perspective graphics.

When you install Enable OA, one of the options is the installation of Perspective. This package does require an EGA card and monitor at a minimum. If your hardware supports this option and you have 2 Meg of Hard Disk space, you may install it.

Perspective is a very large and complex program. In Enable, it comes with its own manual. To give you an idea of size of the capabilities, the manual is as thick as the Enable spreadsheet and graphics manual. The package permits you to develop and use very nice looking graphs. In the earlier versions of Enable, you could generate the graphs and output them to selected devices, both printer and plotters. It was not possible to include the graphs in an Enable document. This problem has been fixed and the graphs can now be inserted in any word processor document version 3.0 and up.

To display a graph using Perspective, you must create it first in Enable. After

GRAPHIC CARD SELECTION MENU

- | | |
|------------------------------------|-------------|
| 1. Color Graphics Adapter (CGA) | — 640 x 200 |
| 2. Hercules Graphics Adapter (HGA) | — 720 x 348 |
| 3. Enhanced Graphics Adapter (EGA) | — 640 x 350 |
| 4. Video Graphics Array (VGA) | — 640 x 480 |
| 5. Extended EGA | — 640 x 480 |
| 6. Extended EGA | — 752 x 410 |
| 7. AT&T 6300 | — 640 x 400 |
| 8. Toshiba 3100 | — 640 x 400 |

Which Graphic Card is installed in your machine (default is #3)?

Figure 1. Configuration Menu One - Graphics Cards.

PRINTER SELECTION MENU

- 1. No Printer.
- | | | |
|----------------------------|-------------------------|----------------------|
| 2. AMT Printer | | |
| 3. Apple Laser Writer | | |
| 4. AST TurboLaser | | |
| 5. Canon LBP-8II Laser | | |
| 6. C.Itoh C-210XP/215XP | C.Itoh 310XP/315XP | C.Itoh 310CXP/315CXP |
| 7. C.Itoh C-715 | | |
| 8. C.Itoh C-815 | | |
| 9. C.Itoh Jet-Setter Laser | | |
| 10. Cordata LP300 Laser | | |
| 11. Cordata LP300X Laser | | |
| 12. DataProducts SPG 8050 | DataProducts SPG 8070 | |
| 13. Epson FX-80/100 | Epson FX-85/185/286 | Epson FX-80+/100+ |
| Epson LX-80/86/800 | Epson MX-80/100 | Epson RX-80/100 |
| 14. Epson EX-800/1000 | Epson JX-80 | |
| 15. Epson LQ-800/1000 | Epson LQ-1500 | Epson LQ-2500 |
| Epson SQ-2000 | Epson 24-Pin Compatible | |

Enter new device # (it's now #1). SPACE shows more. ESC keeps current #.

More (printer/plotter/film recorder) drivers may now be available. Call your dealer for more info.

Figure 2. Configuration Printer Screen One of Three.

PLOTTER SELECTION MENU

- 1. No Plotter.
2. Complot CPS-19 DM/PL language compatible
3. Epson Hplot HI-80
4. Houston Instruments 595
5. Houston Instruments 695
6. Houston Instruments DMP-29
7. Houston Instruments DMP-40
8. Houston Instruments DMP-41
9. Houston Instruments DMP-42
10. Houston Instruments DMP-51MP
11. Houston Instruments DMP-52 MP
12. Houston Instruments DMP-56A
13. Houston Instruments DMP-61
14. Houston Instruments DMP-62
15. HI-Plot 'PC Digital Plotter' Series
16. Hewlett Packard 7090A

Enter new device # (it's now #1). SPACE shows more. ESC keeps current #.

More (printer/plotter/film recorder) drivers may now be available. Call your dealer for more info.

Figure 3. Configuration Plotter Screen One of Three.

FUNCTION KEY SELECTION MENU

1. Vertical (most IBM and IBM compatible keyboards)
2. Horizontal (most other computers)

Are your function keys arranged vertically or horizontally?

Figure 4. Function Key Configuration Menu.

creating the graph, you will see the Perspective option on the graphics output menu. Selecting this option, Enable will load the Perspective package and load the data. The first time through, you will be prompted to configure the program for your computer. This configuration includes the function key layout. Perspective uses the function keys to move through the various options available. They are displayed on the screen with labels. Pressing the correct key normally will display another level of options.

Perspective also prompts you for the type of printer. Because of the capabilities of the program, the printer selection is different than for Enable. There are 34 printer drivers available with the program. Perspective also support different graphics cards and this must be selected. Actually, Enable treats Perspective as a completely different software application. When you select Perspective, you actually move into a shell outside basic Enable so you must select all of these options before you can proceed. You can select one of 38 plotters during this configuration routine.

After you have configured Perspective for your equipment, you will not have to do it again. If you change your hardware setup, you will have to remove the configuration file in the /3D/3DSYS subdirec-

tory. You can not change the setup from within Perspective.

After completing the configuration, Perspective will display the data from the Enable spreadsheet on the screen. It will use a basic 3D bar chart. By using the function keys, you can move between the various menus within the program. The selection of the function keys is carried through to the screen. The key and its function are displayed in the format selected. You select menu levels by pressing the correct function key. To remove the function key display, press the SPACE BAR. To return the function keys, press the SPACE BAR again. If you wish to move the location of the function key display, use the cursor

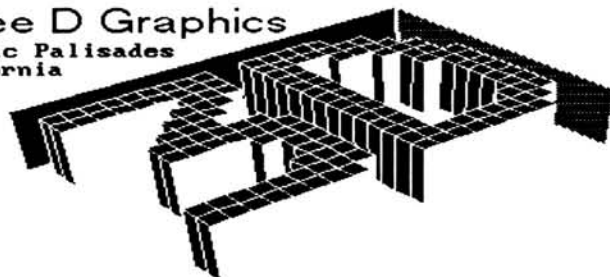
keys or the PgUp/PgDn. Using these combinations, you can move the display to any location on the screen that you want.

You may now modify the chart in many ways. You can select any of the 33 3D, 27 3D stacked, or 20 2D chart types. If you wish to work with 3D charts, chose 3D from the menu, **Preset Graphs** and then **Select 3D Types**. Perspective will now display all 33 types of charts you can use. Using the cursor keys, move the selector mark over the desired type and press F10. The graph data will now be redisplayed in the selected chart type. You can continue to change the chart type until you are happy with the look.

If the chart type you selected is pretty good, but could be better if it was rotated slightly up or down, Perspective can handle this for you. Press the function key that will chose **SelectViewing Angle**. Now displayed are 12 predefined viewing angles. Again, move the selection marker to the desired view and press **F10**. Perspective will now display the chart in the new angle. One advanced feature of Perspective permits you to change the rotation to whatever angle you need. This feature is really nice if you have a high speed CPU. If you select rapid movement, the graphic box outline will rotate in the selected angle at high speeds. This is about as close to real time motion as you get on a PC. To be effective, you will have to slow the rotation down. Again, pressing **F10** will select the current angle and display the chart.

While you would normally enter Perspective with data generated in Enable, you can add more data or edit data already loaded. Perspective even has a basic spreadsheet capability built-in. From the basic Perspective menu, select the **Data Manager** option. This will then display the data that you brought from Enable. You can move around the data screen and change numbers as required. To add you must use the function key options displayed at the top of the screen. After adding the new data, you must increase the range in order for the program to display the data. The program also permits some math functions. These are covered in the manual for the program. The math is not as easy as in the Enable spreadsheet as you must define

Three D Graphics
Pacific Palisades
California



Copyright (c) 1988

all rights reserved

Figure 5. Perspective Initial Screen.

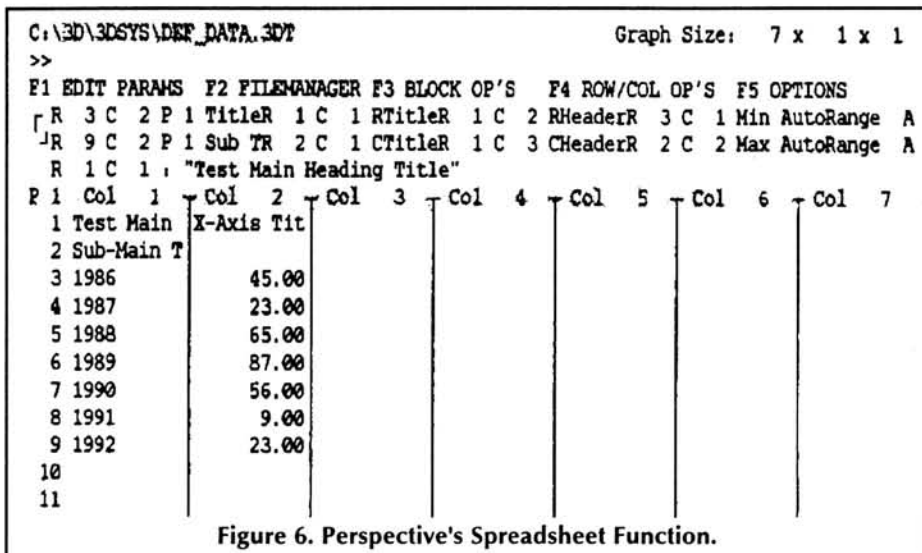


Figure 6. Perspective's Spreadsheet Function.

every row and column, including the block where you want the data to appear. The math functions are add, subtract, multiply, divide, and raise to power. The more advanced functions are sums, averages, count, minimum, maximum, standard deviation, exponential, and some trig functions. After completing the modifications, pressing F10 will display the new chart.

Working with the Label Manager, you can change the size and location of any label. The actual text must be changed in the Data Manager menu. The Label Manager permits you to position any label exactly where you want. It will also permit you to change the size, font, and add slant if desired. The program has nine fonts lo-

cated in two menu options. The basic font types are Roman, Swiss, Script, and Gothic. You can control the amount of boldness of each label. The amount of slant can be controlled from 45 degrees left to 45 degrees right and any angle between those two values.

Another function permitted by Perspective is displaying two charts on the screen at one time. To use this capability you must create the first chart and save it as a picture. Then you create the second chart. Note that the first chart is still displayed during this process.

If you are creating the charts for a visual presentation, you can further modify the chart by changing the colors of any part.

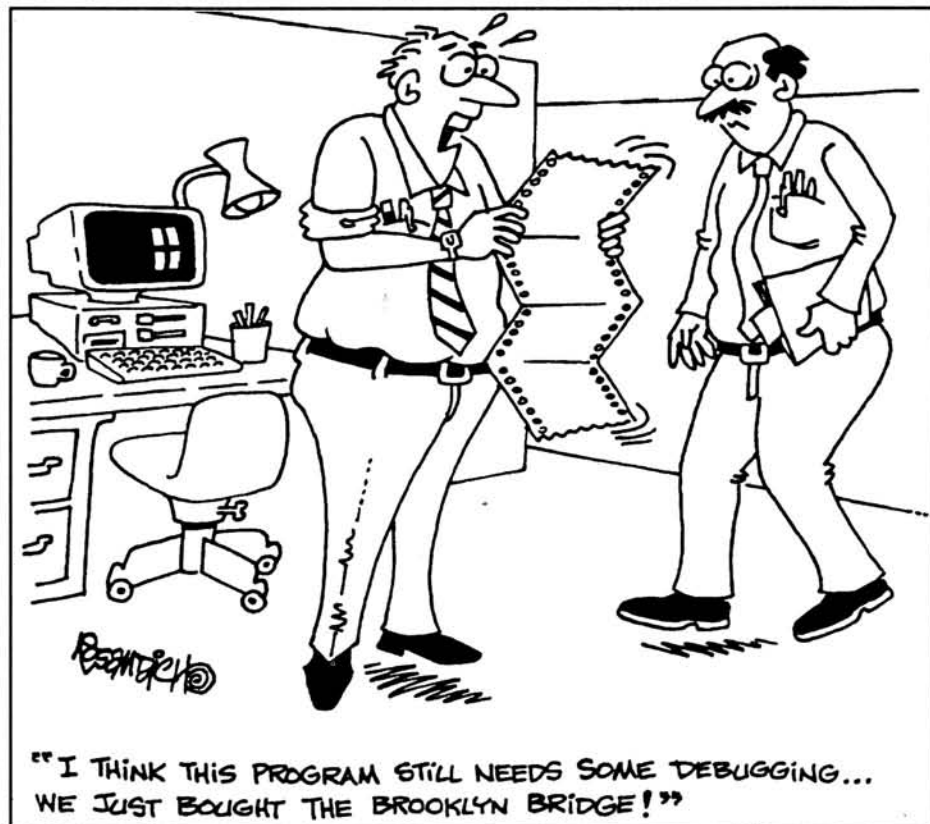
Perspective permits you to cycle through the color combinations by simply pressing a function key. You can also change the background pattern/color in the same manner. Again, these would be for charts that are to be presented visually.

Now that you have completed the chart, you need to make an output. Perspective permits you to make output to a printer. If your printer is black & white, you should change the chart to B&W and then print it, if it is acceptable. Color printers and plotters can be used to output color hard copies of your chart.

If you plan to use Perspective outputs in other programs, you should save the chart in a picture format. The program permits you save the files in .IMG for use in desk top publishing packages or .SCR for use in Paint Programs. If you want to import the picture into Enable's word processing, it must be saved in an IBM 6180 plotter format.

This has been a quick overview of the Perspective software package included with Enable OA. If you create a lot of charts, this package will give you the flexibility to make them look good. It has a lot of power and the output is limited only by your imagination.

In the next article I will provide an overview of the Enable data base module. This is one of the more powerful functions within Enable and has been improved in OA. I used this package to develop an application to run a small tool and die shop in about 300 hours. ✽



SEAGATE ST-251-1 42 MEG HARD DRIVES ON SALE

Seagate HARD DRIVES

MODEL	CAPACITY/FORMAT/SPEED/SIZE	DRIVE ONLY	XT KIT
* ST-125	21 MEG / MFM / 40 MS / 3.5"	\$229.00	\$277.00
* ST-138	32 MEG / MFM / 40 MS / 3.5"	\$277.00	\$325.00
* ST-138-1	32 MEG / MFM / 28 MS / 3.5"	\$307.00	\$355.00
* ST-151	42 MEG / MFM / 24 MS / 3.5"	\$353.00	\$401.00
* ST-157R	49 MEG / RLL / 40 MS / 3.5"	\$286.00	\$339.00
* ST-225	21 MEG / MFM / 65 MS / 5.25"	\$199.00	\$247.00
* ST-250R	42 MEG / RLL / 70 MS / 5.25"	\$248.00	\$288.00
* ST-251-1	42 MEG / MFM / 28 MS / 5.25"	\$289.00	\$337.00
* ST-4096	80 MEG / MFM / 28 MS / 5.25" FH	\$582.00	\$631.00
* ST-238R	32 MEG / RLL / 65 MS / 5.25"	\$218.00	\$271.00
* ST-277R-1	65 MEG / RLL / 28 MS / 5.25"	\$348.00	\$401.00
* ST4144R	122 MEG / RLL / 28 MS / 5.25" FH	\$623.00	\$671.00

* IDE, SCSI, ESDI AND OTHER SEAGATE MODELS AVAILABLE. PLEASE CALL

*** ZENITH PC COMPUTER UPGRADES ***

SmartWatch from FBE RESEARCH

⇒ Installs in ROM Socket on the CPU Board in Zenith computer series Z-100/138/148/150/160 and most all other XT computers. This clock/calendar contains a ten year battery and keeps your computer informed of both date and time at each boot-up. Instructions and software included. \$35.00

Z-150/160 MEMORY UPGRADE

⇒ This kit includes a replacement memory decoder PAL chip for the standard Z-150/160 memory card (not for the Z-157/58). With this PAL and the 18 pieces of 256K RAM chips (included), you will expand the memory on the card to 640K or 704K. ZP640+/18 Kit.....\$59.00. PAL chip only ZP640+.....\$18.00

Z-150 SERIES HARD DISK DRIVE KIT

⇒ These kits include high speed Seagate drives with autopark heads. Each kit includes all cables, hardware and instructions to mount the hard drive under your two floppy drives in your Z-150 series computer.

* ST-125/Z150 Kit	21 Meg, 40 MS,	\$281.00
* ST-138/Z150 Kit	32 Meg, 40 MS,	\$329.00
* ST-251/Z150 Kit	42 Meg, 28 MS,	\$341.00 SALE PRICED

Z-148 HARD DISK DRIVE KIT

⇒ Includes the hard disk drive and a Z-148 compatible controller together with the Z-148 Expansion Card described below. All required cables, hardware and instructions are included for you to replace one floppy with a Seagate Hard Drive in your Z-148. Add only \$30.00 the the following price if your would like us to include a SmartWatch.

* ST-125/Z148 Kit	21 Meg, 40 MS,	\$352.00
* ST-138/Z148 Kit	32 Meg, 40 MS,	\$399.00
* ST-251/Z148 Kit	42 Meg, 28 MS,	\$408.00 SALE PRICED

Z-148 EXPANSION CARD

⇒ Adds one full length and one half length IBM expansion slot to your Z-148 for hard drive controller, video card, modem, etc. ZEX-148.....\$79.00

Z-150 VIDEO ELIMINATOR

⇒ For the Z-150 or Z-180 only. Not required for the Z-157/158/159 computers. A small piggyback board which replaces the scratch pad memory on your current video card. This allows the removal of the original Zenith video card and replacement with an EGA, VGA or any other 8 bit video card. Order VCE-150 \$54.00

2400 BAUD MODEMS

⇒ Fully Hayes compatible 2400/1200/300 Baud with Software. Internal \$84.00 External Model \$128.00 Cable for External Modem \$8.50



15718 SYLVAN LAKE * HOUSTON TX 77062

PHONES: ** ORDERS AND INFO (713) 486-0687 ** FAX: (713) 486-8994 **

PAYLOAD * PAYLOAD * PAYLOAD * PAYLOAD

⇒ MICROSOFT WINDOWS SOFTWARE version 1.04 for PC (not Z-100)
 ⇒ FREE with any order from this ad. Just ask for it and add \$5.00 for shipping and handling. Includes 5 disks and 300+ page manual. Offer good for limited time.

WESTERN DIGITAL HARD DISK CONTROLLER CARDS

⇒ WD XT GEN2+ 8 BIT, MFM, DUAL HARD DRIVES, XT COMPUTERS	\$48.00
⇒ WD1004-27X 8 BIT, RLL, DUAL HARD DRIVES, XT COMPUTERS	\$53.00
⇒ WD1006V-MM2 16 BIT, DUAL HARD, DUAL FLOPPY, 1:1 AT	\$99.00
⇒ WD1006V-SR2 16 BIT, RLL, DUAL HARD, DUAL FLOPPY, 1:1 AT	\$109.00
⇒ HARD DRIVE CABLE SETS, XT.. \$3.50, AT .. \$4.50, AT + Floppy \$6.50	

PRINTERS

⇒ STAR NX-1000 II 9 PIN, 180 CPS, 45 CPS NLQ,	\$179.00
⇒ SEIKOSHA SL-90 24 PIN, 240 CPS, 80 CPS NLQ, PAPER PARKING	\$314.00
⇒ PANASONIC 1180 9PIN, 192 CPS	\$204.00
⇒ PANASONIC 1124 24 PIN, 192 CPS	4335.00
⇒ PANASONIC 1624 24 PIN, 192 CPS, WIDE CARRAGE	\$422.00
⇒ PANASONIC 1695 9 PIN, 288 CPS, WIDE CARRAGE	\$458.00
⇒ PRINTER CABLE	\$12.00

FLOPPY DISK DRIVES

⇒ MITSUBISHI MF501	5.25" 48 TPI DS/DD 320K/360K	\$ 68.00
⇒ MITSUBISHI MF504	5.25" High Density 360K/1.2 MEG	\$ 81.00
⇒ MITSUBISHI M-353	3.5" in 5.25" frame 720K	\$ 84.00
⇒ MITSUBISHI M-355	3.5" in 5.25" frame 1.44 MEG	\$ 94.00
⇒ TOSHIBA ND352	3.5" with 5.25" frame 720K	\$ 74.00
⇒ TOSHIBA ND356	3.5" with 5.25" frame 1.44 MEG	\$ 79.00
⇒ M-355 and ND356 run on AT compatible or special controller only.		

PAYLOAD CUSTOM ASSEMBLED COMPUTERS

⇒ We assemble 8088 XT, 80286 AT, 80386SX or 80386 IBM compatible computers to your specifications. Please write or call for a work-up sheet showing items available and prices.

VIDEO MONITORS

⇒ ZCM-1492	ZENITH Color Flat Screen VGA	\$ 679.00
⇒ MA2565	SAMSUNG Amber TTL 720x350	\$ 89.00
⇒ CW4656	SAMSUNG Color RGB 640x200	\$ 223.00
⇒ CM4592	SAMSUNG Color EGA 640x350	\$ 339.00
⇒ CJ4681	SAMSUNG VGA 720x400	\$ 360.00
⇒ CVB4581	SAMSUNG Multi-sync VGA 1024x768	\$ 429.00
⇒ CM1440	SEIKO VGA 1024x768 .25 dot	\$ 569.00

VIDEO CARDS

⇒ BASIC VGA	PARADISE AUTOSWITCH 640x480	\$ 99.00
⇒ VGAPLUS	PARADISE AUTOSWITCH 800x600	\$ 146.00
⇒ VGAPLUS16	PARADISE AUTO 16 BIT 800x600	\$ 177.00
⇒ VGA 1024-256k	PARADISE AUTO 1024x768	\$ 194.00
⇒ VGA 1024-512k	PARADISE AUTO 1024x768	\$ 247.00

MEMORY CHIPS, ETC.

⇒ Memory chips are once again at reasonable prices. The market prices have been changing daily, therefore we are only able to list estimated prices. Please call for the current price before placing your order. We buy in large quantities and work on the smallest of margins in order to bring you great values.

⇒ 41256 256x1 80 ns.....\$2.95	SIP 1Mx9 80 ns\$68.00
⇒ 41256 256x1 100 ns\$2.65	SIM 1Mx9 80 ns\$69.00
⇒ 41256 256x1 120 ns\$2.30	SIM 256x9 80 ns\$24.00
⇒ 1Mbit 1Mx1 80 ns\$7.40	V-20 replaces 8088 CPU \$14.00

Z-100 SERIES COMPUTER UPGRADES

High Density 1.2 Meg Drives

⇒ External floppy drive set-up complete with drive, power supply, case and cables. Ready to connect to your 8" floppy controller. Single Drive Unit \$207.00
 ⇒ Dual Drive Unit \$299.00 Bare drive and cable for internal mount \$115.00

ZMF100A by FBE Research

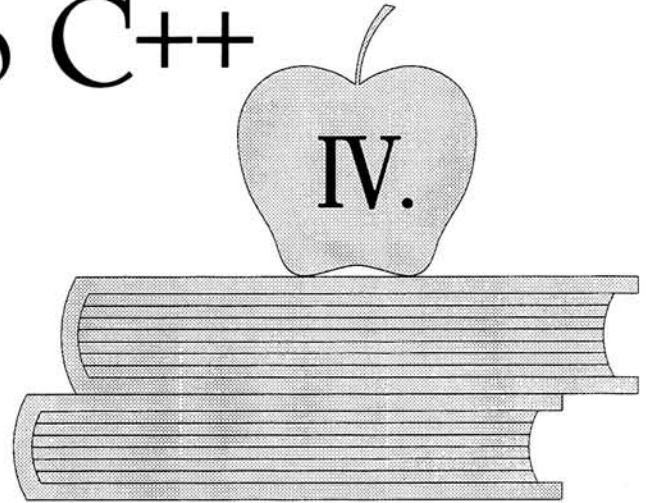
⇒ A modification package which allows 256K chips to be used on the old-style motherboard (part number85-2653-1) to reach 768K. Simple assembly with no soldering or trace cutting. Compatible with Easy PC and Gemini Emulator. Order 27 256x1 RAM chips to complete this kit. ZMF100A.....\$60.00

⇒ Please Mail, Phone or FAX your order. All hardware carries the manufacturers warranty plus the PAYLOAD 90 day guarantee. No surcharge on credit card orders. COD shipments on request. Add \$5.00 to all prepaid orders for handling and shipping in the Continental USA, we pay the balance. Actual shipping costs for foreign, overseas and net billing orders. We accept purchase orders from schools, government and approved accounts. Mail or phone your order for prompt friendly service. Texas residents please add 8.25% state sales tax.

Introduction to C++

Fourth Installment

Lynwood H. Wilson
2160 James Canyon
Boulder, CO 80302



The Fourth Installment

I've been avoiding the hard parts. I'm not apologizing, after all I warned you in the beginning that this series of articles would follow my own progress through the Object Oriented Jungle rather than containing the distilled wisdom of a Guru who always knew. As a result of this process, there is still some question in my mind about what it is all for.

I will tell you more about how to design OO programs, what all this Object stuff is FOR as soon as I know more, and as soon as I've brought you up to speed on the basic language. But for the meantime, here are a few tidbits to hold you over.

OOP began as a tool for simulation. One explanation of the general usefulness of the object languages is that much of what we do is simulation even when we don't think of it like that. A bookkeeping program, for example, is a simulation of some of the financial activities of a company.

One way to make it easier to think about is to think of it as representation rather than simulation. When we read about data structures we think of them as representing something. Data can represent only the state of something, however. Most of the things we want to represent (model) in our programs are not static. In order to represent the change, we write code which acts on the data and changes it. This data is unnaturally separated from the data it modifies. An object, on the other hand, can represent both the state and the activity of the artifact being represented (modeled).

My impression at this stage is that OOP will help us write better programs in less time, reuse more of our code, debug easier, and all that may be less important in the long run than the new perspective it gives us, the new way of mapping a program onto a problem. More on this as I learn more.

Data is. Functions (code) do. Objects both are and do.

Resources

For those of you who would like to read ahead, I recommend Object Oriented Program Design by Mark Mullen. This is a practical sort of book which chronicles the design and construction of a bookkeeping program in C++. It has a lot of gritty details and practical advice. Mr. Mullen has designed and written real programs. More useful for us, he has apparently designed and written programs that didn't turn out so well, and he doesn't want to do it again. I liked it.

Mr. Mullen has a newer book called Rapid Prototyping for Object-Oriented Systems which is just as good, but perhaps a bit less useful. In this book he proposed that programs should be prototyped in an Object Oriented language such as Smalltalk because it has a large number of pre-defined objects. After the prototype is developed to a satisfactory point the program can be translated into a more efficient language such as C++.

I strongly agree with the idea of building a prototype of a program. Nothing can help you learn what the customer wants and needs as well as a prototype which you and the customer have developed together and finally settled on. And an object language will help by allowing you to change parts of the prototype during this process without having to rewrite it all. At least it's better.

Mr. Mullen recommends using Smalltalk rather than C++ because of the large standard library of objects provided in the Smalltalk environment. I am sure he is right that getting the prototype running and changing it round both go more quickly, but when you are done with the prototype, there you are translating it into C++ and writing C++ versions of all those useful Smalltalk objects. It seems that the time

gained in prototyping will be lost in translation. And soon you will have a good library of C++ objects, so you would not need Smalltalk in any case.

I must admit I have never programmed in Smalltalk, and could easily be wrong here.

Upon re-reading this, it sounds like I'm knocking it because I don't understand it, so I am going to order a copy of Smalltalk. More on that when it comes. Or when I find the time to play with it.

In any case, I have learned a lot from Mr. Mullen's books about designing OO programs. I recommend them both.

The Rapid Prototyping book came at a very good time for me. I am currently working on the prototype phase of a project of substantial size. The program will depend on its user interface for its success to an even greater extent than most, and a prototype seemed essential.

I read a bit of the manual for Dan Bricklin's Demo Program, but it seemed as impenetrable as ever.

My client had supplied me with a C based screens and forms package called C-scape and with it a screen generator called Look and Feel. This turned out to be a very effective prototyping tool. Look and Feel lets you design a screen quickly and include data entry fields with various kinds of simple data checking. Then it translates the screen into the C code to generate it, using the C-scape library. Most of the complex processing is left out of a prototype, but if you need to you can drop back into C and do it and then call another screen. It goes very quickly and when the prototype is complete and accepted by all concerned, you have the user interface code running.

With this system or any other, it is important to be very clear in the demarcation between the user interface and the rest of the program. It is good to define very carefully all the interfaces between the sections of a program, but it is especially true in the case

of the user interface.

C-scape is written in C, but it has an OO feel about it. And you can call C functions from C++ code, which I plan to demonstrate as soon as we have covered the basics of C++.

The manuals are only medium bad. The next version, which will be out "Real Soon Now", will be compilable under C++.

It's a week later and I have lost just a bit of my enthusiasm. C-scape makes it easy to do almost what you want to do, but if you want it just exactly the way you want it, you are back to writing the code for it, just like always. It still saved me time on the original prototype.

I see it's past time for class to start. Someone should have said something.

Functions

Some time ago during the first software productivity panic (this is the second) programmers discovered that it was a great deal easier to write small programs than large ones. Not just easier overall, easier per line. A large block of code involves so many things and so many interrelationships between the things that it is much more difficult to write than ten blocks of code each one tenth the size. This may seem obvious to you, but it was news at the time.

This discovery led to the development of new techniques and approaches such as structured programming, top down design, and stepwise refinement. These are all based on simplifying the programmer's job by allowing him to deal with complexity a level at a time. This means he can work on the overall operation of the program at the highest level without having to consider the details of the implementation and just as important, he can work on the details at the lowest level without having to consider the overall operation of the program. If necessary, different people can even work on different parts of a program.

It was a revelation. The business took a big jump forward, programs got easier to write and/or bigger, and the computer world we see today is a result of this breakthrough. And it's all dependent on functions.

The foundation of structured programming is Structured Design by Larry Constantine and Edward Yourdon. Good, although a bit dense.

My favorite book on the subject is A Practical Guide to Structured System Design by Meilir Page-Jones. He worked with Yourdon and is an excellent writer.

And for a look into the time of the transition to structured programming, The Mythical Man-Month by Frederick Brooks. Brooks led the team which designed the IBM 360, and the team which wrote the operating system for it. Here is what he learned. The fact that it normally takes 9

months to produce a child in no way implies that 9 women can do it in 30 days. Not a surprising conclusion, but there are those who are not yet convinced.

The Simplest Kind of Function

Here is a program which contains a very simple function.

```
#include <iostream.h>

void prompt_quit(void);      // function prototype

main()
{
    char ch;

    do {
        prompt_quit();      // function call
        cin >> ch;
    } while(ch != 'Y')
}

void prompt_quit(void)      // function definition
{
    cout << "Would you like to quit now? (Y/N) ";
}
```

Note that the function `prompt_quit()` is exactly the same in structure as the function `main()`. Main is just like any other function except that when the program is run it begins executing with `main()`, and when the last line of `main()` is executed the program ends.

The program begins with the usual include. Next is a prototype (or declaration) for the function `prompt_quit()`. The first void indicates that the function does not return a value. Then comes the name of the function. The void in the parentheses indicates that the function is not passed any parameters. The prototype ends with a semicolon which indicates that it is a prototype rather than the definition of the function.

Note that an identifier with parentheses after it is nearly always a function name. The only exception I can think of is `return`, which I will describe shortly.

In `main()` the function `prompt_quit()` is called. Calling a simple function like this just requires its name.

After `main` comes the definition of the function `prompt_quit()`. The first line of the definition is the same as the prototype except that it lacks the semicolon. This tells us and the compiler that it is the definition rather than a prototype or declaration.

There is a bit of potential jargon confusion here. Old C had no function prototypes and what I called the definition of the function was usually called its declaration. Now we often call the prototype the function declaration and we call the actual function code the function definition. Be alert for misuse of these words by old time C programmers. Me, for instance.

When this code runs it starts as always with `main()`. First it declares a character variable named `ch`. This is a traditional name for an all purpose "char var". Then a

do-while, just the thing when we know we must execute the code at least once. Then we call our function. We send it no parameters, and it returns nothing. Then we read in a character, and if the character is not a capital Y, we repeat. If the character is a 'Y' the program ends.

Note that a character literal consists of

the character inside single quotes.

Functions That Return a Value

A function can also return a value. For example, another way of writing this program would be to replace the stream input with a library function which gets a single character from the keyboard and returns its value. Here is the loop from the above program. The rest of the program is the same as before.

```
do {
    prompt_quit(); // function call
    ch = getche(); // another func call
} while(ch != 'Y');
```

When a function like `getche` returns a value, the function name takes the value as though it were a variable and the value can be assigned to a variable of the appropriate type as in the fragment above. If the value is not assigned, it is lost.

When using a library function such as `getche()` we do not usually write a prototype for it as we did for `prompt_quit`. Prototypes for the library functions are supplied with the compiler in the form of header files. In the case of `getche()` we need only add this include to the head of the program: `#include <conio.h>`. `conio.h` has prototypes for all the console I/O functions. Check your run-time library reference to see which include files you need for the library functions you use.

Here is another permutation of the program. (See Figure 1.)

In this version the user input goes to the function, which returns it to `main()`. The function and its prototype are changed to show that the function returns a character.

In the function a character is collected from the keyboard by the library function `getche()`, and that value is used by `return`. `Return` is not really a function, although it looks like one. It causes the function it is in to terminate and return control to the code

which called it. If a value or an expression which evaluates to a value follows return, that becomes the value returned by the function.

The parentheses after return are optional. `return(getche());` and `return getche();` are the same.

In `main()` the value returned by `prompt_quit()` is assigned to the character variable `ch` which is then tested to see if it's time to quit.

In another variation, seen below, the function `prompt_quit()` decides whether the user wants to quit or not, instead of just returning the character the user enters.

This follows the principle of writing functions which do a single simple job. This is a more complex function but in a sense it does a simpler job. "Find out whether the user is ready to quit" is a simpler job description than "Ask the user if he is ready to quit and then return whatever character he enters". In the second case the job of finding out whether the user is ready to quit is divided between two functions.

If it is difficult to describe a function's job simply without ands and multiple sentences, perhaps it's doing several jobs. (See Figure 2.)

Note that we can get away with returning 1 and 0 when we promised we would return a char. Then we assign the value to a character variable. C++ is fairly permissive in its view of data types, although not as permissive as C.

Note also that the character variable `ch` in `main()` is a separate and different thing from the character variable `ch` in the function `prompt_quit`. The `ch` in the function first comes into existence when the function is called and the code which declared `ch` is executed. At that time memory is allocated for the variable. The variable's scope, the code over which it exists, extends from the declaration to the end of the function. When the function ends the variable ends, too, and its memory is made available for reallocation to some other purpose.

Here is another version, reduced to a form which is a bit cryptic, but quite often used. (See Figure 3.)

Main is reduced to a while loop with a null body. It runs and calls `prompt_quit` until `prompt_quit` returns a non zero value.

Remember the ! (not) makes 0 look like 1 and anything but zero look like 0. In other words, it turns true to false and false to true.

The logic of the function is now all in one line. The call to `getche()` fetches a character from the keyboard which is assigned to `ch`. The value of the assignment expression (`ch = getche()`) is compared to the character `Y`.

The value of an assignment expression is the value being assigned.

If it is true, the character from the keyboard is `Y`, then the or statement is known

```
#include <iostream.h>
#include <conio.h>

char prompt_quit(void);

main()
{
char ch;

do {
ch = prompt_quit();
} while(ch != 'Y');
}

char prompt_quit(void)
{
cout << "\nWould you like to quit now? (Y/N) ";
return(getche());
}
```

Figure 1

```
#include <iostream.h>
#include <conio.h>

char prompt_quit(void);

main()
{
char ch;

do {
ch = prompt_quit();
} while(ch != 1);

}

char prompt_quit(void)
{
char ch;

cout << "\nWould you like to quit now? (Y/N) ";
ch = getche();
if(ch == 'Y' || ch == 'y')
return(1);
else
return(0);
}
```

Figure 2

```
#include <iostream.h>
#include <conio.h>

char prompt_quit(void);

main()
{
while(!prompt_quit())
; // the body of the while
}

char prompt_quit(void)
{
char ch;

cout << "\nWould you like to quit now? (Y/N) ";
return((ch = getche()) == 'Y' || ch == 'y');
}
```

Figure 3

to be true and therefore evaluates to 1 which is the value returned to the calling code.

If it is false, the character from the keyboard is not Y, then the second half of the if is evaluated. If it is true, the character is y, then the if is true and the function returns 1.

If the character is neither Y nor y then the if evaluates to false or 0, and the function returns 0.

This may seem overly terse and difficult to read, but it is worth puzzling through. You will see a lot of code like this, particularly from the experienced C programmers.

Functions That Take Parameters

```
#include <iostream.h>
#include <conio.h>
```

```
char prompt_quit(char quit_char);
```

```
main()
{
    while(!prompt_quit('Q'))
    ;
}
```

```
char prompt_quit(char quit_char)
{
    cout << "\nPlease type " << quit_char
    <<
        " if you would like to quit now. ";
    return(getche() == quit_char);
}
```

In this version main() passes to prompt_quit() the character to test for. In both the prototype and the definition of prompt_quit() there is a declaration of the parameter in the parentheses after the function name.

The prototype must have the data types of any parameters to the function, but it does not have to have the name. This prototype would be perfectly legal:

```
char prompt_quit(char);
```

But it is still a good idea to include the variable name to show what the variable is used for. Another reason for choosing descriptive variable names.

The definition of the function must have the variable name as well as its data type.

Note that the two character variables in the last two versions of prompt_quit are defined in different places. char ch; appears after the first curly brace in the function, where we expect variables to be defined. char quit_char appears in the parentheses after the function name.

These two character variables differ in only one respect, which is that quit_char gets its initial value from the parameter passed in by the calling code. Otherwise they are both local variables which may be changed in the function. They both go out of existence when the function ends.

In order to pass a parameter to the function, main() just put the value in the parentheses after the function name.

When the function runs, the first thing that happens is the allocation of memory

```
#include <iostream.h>

void line(int length);

main()
{
    int len;

    cout << "Please enter a length for the line.
";

    cin >> len;
    cout << '\n';
    line(len);
}

void line(int length)
{
    for(int i = 0; i < length; i++)
        cout << '-';
}
```

Figure 4

for quit_char and the assignment of 'Q' to that memory. Then ch is defined and the prompt is printed to the screen. Since the function is only looking for one character, the test is simpler and we don't need the other character variable. Instead, this function tests the value returned by getche() against the value of quit_char and returns the result, 1 if they match, 0 if they don't. In either case there is no further need for the value returned by getche(), so it is not assigned to a variable. (See Figure 4.)

Here is a different function, to which we pass a number as a parameter. The function then draws a line (with minus signs) from the location of the cursor to the right the length of the number passed to it. It uses a for loop and a counter variable, i, which it increments up to the value of length.

Length in the function line() is a separate variable from len in main(), just as the two variables named ch in the function and the main in our first example. When the function is called, main() evaluates the variable len and sends its value, not the variable

itself, to line() as its parameter. Line() assigns that value to the new variable length. But length is a new variable which is local to line() [its scope is the function line()] and cannot affect anything in main().

This is called parameter passing by value. Passing by reference is the other kind, where you are passing in the only copy of the variable, and any changes to it in the function show up back in the calling code.

You can pass parameters by reference in C++ if you choose, but I am going to leave that until we get to pointers.

Figure 5 is another version of Figure 4. The function line() is simplified or at least compressed. Instead of declaring a counter variable and incrementing it until it reaches the length, this version simply decrements the length until it reaches 0. The test is now simpler too, since it is the variable itself (false when it reaches 0) instead of a comparison between the variable and a terminal value.

And the fact that we have decremented length down to 0 by the time the function returns does not matter a bit back in main().

```
#include <iostream.h>
#include <conio.h>

void line(int length);

main()
{
    int len;

    cout << "Please enter a length for the line. ";
    cin >> len;
    cout << '\n';
    line(len);
    cout << "\nlen = " << len;
}

void line(int length)
{
    while(length-)
        cout << '-';
}
```

Figure 5

Continued on Page 34

dBASE III

Part 11

D. R. Cool
7421 Troy Manor Road
Huber Heights, OH 45424

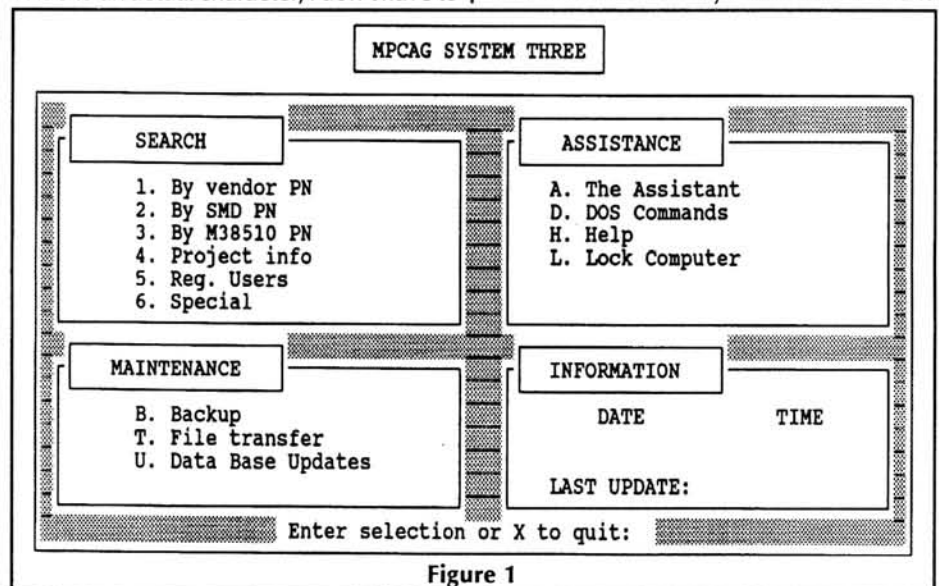
Advanced Techniques

In this article I will discuss some techniques that can be used to make your programs more professional in appearance. An example is a main menu generating program that displays the current time and date and uses the INKEY function for keyboard processing. Listing 1 is a segment of a control program that I wrote for an information system. The first command - "do MAIN_SCRN" - calls a procedure which paints the screen using the TEXT command. Figure 1 shows this screen as it is generated without the date and time. Note the use of the extended graphic symbols. These symbols can be generated by holding the ALT key and pressing in sequence on the key pad the decimal equivalent of the graphic symbol. For example, the shaded rectangle is obtained by holding down ALT while pressing the numbers 1, 7 and 7 in sequence on the key pad, then releasing ALT.

The system date is displayed at row 19, column 48 by a command within the MAIN_SCREEN procedure. The next command in Listing 1 starts the main menu loop. This is followed by initializing the variable KEY to zero. The DO WHILE loop which follows does two things - it checks continuously for input from the keyboard using the INKEY function and it displays the current time at row 19, column 63. The INKEY function returns an integer representing the most recent key pressed. If no key is pressed, it returns a zero; otherwise, it returns an integer corresponding to the ASCII code value of the key pressed.

The command "@ 23,53 ..." prevents the cursor from being continuously displayed following the time. Whenever a key is pressed, the value of KEY is compared against the input string "123456ABDHLTUXZ". This string represents all valid key inputs for this particular menu. Since I

use the CHR function to convert the KEY value to an actual character, I don't have to know the ASCII values. Also, the UPPER function converts any lower case value to



```
Listing 1

do MAIN_SCRN

do while .T.          && MAIN MENU LOOP

do LOCKOUT

KEY = 0
do while KEY = 0
  KEY = inkey()
  @ 19,63 say time()
  @ 23,53 say ""
  if upper(chr(KEY)) $ "123456ABDHLTUXZ"
    exit
  endif
  KEY = 0
enddo

do case

  case chr(KEY) $ "Xx"
    clear
    quit
```

upper case. If the key pressed matches one of the characters in the string, the program exits from the DO WHILE loop; otherwise, KEY is reset to zero and the loop continues.

Once the key-scanning loop is exited, a DO CASE structure processes the key and performs whatever program corresponds to the selection. For a numeric key, use the command

```
case chr(KEY) = "1"
```

whereas for a letter key you would use a command in the form

```
case chr(KEY) $ "aA"
```

to check for either upper or lower case. Of course, you want to make sure that the menu options correspond to the input string in the key scan loop. In other words, simply adding an option to the DO CASE structure isn't enough. The program will simply ignore that key unless it is added to the key input string.

The INKEY function can also be used for special keys, such as arrow keys and CTRL-HOME. For example, the right arrow key returns the value 4. (See the dBASE manual, INKEY function, for the values of special keys.) I have used this function in a document printing program to allow the user to select what portions of a document to print using the arrow keys. Figure 2 shows a portion of the print menu. The user can choose to print the cover page, the body of the document, the last page or any combination of the three. A right arrow points to the current selection. The arrow can be moved up or down using the up and down keys. Pressing ENTER highlights the selection; pressing ENTER again dehighlights it. Pressing END aborts the print menu and returns to the main menu. Finally, pressing CTRL-END starts the printing process.

Listing 2 shows the portion of the print program that processes the key input. Lines 1-6 initialize variables that are used in the print procedure. Lines 7-10 paint the screen display. The procedure DISPLAY called by line 9 is shown in Listing 3. This procedure displays the print selections in normal or reverse video, depending on the value of the FLAG_ variables. This procedure also displays the right arrow symbol (ASCII code 26) opposite the current selection.

Line 11 starts the main key scanning loop. The variable MKEY is set to zero prior to the key scanning loop which begins at line 13. When a key is pressed, the program exits from this loop to a DO CASE structure which performs a function based on the value of MKEY.

If the user presses the UP cursor key, MKEY equals 5 and the variable ARROW_POS is decremented by one. In the event that ARROW_POS becomes less than 1, it is assigned the value of three. This recycles the arrow to the third selection. Likewise, if the user presses the DOWN cursor, ARROW_POS is incremented by

Listing 2

```

1 store .F. to FLAG_CP      && Flag to indicate print cover page
2 store .F. to FLAG_BD      && Flag to indicate print body of doc
3 store .F. to FLAG_LP      && Flag to indicate print last page
4 store 1 to ARROW_POS       && Position of indicator arrow
5 store 12 to D_ROW          && Row where display starts
6 store 10 to D_COL          && Column where display starts

7 @ 8,10 say "PRINT SPECS AND DRAWINGS IN PROCESS"
8 @ 10, 0 say "Use cursor keys and <ENTER> to select portion to print: "
9 do DISPLAY
10 @ 16,10 say "Press END to return to menu, Ctrl-END to print."

11 do while .T.
12   MKEY = 0
13   do while MKEY = 0
14     MKEY = inkey()
15   enddo
16   do case
17     case MKEY = 5          && CRSR UP key
18       @ D_ROW + ARROW_POS - 1,D_COL - 2 say " "
19       store ARROW_POS - 1 to ARROW_POS
20       if ARROW_POS < 1
21         store 3 to ARROW_POS
22       endif
23     case MKEY = 24         && CRSR DN key
24       @ D_ROW + ARROW_POS - 1,D_COL - 2 say " "
25       store ARROW_POS + 1 to ARROW_POS
26       if ARROW_POS > 3
27         store 1 to ARROW_POS
28       endif
29     case MKEY = 13         && ENTER key
30       do case
31         case ARROW_POS = 1
32           store .not. FLAG_CP to FLAG_CP
33         case ARROW_POS = 2
34           store .not. FLAG_BD to FLAG_BD
35         case ARROW_POS = 3
36           store .not. FLAG_LP to FLAG_LP
37         endcase
38     case MKEY = 6          && END key
39       return
40     case MKEY = 23         && Ctrl END
41       exit
42     otherwise
43       ?? chr(7)
44     loop
45   endcase
46   do DISPLAY
47 enddo

```

one. Again, if ARROW_POS takes on a value greater than 3, it is assigned the value of one.

For the ENTER key (MKEY = 13), the action depends on the value of ARROW_POS. In each case, a logical variable is toggled from TRUE to FALSE or FALSE to TRUE. The END key (MKEY = 6) causes the program to return to the main

menu. Finally, CTRL-END (MKEY = 23) causes the program to exit the main loop and begin the printing process.

This is a very minimal structure for a display of this type; however, these basic ideas can be applied to create a much more complex menu scheme involving a large number of key combinations. You are limited only by your imagination and

PRINT SPECS AND DRAWINGS IN PROCESS

Use cursor keys and <ENTER> to select portion to print:

```

->COVER PAGE
  BODY OF DOC
  LAST PAGE

```

Press END to return to menu, Ctrl-END to print.

Figure 2

Listing 3

```

procedure DISPLAY
@ D_ROW + ARROW_POS - 1, D_COL - 2 say chr(26)
if FLAG_CP
  set color to n/w
  @ D_ROW, D_COL say " COVER PAGE "
  set color to
else
  set color to
  @ D_ROW, D_COL say " COVER PAGE "
endif
if FLAG_BD
  set color to n/w
  @ D_ROW + 1, D_COL say " BODY OF DOC "
  set color to
else
  set color to
  @ D_ROW + 1, D_COL say " BODY OF DOC "
endif
if FLAG_LP
  set color to n/w
  @ D_ROW + 2, D_COL say " LAST PAGE "
  set color to
else
  set color to
  @ D_ROW + 2, D_COL say " LAST PAGE "
endif
return
  
```

programming skills.

It is possible to even simulate "pop-up" menus. Listing 4 is a short procedure file that demonstrates this technique using different colors for the main and pop-up menu. The program initially sets the colors to white on blue, then calls the MAIN_MENU procedure, which uses the TEXT command to display the main menu. If the user chooses item 2 of the main menu, the program then clears the screen from row 10, column 40. A report menu is then generated using red on white. Returning to the main menu by selecting item 4 causes the program to clear the screen and redraw the main menu.

An important point to note in any program that paints different parts of the screen with different color combinations is that the SET COLOR TO commands must precede any CLEAR commands or the results will be totally unexpected. To see what I mean, in the main loop, relocate the SET COLOR TO command after the CLEAR command and run the program, bringing up the reports menu and then returning to the main menu. Not exactly what you would want, is it?

In my next and last article, I will discuss the LOAD and CALL commands, which are almost totally ignored in most books on dBASE III.

Listing 4

* PROCEDURE FILE TO DEMONSTRATE "POP-UP" MENUS
 * WRITTEN BY: D.COOL 07/05/90

```

procedure MAIN

set scoreboard off
set talk off
set bell off

do while .T.
  set color to w/b
  clear
  do MAIN_MENU

  store " " to MAIN_OPT
  set intensity off
  @ 16,27 get MAIN_OPT picture "!"
  read

  do case

    case MAIN_OPT = "1"
      do INVOICES

    case MAIN_OPT = "2"
      set color to r/w
      @ 10,40 clear
      @ 10,40 say "-----+"
      @ 11,40 say "|          REPORTS MENU          |"
      @ 12,40 say "|          |"
      @ 13,40 say "|          1. Print report A.      |"
      @ 14,40 say "|          2. Print report B.      |"
      @ 15,40 say "|          3. Print report C.      |"
      @ 16,40 say "|          4. Return to main menu. |"
      @ 17,40 say "|          |"
  
```

Quality Enhancements!

EaZy PC Products

EZM-128: Expand 512K base memory to 640K. Simple, plug-in installation. \$125.00
EZCLOCK: Calendar/Clock. Piggy-back add-on for EZM-128. \$35.00

No Slot Clock/Calendar

FBE SmartWatch: Automatic date/time on bootup. Installs under BIOS/Monitor ROM. Ten year battery. Works with all Heath/Zenith MSDOS computers. For PC's \$32.00, Z-100 \$33.00 Module: \$25.00

H/Z-148 Expansions

ZEX-148: Adds one full-size and one half-size expansion card slot. \$79.95
ZP-148: PAL chip expands existing 640K memory to 704K. CGA/MDA only! \$19.95

Configuration Control

CONFIG MASTER: Menu-select active CONFIG.SYS during bootup. Software for PC/Z-100 MSDOS. \$29.95

H/Z-150 Items (Not for '157, '158, '159)

VCE-150: Eliminate video card. Install EGA or VGA card. All plug in. Includes circuit board, SRAM and RM-150. \$49.95
ZP640 PLUS: PAL chip to expand standard memory card to 640/704K with 2 banks of 256K RAM chips (not included). \$19.95
ULTRA-PAL: Three PAL chips: MR150 for 704K + 512K RAM Disk; MR150T for 640K + 512K RAM Disk; LIM150 for 640K + 512K (32 pages) of simulated v3.2 Lotus/Intel/Microsoft Expanded Memory. With software. Install on standard memory card. No soldering. Needs 45 256K RAM chips (not included) for maximum memory configuration. \$39.95
COM3: Change existing COM2 to COM3. Put internal MODEM at COM2. Don't lose serial port. With software. \$29.95

H/Z-100 Modifications

ZMF100A: Expand "old" motherboard (p/n 181-4917 or less) using 256K RAM chips (not included). No soldering. \$65.00
ZRAM-205: Put 256K RAM chips on your Z-205 board. Get 256K plus 768K RAM disk. Contact us for data sheet before ordering. Without RAM chips. \$39.00

H/Z-89 Add-Ons

H89PIP: Parallel printer 2 port interface card. With software. \$50.00 Cable \$24.00
SLOT4: Add fourth expansion slot to right-side accessory bus. \$39.95

Order by mail, FAX, telephone, or see your dealer. UPS/APO/FPO shipping included. VISA/MasterCard. WA residents add 8.1% tax. Hours: M-F 9-5 PST. We return all calls left on our answering machine!

FBE

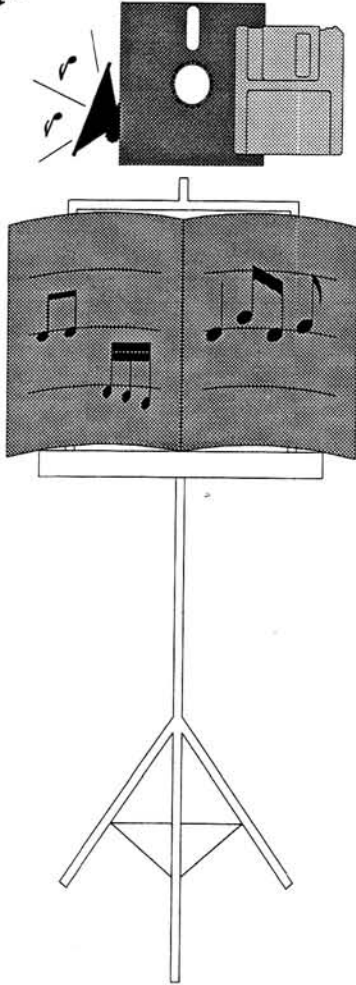
FBE Research Company, Inc.

P.O. Box 68234, Seattle, WA 98168

206-246-9815 Voice/FAX TouchTone Selectable

Reader Service #104

New Software Product!



The Electronic Clavier
P/N 885-6016

Listing 4 (Con't.)

```

@ 18,40 say "| Select:
@ 19,40 say "+-----+"

do while .T.
  store " " to PRINT_OPT
  @ 18,50 get PRINT_OPT picture "!"
  read

  do case

    case PRINT_OPT = "1"
      do PRINT_A
      exit

    case PRINT_OPT = "2"
      do PRINT_B
      exit

    case PRINT_OPT = "3"
      do PRINT_C
      exit

    case PRINT_OPT = "4"
      exit

    otherwise
      ?? chr(7)
      loop

  endcase
enddo

case MAIN_OPT = "3"
  exit

endcase
enddo
set scoreboard on
set bell on
set talk on
return

procedure MAIN_MENU

@ 5,0
text

          +-----+
          | MAIN MENU |
          +-----+

1. Generate invoices.
2. Print reports.
3. Exit.

Select:

endtext
return

```

RAPID INDEX: A computer program that lists titles of magazine articles by subject or in alphabetical order.

CPM or PC/MS-DOS 2.1 or later.

DRIVE:

COMPUTER: 5.25"DD...\$21.00
 Z-100 Models 5.25"HD...\$23.00
 IBM Compatible 3.5"DD....\$23.00

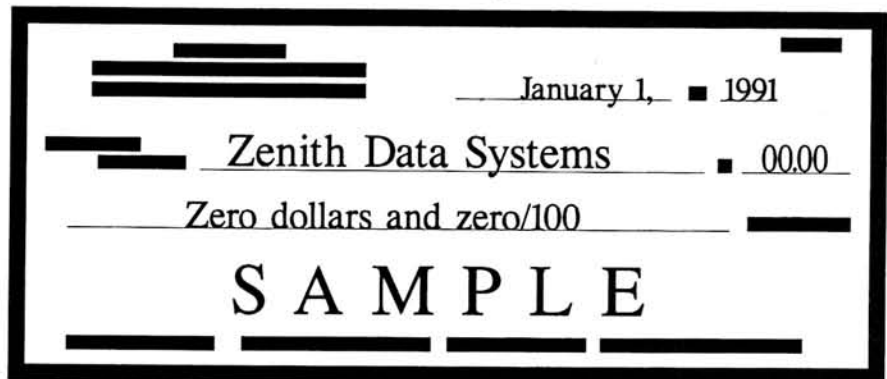
Magazines currently available:
 Elect. Service & Tech. (1984 - 89)
 REMark computer magazine (1985-89)
 Calif. residents add 6.75% sales tax.
 Send check or money order (US funds) only to:

B.U.D. Unlimited,
P.O. Box 503, Elverta, CA 95626.

REMark is a trademark of Heath. Zenith User's Group.

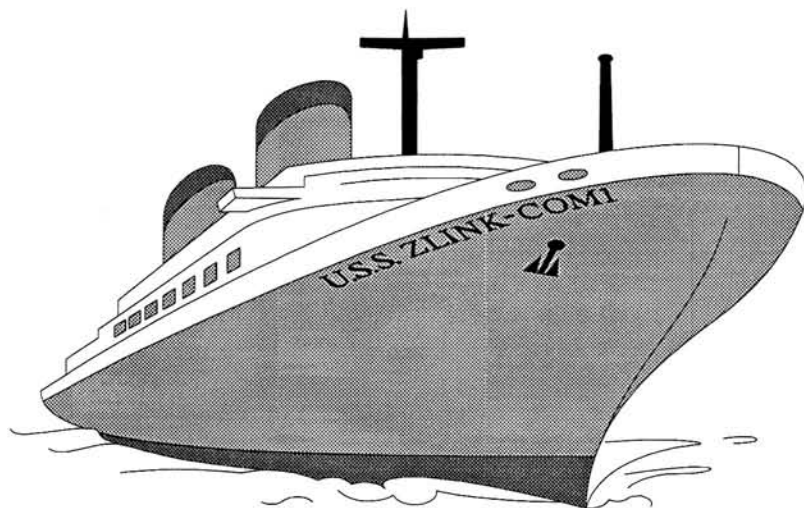
Reader Service #250

All Checks must be made out to
Zenith Data Systems



Port-O-Call:

COM1



**Laura White
759 Polfus
Benton Harbor, MI 49022**

As a member of the Zenith User's Group, you are privileged to purchase electronic equipment from the COM1's infamous "Bargain Centre." Recently however, the question has been asked whether or not this is indeed a privilege?

For those ZUG members who have never called the COM1 bulletin board, the bargain centre is designed to offer its subscribers electronic equipment at "unheard-of-prices." This equipment is not available for huggies to purchase at any time they choose; the bargain centre's sales happen randomly and with next to no warning, so for this reason, huggies have to be on their toes.

In addition, since the equipment is returned — end users as well as stores, huggies are made aware of the varying conditions of equipment prior to even making a purchase. Cited in the "information section" of the bargain centre you will find the following disclaimer, "as you can imagine, the testing of these units is minimal, and obscure problems can emerge at a later time . . . the fallout rate is about three to five percent."

The system operator also takes the time to inform huggies of the labeling system to better inform you of the equipment's condition. Guarantees and warranties are available, but on a limited basis depending upon the condition of the item. Equipment is available anywhere from factory new (NW), and includes a warranty, to equipment that is sold only for parts (JK), and has no warranty.

With these factors in mind, I will pose the question once again—is access to the bargain centre really a privilege? On the one hand, you are given the opportunity to

purchase electronic equipment at low prices, however, if you receive a unit that falls in the three to five percent fallout category, what do you do?

This seems to be the biggest draw-back to the bargain centre. Once you purchase the unit it's yours — no returns, and limited warranties and guarantees. It *could* be interpreted that bargain centre equipment comes with no support. You as a new owner might feel as if you're stranded on a desert island with your "new" computer, a printer and an empty bottle and no way of writing a message for help!

There is one very major flaw with this scenario. If you purchased your computer through the bargain centre, you have access to the COM1 bulletin board, regardless of whether or not you are an established member of HUG. You might not have the kind of support you are accustomed to — warranties, guarantees and factory repairs, however, you have an even better situation — you have the bulletin board. You would be amazed at how quick you get answers and the number of people who come to your rescue. In addition, word has it the Heath/Zenith dealers are most cooperative when it comes to getting the necessary parts to fix your problem. Quick results, cooperation with replacement parts, and a personal relationship with the maintenance person — this seems to be long way away from no support!

Yet another advantage to purchasing equipment through the bargain centre is its affordability. This simple fact exemplifies a situation where people who normally would not be in a position to get this equipment now are. This again adds to the list of resources you can tap into through COM1.

One huggie called in and informed us that he was able to ". . . place two ham radio TCP/IP switches on the air just because of the affordability . . ." He went on to say that he placed his bargain centre '2526 on the air as a full time packet BBS on a frequency of 5797.5 kilohertz to be used by volunteers with the U.S.Navy-Marine Corps. Military Affiliate Radio System (MARS). During the holiday season, this huggie's equipment gave sailors and marines participating in Operation Desert Shield the chance to send a message home. If the equipment weren't as affordable as it is, none of this could have happened.

So once again — is the bargain centre really a bargain, and does it give Zenith Data Systems a bad name? I think I can speak on behalf of the vast majority of huggies when I say that YES, the bargain centre is truly a bargain. After the question was placed on the bulletin board, I was totally amazed to see the resounding number of ayes. Except for the original huggie who stated that the bargain centre and Zenith Data Systems as a whole, wasn't offering him the support he felt he needed, (he outfitted his entire business with bargain centre equipment) no one had anything negative to say about the bargain centre.

Does it give Zenith Data Systems a bad name? Those who addressed this part of the question felt that if nothing else, the bargain centre gave ZDS a better name. This was concluded after huggies mentioned that they had no problem with their local Zenith Data Systems service center when it came to getting replacement parts.

It looks as if the consensus is that the COM1 bulletin board and its bargain centre

are like a library. There's an enormous amount of information available — you just have to know how to use the resources offered. Take some time and get to know COM1. Even if you only know how to get

into the message system, ask fellow huggies how to walk through the board. Believe it or not, huggies are very helpful!

I'd like to thank everyone who called in with their points of view. Fortunately for

the staff of REMark and ZUG, there are too many of you to mention here in this article. Take care, and always remember which port of call to ask for. ✨

Continued from Page 6

Key = _

Type "8,17" and press <Enter>. All I get on my screen is a solid square box, but the printed document contains the Greek letter itself. The "8,17" means character 17 in character set 8. Character sets are listed in Appendix P of the WordPerfect manual.

Since this process is a bit tedious, I have

created macros for the Greek letters I use frequently.

Summary

Version 5.1 of WordPerfect includes some significant new features. Spreadsheet hot links should be very useful to people who often need to include spreadsheet information in a word processing document.

The equation builder may render separate scientific word processors obsolete. Users who would rather use a mouse than the function keys will like the new drop down menus.

WordPerfect has taken an excellent product and made it even better. ✨

Continued from Page 28

The project I am working on should be quite educational. You don't know a language until you have written a fair sized program in it. I will have a few stories by next time, I am sure.

Sources

C-scape (includes source) \$399

C-scape with Look & Feel \$499
Oakland Group, Inc.
675 Massachusetts Avenue
Cambridge, MA 02139

The Mythical Man/Month by Frederic P. Brooks, Addison-Wesley, 1975.

Object Oriented Program Design by Mark Mullin, Addison-Wesley, 1990.

The Practical Guide to Structured Systems Design by Meilir Page-Jones, Prentice-Hall, 1988.

Rapid Prototyping for Object Oriented Systems by Mark Mullin, Addison-Wesley, 1990.

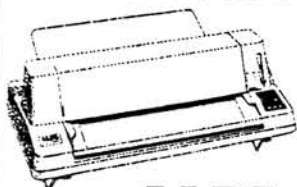
Structured Design by Edward Yourdan and Larry L. Constantine, Prentice-Hall, 1979. ✨

WS Electronics

(513) 376-4348 **** Since 1975 **** (513) 427-0287

1106 State Route 380, Xenia, Ohio 45385

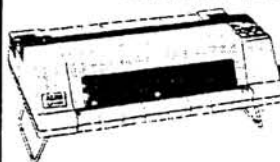
YOU AND YOUR BIG IDEAS.



- * 300 cps draft speed
- * Wide carriage
- * 24 pin printing
- * Front panel controls

ALPS Allegro 500XT™

THE ALPS ASP1600 PRINTER. COSTS VERY LITTLE, JAMS NOT AT ALL.



* Auto tear bar prevents paper waste. With the flick of a button, your fanfold paper advances to the perforation, then automatically returns to top-of-form position.

* Rugged 9-pin head delivers crisp output at 192 cps in draft mode, 38 cps in letter quality.

* Compact 9-lb. body makes for easy portability.

* Printer stand is built-in.

* Prints labels easily.

* Full Epson FX-85 compatibility.

ALPS
AMERICA

Built by popular demand.

SPECIAL

Z-315	EMS Kit for Z-159	\$ 10.00
Z-417	H D Controller Z-248	\$100.00
Z-317	H D Controller Z-150	\$ 75.00
ZCM-1400-1	Swivel Base for 1490	\$ 5.00
CB-4364-39	Diagnostics for 184	\$ 5.00
ENABLE 2.0	Word Processing	\$ 99.00

Quantities limited to stock on hand

ATTENTION:

Federal Government Offices
We stock ALL ALPS Printer Models and we stock ALPS PARTS and RIBBONS for all ALPS models including your P2000's and ASP-1000's

****GOVERNMENT DISCOUNTS OFFERED****

We are looking for good dealers.
ALPS Authorized Distributor and Service Center

Breaking the Churlish Bounds of DOS

Part 1

David W. Lind
RR 1, Box 3114
Bar Harbor, ME 04609

When IBM designers set the standards for their first personal computers, they limited the amount of memory available for applications programs to 640 kilobytes. The remaining 360 kilobytes of memory was reserved for use by the Basic Input/Output System (BIOS) and the Disk Operating System (DOS). The one megabyte address space limitation was imposed by the microprocessor that IBM chose - the Intel Corporation's 8088. Almost at once, business users of the IBM PC found that 640 kilobytes of memory was insufficient for some applications - particularly large spreadsheets. Independent developers, as well as IBM and its associates, searched for ways to overcome this limitation. This article is the first in a series of four which discuss the results of this search and show how to apply the results in applications programs to use memory beyond the one megabyte limit. These articles also show how to store parts of a program system on mass storage and call these parts into memory when needed. Such techniques permit the execution of almost any program, no matter how complex or large, on a personal computer.

The use of memory beyond one megabyte with Intel microprocessors is fraught with horrors. Some of this horror derives from confusing terminology, most of the rest from the determination by Intel Corporation engineers to make newer microprocessors compatible with earlier microprocessors. Fortunately, it is possible to understand and use the techniques which permit the addressing of memory above one megabyte without drowning in the morass of jargon and complex technology that make the techniques possible. Few articles or books I've read develop the techniques well. Too much space is devoted to technical discussion and quite often very important concepts are not explained well. This article and the three to follow emphasize the techniques and explain enough of the

concepts to use and understand the techniques. The reader will find more detailed discussions in the excellent references listed at the end of this article. In this article, some basic concepts are discussed and overlay techniques are introduced. Part two of this series completes the description of overlays. Part 3 discusses "expanded memory" techniques. The final article explains how to use "extended memory." Now to explain the basic concepts.

Basic Concepts

There are three basic means of overcoming memory limitations: overlays, expanded memory, and extended memory. Overlaying memory is a process by which part of a program is kept on a mass storage device, like a floppy or hard disk, and placed in memory when needed. Part of the program already in memory may be destroyed when the stored part of the program is placed in memory, hence the term "overlay." There are several variants of this process. Some are controlled by the operating system and are transparent to the applications programmer. Some such systems are called virtual memory systems. IBM has used them on main frame computers for years. Personal computers may have similar systems. If the applications programmer prefers to manage the overlays, that can be done. But, the overlays must be carefully controlled because they can destroy data outside the application program's space.

The advantage of overlays is that a program's length is limited only by the amount of mass storage available. Some disadvantages are that the process is slow and cumbersome. But, it may be the most economical means to execute large programs.

Another means to execute large programs is to add memory. Because the IBM PC and XT computers use the Intel 8088 microprocessors, one cannot directly address more than one megabyte of memory on these machines. One solution

to this problem is to use memory boards which contain circuits to address any amount of memory and transfer data to/from a specific segment of memory that the microprocessor in the computer can address. The memory transfer process is controlled by a special resident program called a "driver" which is normally installed by the CONFIG.SYS process when the computer is booted. The applications program controls access to a virtually unlimited amount of memory through the driver. The microprocessor and operating system only interface with the specific memory segment and have no direct control over memory above one megabyte. The memory above one megabyte in this type of system is called "expanded memory."

The advantage of expanded memory is that it is much faster than overlaying memory. Unfortunately, it requires special memory boards and software that is peculiar to a board design.

Soon after IBM began to market the PC and XT computers, Intel introduced the 80286 microprocessor. This computer chip could directly address 16 megabytes of memory in the "protected" mode of operation. To make this chip compatible with the earlier computer operating systems, it can also operate in a "real" mode which emulates the 8088/86 microprocessors exactly. Thus, in real mode, the 80286 microprocessor has the same memory limitations as the 8088/86 chips. Unfortunately, Microsoft DOS (MS-DOS) or Personal Computer DOS (PC-DOS) will operate only in real mode. Unless a protected mode operating system like OS/2 is used, an applications program cannot directly access memory above one megabyte. This limitation is also present in the newer 80386 and 80486 microprocessors.

The designers of the BIOS code in the 80286/386/486 microprocessor based computers recognized this limitation. Therefore, they provided means by which

one could directly access memory above one megabyte from MS-DOS and PC-DOS. This memory was named "extended memory" which some folks confuse with expanded memory. Keep in mind that expanded memory can never be directly accessed by the computer's microprocessor. Extended memory is directly accessed by the microprocessor. Of course, an 8086/88 microprocessor based computer like the IBM PC or XT cannot have extended memory, but can have expanded memory. Computers using the 80286, 80386, or 80486 microprocessors can have both expanded and extended memory.

The advantages of extended memory are speed of data access and the absence of special switching circuits on the memory boards. The disadvantages are that one must either access extended memory through a buffer in the memory space which can be addressed in real mode or one must switch the machine to the protected mode and lose all support from the disk operating system. An exception to this rule is the High Memory Area which can be directly accessed in real mode. This area is discussed in the last part of this series of articles. In addition, returning to real mode from protected mode is an esoteric process. Accessing extended memory through a real mode buffer is similar to the use of expanded memory and may be only slightly more efficient. Further discussion of extended memory will be left to the final article in this series.

The Execute Program Function

Before the reader can hope to understand the overlay process, some basic microprocessor characteristics should be reviewed. The disk operating system establishes a table called the "Interrupt Vector Table" beginning at address 0000:0000 in memory. The four-byte entries in this table are numbered from 0 to FFh (0 to 255). Each number corresponds to a specific subroutine in the BIOS or DOS code. Each of these subroutines perform standard functions, e.g. print screen data. The four byte entries are the segment:offset addresses of these subroutines. When a BIOS or DOS subroutine is called, it is normally called by number using the microprocessor's INT (interrupt) operand. The microprocessor multiplies this number by four to get the address in the interrupt vector table at which the address of the subroutine is found. This scheme allows the BIOS and DOS to be changed without having to change the application programs written for a previous BIOS or DOS version. It also allows the DOS to relocate its own subroutines or BIOS subroutines. Among the DOS interrupts is a subroutine which loads and executes programs or overlays. This interrupt is numbered 21h (21h is in the hexadecimal number system, 33 is the equivalent decimal number) and

is called the "Function Request" interrupt. It has numerous functions only one of which is to load and execute programs or overlays. The various functions are numbered. The "Program Execute" function, numbered 4Bh (75), is used to load and execute programs or overlays. The DOS uses this function for its own purposes, but this function can also be called by an applications program.

The Program Execute function has two modes of operation, which are available to application programmers, called "subfunctions." The first subfunction, numbered 0, loads and executes a new program without terminating the current program. The second subfunction, numbered 3, reads a segment of code or data from a mass storage device and places that segment in memory at the location the programmer designates. Subfunction 3 is dangerous because the memory location can be anywhere and all data at that memory location will be erased and replaced by the new segment. If the memory location happens to be the video display subroutines or data, the video monitor's circuits could be damaged. Thus, subfunction 3 must be used with great care. Subfunction 3 will be discussed further in the next part of this series of articles.

Parent and Child Processes

Subfunction 0 is far less dangerous and very useful. With this subfunction, one can call another executable program from a program already executing without terminating the first program or changing it. Essentially, one is using another program as a subroutine. The invoking program is called the "parent" process and the invoked program is called the "child" process. When the child process terminates, control reverts to the parent process, not to the disk operating system. While the child process is executing, it has access to all the files the parent has opened and can even pass a termination code to the parent. When the child process terminates, the memory space it used is released.

This technique can be very powerful. Assume one has a set of data which requires different processing depending upon some characteristic of the data. Also assume that the various programs used to process the data cannot be combined into one program because of memory limitations. One can create a parent program which determines the characteristics of the data and then calls only the executable programs necessary to process the data. This technique is often used by large scientific or engineering programs.

The assembly language program, LOADPGM.ASM, listed at the end of this article demonstrates the parent/child processes. It uses INT 21h-Function 4Bh-Subfunction 0 to call two executable programs. The first is COMMAND.COM which is the disk

operating system program loader and control program. The "dir" function of COMMAND.COM is used to display the current disk directory. Then the user is asked to select a file to be processed by the second child process which is a program the user entered when LOADPGM was executed. The processes repeat until the user terminates LOADPGM. This program is useful for repetitive processing as with computers used for a single purpose like word processing or spreadsheet applications.

Unfortunately, many people who attempt to use Function 4Bh are frustrated by their efforts. Some knowledge of the operating system is required. When any program is loaded into memory, the program loader in COMMAND.COM is loaded first. COMMAND.COM has a resident part which is always present after booting the system, but the program loader part is deleted after a program is loaded. There must be enough memory available to contain this program loader, the parent program, child program, and all resident programs. If memory is insufficient, the parent or child program must be down sized or some of the resident programs removed.

Memory is seldom insufficient. However, one operating system quirk will always present a problem. The operating system normally allocates all available memory to an applications program. Thus, a very small program may have hundreds of thousands of bytes of unused memory space. This unused memory space is where the operating system should load the program loader and child process, but it won't because the parent program "owns" the space. Before Function 4Bh is called, this unused memory space must be released to the operating system. Interrupt 21h-Function 4A-Set Block is used for this purpose. Do not attempt to use Function 4Bh-Allocate Memory for this purpose. The Allocate Memory function increases the size of the parent program, just the opposite of what one wishes.

Unfortunately, many descriptions of the Set Block function are misleading. The Set Block function attempts to size the memory a program uses. That is, it attempts to increase or decrease the program's allocated memory. Since the operating system has already allocated all the memory, with the exception of the memory used by the program loader, increasing memory is not normally attempted unless the program was previously down sized. To use Set Block, the number 4Ah is placed in the microprocessor general register ah, the number of paragraphs of memory requested is placed in general register bx, the extra segment register (es) must point to the Program Segment Prefix (PSP), and Set Block is then called by the INT 21h command.

Set Block uses two concepts with which the reader may not be familiar. The first and easiest concept is paragraphs of memory. The other is the PSP. Consider the easiest concept first. The disk operating system allocates memory in chunks of 16 bytes, called "paragraphs." All programs, data blocks etc. which are located by the operating system will begin and end at physical addresses divisible by 16. The reason is quite simple. The operating system only controls segment addresses. It leaves control of the offset addresses to the other system and applications programs. In real mode, the microprocessor determines a physical address in memory by shifting a segment register four bits to the left (equivalent to multiplying the segment register by 16) and adding the offset address to the result. Of course the microprocessor uses special registers for this purpose. The operating system and applications programs do not normally have access to this process. As a general rule, the operating system controls the values in the segment registers, which are the paragraph numbers, and the currently executing program controls the offset addresses. However, a program can override the operating system and set the paragraph numbers in the segment registers to access specific areas in memory. But, the operating system determines where the program and any other code or data actually resides in memory and allocates space in paragraph increments.

The Program Segment Prefix is a segment of code 256 bytes long which the operating system adds to the beginning of a program before the program is executed. When a program is loaded, the actual beginning address of the program is the beginning of the PSP, not the beginning of the code or data from the program file. As will be apparent, the PSP address is very important.

The PSP contains a variety of information important to both the operating system and the program. The first 24 bytes contain information on the size of the program and what to do when the program terminates. For example, at offset 0Ah one finds the address to which the program transfers when it terminates. At offset 16h is the segment (paragraph) address of the parent program's PSP. If the program is a parent program, this address is the address of the program's PSP. Thus, the operating system can look at the PSP to determine what to do when a program terminates, that is, return to the operating system or to a parent program. Following this information is some housekeeping data. At offsets 5Ch and 6Ch are unopened file control blocks which are used only by the earliest operating system versions. They should not be used with MS-DOS or PC-DOS versions 2.0 or greater. However, they are referenced by the Execute Program func-

tion. Starting at offset 80h is the Unformatted Parameter Area which is very important for use by the Execute Program function.

The Unformatted Parameter Area is also called the "Default Disk Transfer Area" or the "Command Line Buffer." It is 128 bytes long. The first byte, at offset 80h, contains a number which indicates how many characters (except redirection symbols) were typed as a part of the command line following the program name. These characters are then found in a string beginning at offset 81h. For example, if a file named, "WORD.EXE" were executed with the following command line:

```
WORD DATA.FLE, DATUM.FLE/p
```

the following American Standard Code for Information Interchange (ASCII) data would appear in the PSP for WORD.EXE beginning at offset 80h:

```
! DATA.FLE, DATUM.FLE/p
```

where "!" is the ASCII equivalent to the binary number 10101 which is the decimal number 21, the number of characters typed after the program name. Note that the blank after WORD is included. Only redirection symbols would not be included.

There is nothing unusual about the data in the Unformatted Data Area which is the reason it is called "Unformatted." Some folks believe that the data can only be file names and program switches which is the way this area is normally used. But any symbols, except redirection symbols and carriage returns (which terminate the command line), may be found in this area. It is up to the programmer to determine how these symbols are used if at all. This information will be useful when calling the Execute Program function. Returning to the Set Block function, one should now understand why Set Block requires the number of paragraphs of memory to be allocated and also requires the segment address of the PSP.

The segment address of the PSP is placed in the ds and es segment registers when a program executes. If the program has a COM extension (e.g., PROGRAM.COM), all the segment registers are set to the PSP segment address. In COM programs, the instruction pointer is set to 100h or 256. Readers familiar with such programs will recognize that the instruction "ORG 100h" sets the instruction pointer and leaves room for the PSP. Thus, the segment address of the PSP can be easily obtained in a COM program from any of the segment registers. In an executable program with an EXE extension, the segment address must be obtained from the ds or es segment registers before they are set to the data or extra segment addresses of the program. LOADPGM.EXE accomplishes all the processes requiring the PSP segment address before setting the es register, but after setting the ds register. Thus, the PSP segment address remains in es while data can

be moved to the program's data segment.

LOADPGM determines its size by subtracting the address of the PSP from the segment address of a dummy segment located at the end of the program. This data is moved to register bx before Set Block is called. When Set Block returns, the memory allocated to LOADPGM is reduced to its actual size. The released memory can be used for the child programs.

To use the Execute Program function, the path and program name of the child program must be located at the segment:offset address given by registers ds:dx and the segment:offset address given by es:bx must contain a parameter block. Of course, register ah must contain 4Bh and register al must contain 0. Then, the function is called with INT 21h.

The parameter block is 14 bytes long and contains the addresses of the environment, command line (for the child program), and file control block data. Normally, one is only concerned with the command line information. The first word of the parameter block is the segment address of the environment to be used by the child program. Usually, this word is set to zero which copies the parent's environment. The next two words (starting at offset 02h) of the parameter block contain the segment:offset address of the data to be placed in the child program's PSP Unformatted Parameter Area, that is, the command line data. This data must begin with a number corresponding to the number of characters in the command line, followed by an ASCII string of the command line data, and ending with the ASCII character for carriage return, 0Dh or 13. The remaining part of the parameter block contains the addresses of two file control blocks to be placed in the child program's PSP. Usually, this part of the parameter block is set to zeros. All except the earliest versions of the disk operating systems use file handles which obviate the need to pass file control blocks. All the files opened by the parent program are available to the child programs through the use of file handles.

LOADPGM copies its Unformatted Parameter Block to the data segment after re-allocating memory, but before setting the es register (remember, the es register contains the segment address of the PSP). The information in this block is assumed to be the program name of the second child program. When this process is complete, the es register is set to the address of the program's extra segment. The parameter block for the first child program is then set up.

The first child program is COMMAND.COM. This program is called to display a directory of files on the current disk. Note that the command line, at "cmddir" must begin with "/c" to indicate to the program that one wishes to execute

a specific function. Also ensure that the path is included in "stringd." Otherwise, COMMAND.COM may not execute. This process illustrates how to call a disk operating system function from an applications program. The ax register points to the parameter block and the dx register points to the path and file name. The ax register is set to 4B00h and the Program Execute function is called with INT 21h. When the child program is finished executing, it releases its memory space.

The second child program is the program file the user typed on the command line when LOADPGM was executed. The command line data for the second child program is data that the user is prompted to input, namely, a file for the second child program to process. The user is expected to select the file from the directory list, but any data is acceptable depending upon the requirements of the second child program. This child program is executed the same as the first except the pointers reference different data areas.

These processes are repeated, starting with the execution of the first child program, until the user requests termination. LOADPGM.EXE should execute on all IBM compatible computers and Zenith Z-100 series computers if the disk operating systems are at least version 2.0. The assembly language listing has many comments to prevent confusion. These comments need not be included in a source file the reader might produce.

Earlier in this article it was stated that termination codes may be transferred to the parent program. This feat is accomplished through Interrupt 21h-Function 4Ch-Terminate Program. When the child program terminates with this function, it can place a return code in register al which the parent program can read. Other means of handling termination codes are beyond the scope of this article.

Summary

In this article, the discussion of memory overlays was begun. Thus far, only overlays of unused memory was described. The size of the parent program must be reduced with the Set Block function, then child programs can be loaded and executed by the Execute Program function. The parent program regains control when the child process terminates. This technique permits the execution of selected processes from a large number of processes stored on mass storage, all of which would not fit into memory. These processes can be any executable files, generated by any means, which are compatible with the data sets of the parent program if any. In the next article, the use of overlay linkers and Subfunction 3 of the Execute Program function will be discussed. Overlay linkers, like the Microsoft Overlay Linker, are the easiest way to overlay program

code. Subfunction 3 of the Execute Program function differs from Subfunction 0 in that it overlays and erases part of the parent program.

References

1. Duncan, Ray, *Extending DOS*, 1990, Addison-Wesley Publishing Company, Inc., Reading Massachusetts.
2. Microsoft Corporation, *Microsoft Macro Assembler 5.0 Microsoft CodeView and Utilities*, 1987, Microsoft Corporation, Redmond, Washington.
3. Microsoft Corporation, *Microsoft Macro Assembler 5.0 Programmers Guide*, 1987, Microsoft Corporation, Redmond, Washington.
4. Microsoft Corporation, *Microsoft MS-DOS Operating System Programmers*

Reference, 1984, Microsoft Corporation, Bellevue, Washington.

5. Phoenix Technologies Ltd., *System BIOS for IBM PC/XT/AT Computers and Compatibles*, 1989, Addison-Wesley Publishing Company, Inc. Reading, Massachusetts.
6. The Waite Group, *The Waite Group's MS-DOS Developer's Guide*, second edition, 1989, Howard W. Sams & Company, Indianapolis, Indiana.

Trademarks

IBM, XT, and PC-DOS are trademarks of the International Business Machine Corporation.

Intel is a trademark of the Intel Corporation.

Microsoft and MS-DOS are trademarks of the Microsoft Corporation.

Listing

```

COMMENT * PROGRAM LOADPGM.EXE
This program accepts the name of another executable program
designed to process data files, lists the data files to be
processed, and processes the data file selected. Once the
program is initiated, the user may process as many data files
as desired by simply typing the name of the data file. This
program is intended to illustrate parent/child processes
although it does have practical applications. The executable
program, including path information, must be included as the
first and only parameter in the command line and the
file extension must be explicitly included. The following
example shows how to execute this program:

LOADPGM C:\PROGRAMS\FILE.EXE

Switches and other parameters associated with the invoked
program are added to the data file name information when
LOADPGM asks for that information. For example:

FILE.DAT /w/e

Copyright (C) 1990 by David W. Lind.

No expressed or implied warranties are made for this program
with respect to merchantability or fitness for a particular
purpose. The user assumes all responsibility for any damages
resulting from the use of this program. *

data SEGMENT ;Program Data Segment
istop DB ? ;Counter for command line data
mess0 DB ' ERROR - Could not re-allocate memory.',10,13,'$'
mess1 DB ' What file do you wish to process?',10,13,'$'
mess2 DB ' Do you wish to terminate processing (Y or N, default is N)?'
DB 10,13,'$'
mess3 DB ' ERROR - The command line must identify an executable program.'
DB 10,13
DB ' For example: "LOADPGM C:\PROGRAMS\FILE.EXE"',10,13,'$'
stringd DB 'C:\COMMAND.COM',0 ;Path and file name terminated by 0
string DB 128 DUP(0) ;Values in string MUST be set to zero
cmddir DB 8,'/cdir /w',13 ;Command line data for directory
cmd DB 128 DUP(0) ;Command line data for data files
spsave DW ?
data ENDS
esdata SEGMENT
edata DB 14 DUP(0) ;INT 21h-Function 4Bh Parameter Block
esdata ENDS

code SEGMENT ;Program Code Segment
ASSUME cs:code,ds:data,es:esdata

start: ;Program Entry Point

```



```

mov     ax,data           ;Put address of data segment in ax
mov     ds,ax            ;Put address of data segment in ds

```

COMMENT * The following code uses data in the Program Segment Prefix (PSP) to release memory for additional code and read data from the Unformatted Parameter Area of the PSP. The PSP is the beginning of the main program. It contains instructions and data to return control to the Disk Operating System when the program terminates, error handling code, housekeeping data, two file control blocks (for the program files), and the Unformatted Parameter Area which contains the ASCII characters typed as a part of the command line when the program is called.

When the program is executed, the ds and es registers contain the SEGMENT address of the PSP which is also the beginning of the program in memory. The first program segment begins immediately after the PSP, usually at offset 100h (256) which is the size of the PSP. The cs register contains the segment address of the first executable instruction. The offset address, or value of register ip (instruction pointer), is set to 0. Thus, this program will always begin to execute at a memory location after the end of the PSP at the program entry point "start:". If one fails to include the statement "END start" at the end of the program; that is, one simply uses "END"; the program will begin execution immediately after the PSP which is a data area and will produce undesirable results. The ds and es registers need to be set to the segment addresses for the data and extra segments of the program respectively. The cs register is set by the operating system.

The ds register was set to the data segment address by the first two lines of code above. The es register will continue to point to the PSP segment address until the program memory is decreased and the command line data is transferred to the data segment.

Normally, the invoking (main or parent) program is allocated all the memory space available regardless of the program size. This memory space is decreased by INT 21h-function 4Ah, Set Block. The bx register must contain the number of paragraphs of memory to be allocated for the invoking program. One paragraph is 16 bytes of memory (16 bytes is the value of one unit in a segment register which is the way the operating system allocates memory). The amount of memory released for the invoked programs must be equal to or greater than the size of the largest invoked program. The es register must point to the beginning of the program's memory area, which is the PSP segment address, when the Set Block function is called. *

COMMENT * Release memory for invoked (child) programs. The size of this invoking program is determined. Then the memory allocated to this program is decreased by function 4Bh. The remaining memory, which is considerable, is then available to the invoked programs. In the unlikely event that the system cannot release memory, a message to that effect is displayed and the program is terminated. *

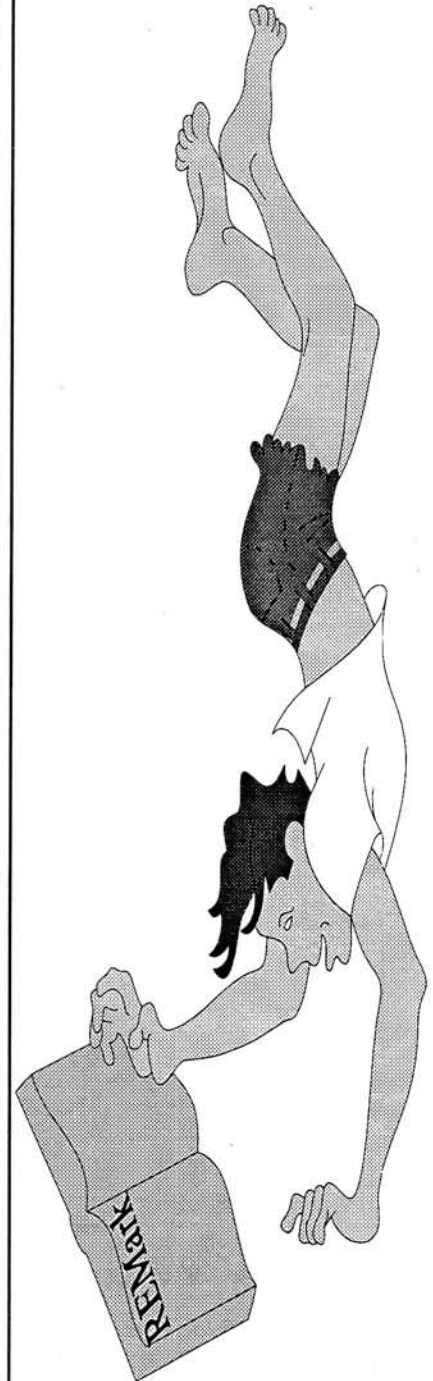
```

mov     cx,es             ;Put PSP segment address in cx
mov     bx,seg pgmend    ;Put address of last segment in bx
sub     bx,cx            ;Find length of this program
mov     ax,4A00h        ;Put function number in ah
int     21h             ;Call Set Block function
jnc     memok           ;If carry flag is clear, continue
mov     ah,09h          ;Else, put function number in ah
lea     dx,mess0        ;Put message offset in dx
int     21h             ;Call display string subroutine
jmp     end2            ;Terminate program

```

memok:

COMMENT * Read command line data from Unformatted Parameter Area in PSP segment, offset 80h. This data is the information which is typed after the name of the executable file when the file is executed and can include the names of other programs or files, paths, and switches, e.g. "C:\WP\WORD.EXE /B/e". The number of characters in the command line data is in the first



We have all kinds
of Subscribers

Why not YOU!

byte at offset 80h. This number includes the delimiter (a space character) at offset 81h. The useful data begins at offset 82h. For the purposes of this program, the command line data should only contain path data and the full name of an executable file because this program does not use switches and will prompt for the data files. *

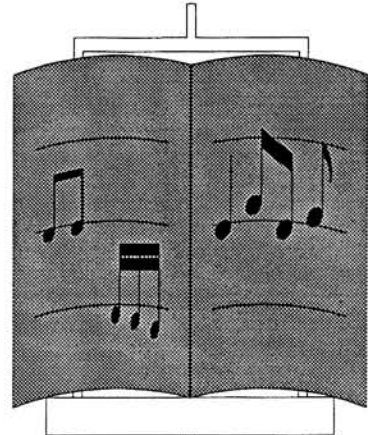
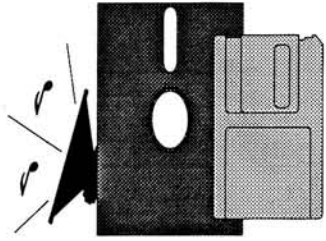
```

mov     di,80h                ;Put offset of Unformatted Parameter
                                ;Area in di
mov     al,es:[di]            ;Move number of characters in command
                                ;line to al
mov     istop,al              ;Put number of characters in istop
cmp     istop,0               ;Compare istop to 0
ja     continue              ;If non-zero continue
mov     ah,09h                ;Else, put function number in ah
lea     dx,mess3              ;Put address of error message in dx
int     21h                  ;Call display string subroutine
jmp     end2                  ;No program is to be invoked, exit
continue:
mov     bx,0                  ;Set string index to 0
inc     di                    ;Go to beginning of command line data
dec     istop                 ;Decrement number of characters to
                                ;compensate for skipping delimiter
psploop:
COMMENT * The index, di, is incremented at the beginning of this
        loop to skip the delimiter. Subsequently, it is
        incremented to point to the next character. *
inc     di                    ;Go to next character
mov     al,es:[di]            ;Put character in al
mov     string[bx],al         ;Move character to the data segment
inc     bx                    ;Increment the string pointer
dec     istop                 ;Decrement the number of characters
jnz    psploop               ;Go to psploop if istop is not 0
COMMENT * "string" must be terminated with 0 which was accomplished
        by initializing the values in "string" to 0. *
end0:
mov     ax,esdata             ;Put segment address of esdata in ax
mov     es,ax                 ;Set es to address of esdata

COMMENT * The following code (loop0) puts a copy of COMMAND.COM in
        memory, calls the dir function from COMMAND.COM, and
        displays a message asking the user to select a data file.
        Note that the switch "/c" must be used with COMMAND.COM to
        call a function when the disk operating system has
        relinquished control of the system to the program (see the
        definition of "cmddir" in the data segment). While
        COMMAND.COM executes, this program is also in memory. When
        COMMAND.COM completes the directory display, it terminates
        and releases the memory it used. Then execution control
        reverts to this program not to the operating system. The
        restoration of the segment registers and stack pointer after
        the interrupt call is only necessary for some subversions of
        MS-DOS version 2. For other versions, the restoration code
        can be eliminated. Of course, it does no damage to include
        this code. *
loop0:
mov     word ptr edata[02h],OFFSET cmddir ;Put command line data
mov     word ptr edata[04h],ds           ;address in parameter
block
mov     bx,OFFSET edata ;Put offset of parameter block in bx
mov     dx,OFFSET stringd ;Put offset of pathname etc. in stringd
mov     ax,4B00h ;Put function and subfunction # in ax
mov     sps,sp ;Save stack pointer for restoration
int     21h ;Call load and execute child program
mov     ax,stack ;Restore stack segment register
mov     ss,ax
mov     sp,sps ;Restore stack pointer
mov     ax,data ;Restore data segment register
mov     ds,ax
mov     ax,esdata ;Restore extra segment
mov     es,ax
lea     dx,mess1 ;Put offset of mess1 in dx
mov     ah,09h ;Put function number in ah
int     21h ;Call display string subroutine
mov     bx,1 ;Initialize bx to 1

```

New Software Product!



The Electronic Clavier
P/N 885-6016

COMMENT * The following code (loop1) reads characters from the keyboard until a carriage return (ASCII code 0Dh) is read and places the characters in "cmd" which is the command line for the next invoked program. *

```

loop1:
mov     ax,100h                ;Put function number in ax
int     21h                    ;Call Keyboard Input with Echo
cmp     al,08h                 ;Compare input to backspace
jne     nob                    ;If not backspace go to nob
cmp     bx,1                   ;Else, compare bx to 1 for first time
je      loop1                  ;If first time, try again
dec     bx                     ;Else, decrement bx
jmp     loop1                  ;Try again

nob:
mov     cmd[bx],al             ;Put character in "cmd"
cmp     al,0Dh                 ;Compare to carriage return
je      end1                   ;If carriage return, exit loop1
inc     bx                     ;Else increment string pointer
jmp     loop1                  ;Read next character

end1:
dec     bx                     ;Remove carriage return from count
mov     cmd,b1                 ;Put count at beginning of "cmd"
mov     word ptr edata[02h],OFFSET cmd ;Put command line data address
mov     word ptr edata[04h],ds  ;in parameter block
mov     bx,OFFSET edata       ;Put offset of parameter block in bx
mov     dx,OFFSET string      ;Put offset of "string" in dx
mov     al,00h                ;Put subfunction number in al
mov     ah,4Bh                 ;Put function number in ah
mov     sps,sp                 ;Save stack pointer for restoration
int     21h                    ;Call Load and Execute function
mov     ax,stack               ;Restore stack segment register
mov     ss,ax
mov     sp,sps                 ;Restore stack pointer
mov     ax,data                ;Restore data segment register
mov     ds,ax
mov     ax,esdata              ;Restore extra segment
mov     es,ax
mov     ah,09h                 ;Put function number in ah
lea     dx,mess2               ;Put offset of mess2 in dx
int     21h                    ;Call Display String subroutine
mov     ax,100h                ;Put function number in ax
int     21h                    ;Call Read Keyboard with Echo
cmp     al,79h                 ;Compare character to "y"
je      end2                   ;If "y", terminate
cmp     al,59h                 ;Compare character to "Y"
je      end2                   ;If "Y", terminate
jmp     loop0                  ;Else, process next data file

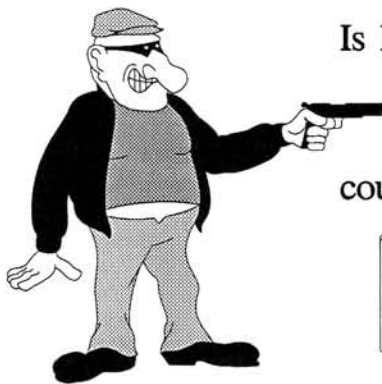
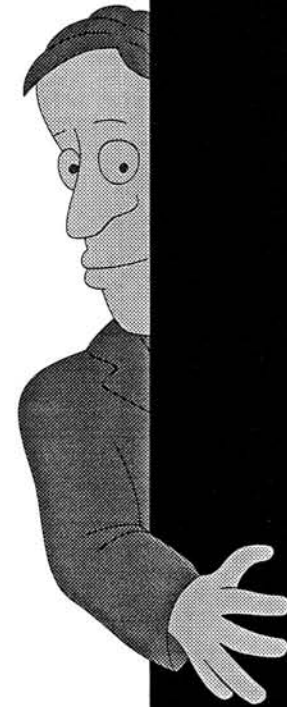
end2:
mov     ah,4Ch                 ;Put function number in ah
int     21h                    ;Call terminate function

code    ENDS

stack   SEGMENT stack          ;Program stack segment
        DW      64 DUP(?)      ;Define stack space
stack   ENDS
pgmend  SEGMENT                ;Dummy segment to define end of program
pgmend  ENDS

END     start                  ;Mark end and define start

```



Is lack of computer knowledge
holding you up?

"Powering Up"
could be your best protection!



"Powering Up"
by Bill Adney
Order today!
ZUG (616) 982-3463

We are still trying to get
caught up on our issues.

We would like you to know how
much we appreciate your patience.
Thank You!

QUIKDATA - 14 YEARS OF H/Z SUPPORT!

YOUR H/Z RAM & DRIVE EXPERTS!

ACCELERATE YOUR PC!

From Sota Technologies, Inc., the fastest and most proven way to breathe new life in your Heath/Zenith PC/XT computer, giving it -AT compatible speeds! Turn your turtle into a -286 rabbit with a 12.5 Mhz 80286 accelerator board. Complete with 16K on-board CACHE.

There are other ways to speed up your H/Z PC/XT computer, but you spend too much and get too little. Our EXP12 286i is the effective solution, making your H/Z150/160/150/158/159 series of computers, or any general PC/XT computer faster, in many cases, than a standard IBM AT type computer! Simple half-card plug-in installation with switchable speed control. **You won't believe your stop watch!**
EXP-12 - \$295
EXP386 - \$395 Much faster 16Mhz 80386 SX version

MEMORY UPGRADES

Note: All memory upgrades come without memory chips. 150ns 256K DRAMs are \$1.79 as of this printing.

Z150MP - \$17 Will allow you to upgrade your H/Z150/160 to up to 704K on the main memory board, using up to 18 256K DRAM chips.

MEGARAM - \$43 Upgrades your H/Z150/160 series with up to 704K of main memory, and about 512K for RAMDRIVE memory. Includes documentation, software RAMDRIVE disk, PAL and jumper wire. For the full 1.2 megs total memory, 45 256K DRAM chips are required.

EAZYRAM - \$95 Upgrades EaZy PC from 512 to 640K

ZMF100 - \$53 Will allow you to upgrade your H/Z110/120 (old motherboards; with p/n less than 181-4918) to 768K system RAM. Requires 27 256K DRAM chips.

Z100MP - \$76 Similar to ZMF100 above, but for new motherboards with p/n 181-4918 or greater.

Z159 EMS LOGIC UPGRADE includes the PAL chip, logic chips and one bank of 256K (9) chips. Up to two banks (18) can be installed for a total system RAM of 1.2 meg on the Z159 main board.
Z315-1 - \$79!!

Z248/12, Z286LP RAM UPGRADE Z605-1 consists of 2MB SIMM 80ns RAM kits to upgrade your H/Z systems.
Z605-1 - \$179

Z386/25, Z386/33, Z386 EISA 2MB SIMM 80ns UPGRADE to add increments of 2MB to these systems.
38633M-2 - \$295

SIP1X9-B - \$79 1 megabyte x 1 SIPP module, 80ns

WINCHESTER UPGRADE KITS

PCW20 - \$249 Complete MFM winchester setup for a H/Z150, 148, 158, 159, 160, PC etc. Includes 21 meg formatted half-height Segate ST-125 37ms drive, WD (DTC for Z148) controller, cable set, doc.

PCW30 - \$295 32 meg with 38ms Segate ST-138.
PCW40 - \$319 42 meg with 28ms Segate ST-251-1.
PCW80 - \$595 80 meg with Segate ST-4096 full size drive.

We also have the DTC controllers (\$59) and daughter board expansions (\$65) to place a hard drive in the H/Z148 computers.

BARE WINCHESTER DRIVES/CONTROLLERS

ST-125 - \$219 21 meg 37ms autopark 3.5" drive in 5" frame
ST-138 - \$249 32 meg 38ms autopark 3.5" drive in 5" frame
ST-251-1 - \$289 42 meg 28ms autopark HH drive
ST-4096 - \$559 80 meg 28ms autopark FH drive
WDCON - \$59 PC/XT hard drive controller board
WDATCONF - \$129 1:1 interleave HD/floppy controller for AT's
ST-2182 - \$995 160MB ESDI half-height drive
DTC-6260 - \$149 ESDI HD/floppy controller
IDE hard drives and controllers for AT computers:
ST08 - \$54 IDE controller for 2 hard drives and 2 floppies
ST-1090 - \$495 80mb 3.5" 15ms IDE hard drive
ST-126 - \$595 111mb 3.5" 15ms IDE hard drive
ST-157A - \$279 40MB 3.5" 15ms IDE hard drive

FLOPPY DRIVES

T28M - \$159 2.8MB 3.5" floppy
MF501 - \$71 5" 360K DS/DD drive
MF504 - \$81 96 TPI 1.2 meg AT/Z100 drive
MF353 - \$79 720K 3.5" drive in 5" frame
MF355 - \$89 1.4 meg 3.5" AT drive in 5" frame
SIEM-R - \$39 40 track SS refurb (H8/H89 type)
TM100-4R - \$75 80 trk DS refurb (H8/H89 type)
TM100-2R - \$69 40 trk DS refurb (H8/H89 PC type)

ANY DRIVE IN YOUR PC/XT/AT

With the CompatiCard, you can install up to four additional drives, of any type in your PC/XT/AT computer. Add a 1.2 meg 5" floppy, or a 1.44 meg 3.5" floppy, or any other drive, including 8" to your system. The CompatiCard (CCARD) will handle up to four drives, and the CompatiCard II (CCARD2) will handle up to 2 drives. CCARD4 has boot ROM to allow it to be used as primary boot controller in systems that allow you to remove floppy controller in some systems. Also handles 2.8MB 3.5" floppy drives. Additional cables and external enclosures may be required.
CCARD2 - \$79 CCARD - \$99 CCARD4 - \$125

ADD ANY FLOPPY TO ANY LAPTOP OR PC WITH A PARALLEL PORT easily and inexpensively. With Backpack, you simply connect the external unit to your parallel printer port (do not lose printer function), install software, and away you go! No expansion boxes needed for laptops, and no slots required. Too easy to be true, but it is. Want a 2.8mb/1.4mb/720K floppy on your MinisPort? Want to add a 1.2 meg to your laptop? Want to add an additional drive to your desktop? Plug it in and go. 2.8MB 3.5" version will read and write 2.8 megabyte format, 1.4 meg format and 720K format. 1.2 meg version will read 1.2 meg and 360K floppies, and write 1.2 meg format.

BPACK2.8 - \$319 2.8 Mb version
BPACK1.4 - \$269 1.4Meg/720K version
BPACK1.2 - \$269 1.2/360K version
BPACK360 - \$269 360K only version

8-BIT/Z100

We carry a full line of replacement boards, parts and power supplies for the H/Z89/90 and Z100. We also have some H8 boards available. We continue to fully support and carry a full line of hardware and software products for the H8/H89/90 and Z100 computers including almost the full line of Software Toolworks items. Other items we carry include diskettes of all types, printers, modems, hard drives, etc.

OTHER STUFF

Quikdata also carries BIOS ROM upgrades for most H/Z PC/XT/AT computers, spike protection filters, backup power supplies, tape backup units, modems, printers, cables and ribbons, disk drives and diskettes of all types, external hard drive and floppy drive enclosures, cables and connectors, laptop batteries, CMOS batteries, video monitors and video cards, memory cards, memory chips and ICs, joysticks, accessory cards, serial and bus mouse, a variety of useful and most popular software and much more! **Need a PC/XT/AT computer?** Tell us what you want and we will quote you a price on one of our custom assembled QD computers made up to your exact specifications! Price is lower than you think for a high quality system.

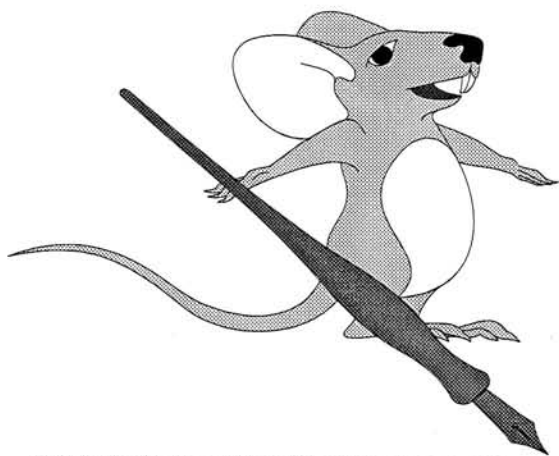
Call or write in to place your order, inquire about any products, or request our new free no obligation catalog. VISA and Master Card accepted, pick up 2% S&H. We also ship UPS COD and accept purchase orders to rated firms (add 5% to all items for POs). All orders under \$100 add \$4 S&H. Phone hours: 9AM-4:30PM Mon-Thu, 9AM-3PM Friday. Visit our bulletin board: (414) 452-4345. FAX: (414) 452-4344.

QUIKDATA, INC.

2618 PENN CIRCLE
SHEBOYGAN, WI 53081-4250
(414) 452-4172

Getting Started With . . .

Alan Neibauer
11138 Hendrix Street
Philadelphia, PA 19116



Mice and Pens

While the keyboard is still the most common means of communicating with your computer, there are other ways of inputting data to your PC. One of the most familiar alternatives is the mouse.

The mouse is electronically connected to the screen's cursor, so when you move the mouse on your desktop, the cursor moves along with it. Since you can move the mouse in any direction, you get total flexibility to point anywhere on the screen without moving in row and column increments as you must with the keyboard's arrow keys. So with graphic programs, you draw on the screen by dragging the mouse across the table top, just as you would draw on paper using a pen or pencil.

On top of the mouse are one or more buttons. Use the buttons to select on-screen options, in what is known as a point-and-shoot operation. You move the mouse until the cursor is on the option you want to select, then quickly press and release (or click) one of the mouse buttons. This can initiate a program command without having to lift your hand from the mouse or press a keyboard key.

Since most mice have two or three but-

tons, programs designed to work generically with mice will have left and right mouse commands. Click the right button to perform one function, the left button for another.

There are two general ways that mice operate, mechanically and optically. In the bottom of a mechanical mouse is a rubber ball that rolls along the table top as you move the mouse. On the inside of the mouse, the ball rotates against two rollers. Circuits connected to the rollers convert the physical movement into electronic signals that are transmitted to the computer (Figure 1). There is a third roller that keeps the ball tightly against the other two, but it does not record any movement.

Picture moving the mouse straight down on the table. The downward movement of the ball causes one of the rollers to move as well. The roller's movement is transmitted to the computer and interpreted as downward cursor movement, the same as pressing the down arrow key on the keyboard.

While there are only two rollers that detect movement, diagonal activity is detected when both rollers rotate at the same time. The relative movement of each roller

determines the direction. For example, when both rollers are moving at the same rate, the cursor is being moved diagonally at a 45 degree angle to the horizontal axis.

Rather than using a rubber ball, optical mice sense movement using light beams and photodetectors mounted at right angles to each other (Figure 2). You move the mouse across a specially constructed mouse pad printed with a grid pattern. The mouse detects variations of light, and converts it into electronic signals.

Installing Mice

In order to use a mouse, you need three things.

1. You must install a mouse driver.
2. You must physically connect the mouse to your computer.
3. You must use a program that accepts mouse input.

A mouse driver is a program that works in the background, communicating electronic signals back and forth between the mouse and the operating system. A mouse driver can be either a system file (with the SYS extension) or an executable program. The format of your driver determines how you install it on your system. If your driver program is an executable file, such as MOUSE.COM, you have to add the MOUSE command to AUTOEXEC.BAT. If the driver is a system file, such as MOUSE.SYS, you will need to add a device command in CONFIG.SYS.

The first step in installing the driver is to determine the type of driver you have and to place it on the root directory. Locate the driver on the disk that came with your mouse. You might also find a Microsoft mouse driver with programs such as Windows and various Windows applications. Copy the driver to the root directory of your hard disk or place it on the floppy disk

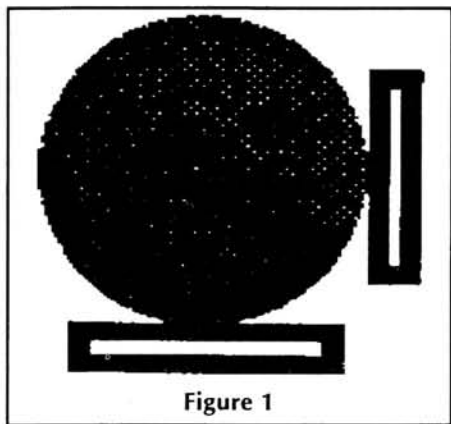


Figure 1

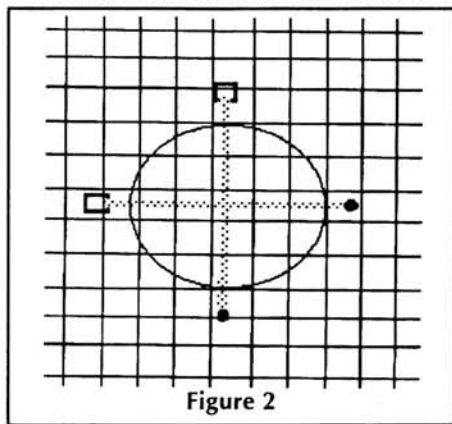


Figure 2

that you use to start your system.

Using a Device Command

If your driver is a system file, you have to create or modify a file called CONFIG.SYS that is on your root directory or bootable floppy disk. Use these steps to install support for a system file.

1. Log onto the root directory of your hard disk, or insert your boot disk into drive A and log onto drive A.
2. Display a directory of the disk to see if a file called CONFIG.SYS already exists. Use the command DIR CONFIG.SYS. If the file is not on your disk, skip ahead to step 5.
3. Type COPY CONFIG.SYS CONFIG.BAK to make a backup copy of the original file. This preserves the original in case you make any mistakes when creating the new file.
4. Type TYPE CONFIG.SYS, then press Enter to display the file on your screen so you can copy the lines to the new version.

If the file is large, press Ctrl-P, then use the type command to print a copy of the file. When the printout is complete, press Ctrl-P to turn off the printer function.

5. Type COPY CON: CONFIG.SYS, then press Enter.

The cursor will go to the next line but nothing else will happen.

6. Type DEVICE=MOUSE.SYS, then press Enter. The cursor moves to the next line.
7. Now retype any other lines that were in the original file. Copy them exactly as they appeared in the original file, pressing Enter after each line.
8. When you are done, press Ctrl-Z. ^Z will appear on the screen signifying the end of your input.
9. Press Enter. The commands you entered are written to the file CONFIG.SYS.
10. Press Ctrl-Alt-Del to reboot your system. DOS will read CONFIG.SYS and execute its commands.

If your mouse driver is an executable program, you have to add it to your autoexec batch file. This way, it will be run every time you start your computer. Follow these steps if your driver is a COM or EXE program.

1. Log onto the root directory, or insert your boot disk in drive A and log onto drive A.
2. Type COPY AUTOEXEC.BAT+CON: AUTOEXEC.BAT, then press enter. You'll see

```
autoexec.bat
con
```

If you see the message File Not Found, then the batch file doesn't yet exist on your disk. Type COPY CON: AUTOEXEC.BAT, then press Enter.

3. Type MOUSE, then press Enter.
4. Press Ctrl-Z to display ^Z.
5. Press Enter.

Adding the command does not execute

it until the next time you start your computer. To use the mouse immediately, type MOUSE to run the driver or AUTOEXEC to execute all of the batch file instructions.

If the mouse has more than two buttons, you must use special software supplied by the manufacturer to use anything other than the left and right buttons.

Connecting Your Mouse

Mice come in three versions, depending on how they interface with your computer.

Serial mice connect to your computer's serial port. Many mice use a nine-pin connector but come with an adapter for use with a 25-pin interface. If you have a 25-pin serial port, plug in the adapter, then attach the mouse.

Since many systems have only one serial port, you might not want to use it for a mouse. An alternative is the bus mouse that comes with its own interface board that you plug into an empty expansion slot in the computer. With the bus mouse, you can use your serial port for a modem, printer, or other peripheral while using your mouse to draw and point. The boards are designed to work in almost any computer, but because of the different bus arrangements, a jumper block is usually provided to configure the board for your system. When installing a mouse controller, carefully follow the instructions packaged with it.

As mice become more popular, many systems come with a dedicated mouse port. In most cases, they are 9-pin female connectors that resemble a 9-pin serial port, but are labelled Mouse. (Some serial mice can plug directly into the port, but check your documentation carefully before plugging it in.)

Selecting A Mouse

In selecting a mouse, consider the type of interface that is best for your system and which mice support your applications.

The standard of compatibility is the two-button Microsoft Mouse. You can use a Microsoft mouse with every program that supports a mouse. Mice that are truly Microsoft-compatible will work with a Microsoft mouse driver. There are other mice that come with their own drivers that provide various levels of Microsoft compatibility. Select a non-compatible mouse only if it is supported by your application or has its own drivers. If possible, test the mouse with your software before purchasing it.

Other features to look for are resolution, tracking, and acceleration.

The mouse rollers or photodetectors detect movement as a series of small points. These points are then converted into screen pixels to move the cursor. The number of dots in each inch of space is called the resolution. With a high resolution, your movements will be more exact and you'll

have to physically move the mouse less for a corresponding move on the screen. Most mice have a resolution in the range from 200 to 400, although some, like the Logitech Series 9 mouse, can be adjusted using software from 50 to 19,000.

Resolution isn't critical for text applications, such as selecting menu options. But it is important for drawing and drafting applications where you want precise movement in small increments.

Tracking is the ability of the mouse to glide smoothly and straight. Using a mouse with poor tracking, you have to constantly push the mouse to compensate for its defect. A mouse with good tracking will make it easier to draw straight horizontal and vertical lines with drawing and painting programs.

A mouse with high acceleration can accurately translate quick movements, positioning the cursor at the proper screen location. If a mouse has poor acceleration, you'll have to physically move it slower to accomplish corresponding cursor movement on the screen.

Software

Finally, you'll need applications that can recognize mouse movement. Many older, text oriented programs, will not work with a mouse even if it is installed properly. Newer programs, including text-oriented word processors, have built-in mouse support. You can use the mouse to position the cursor and select menu options.

Some mice, however, come with installable application drivers. Use these to add mouse support to older programs that do not include it. There are drivers, for example, that modify older versions of Lotus 1-2-3 to include mouse pull down menus.

Mice Alternatives

In addition to mice, there are a number of other available input devices. A trackball, for instance, is a mechanical mouse turned upside down, with the roller ball facing up. Instead of rolling the mouse around the desk, you roll the ball with your fingertips or palm. Trackballs require minimum desk space because they physically remain in one position, and because you don't have to move your wrist or arm, many users find trackballs less tiresome than mice.

A puck is identical to a mouse except it has crosshairs in front to indicate its exact position. It is used for tracing and drawing, or where precise positioning is required. You can also use a digitizer pen that gives the feeling of drawing on paper. Both devices require a digitizer tablet which has a fine grid of wires just beneath the surface. The pointing device determines its position electronically by sensing the coordinates of the grid.

The WIZ

The WIZ, manufactured by CalComp, is a puck system that includes extensive driver support for application programs. As shown in Figure 3, the WIZ has crosshairs and three rocker-type buttons. Because each button has two positions, the puck can generate six distinct states (Figure 4).

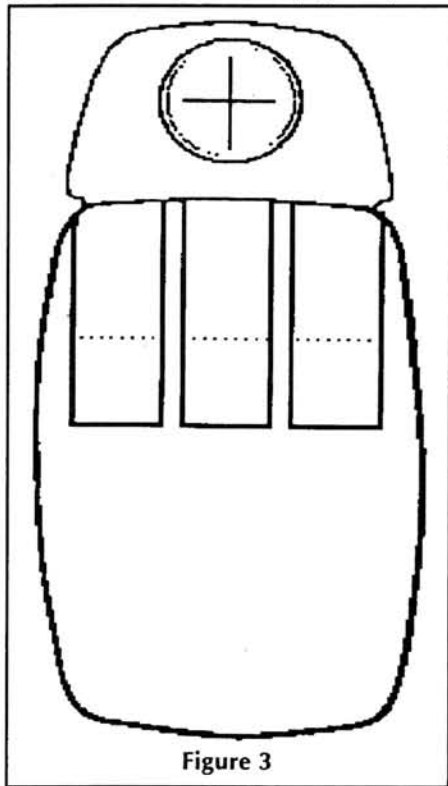


Figure 3

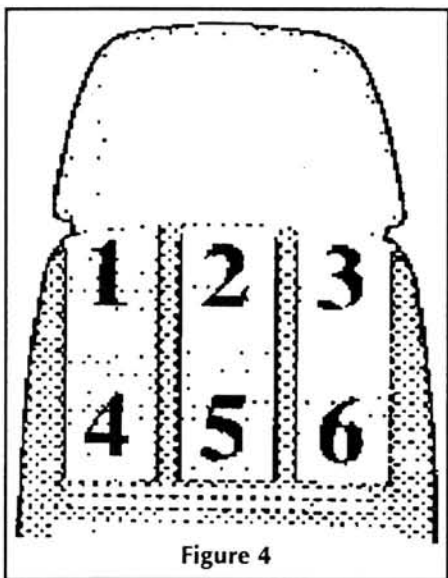


Figure 4

The WIZ can be used for drawing and interacting with programs that work with mice. But what makes the WIZ unique is a collection of templates and drivers designed for popular applications that are not mouse-compatible, such as WordPerfect 5.0 and Lotus 1-2-3. Each template contains a series of menus that include the program's major functions. You place the template on the digitizer tablet, then ac-

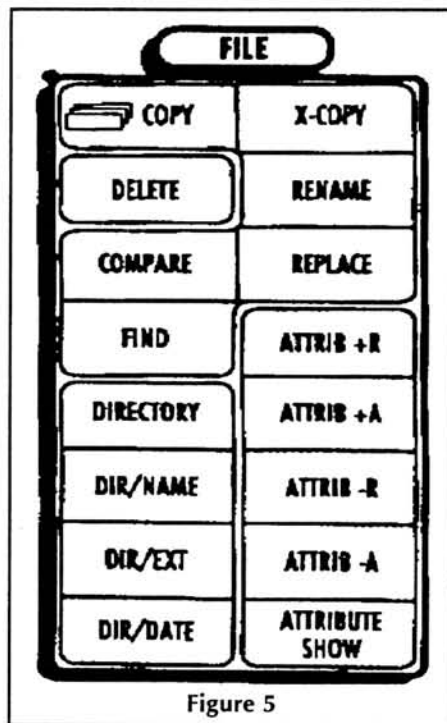


Figure 5

cess a function by pointing to the item with the crosshairs and clicking the puck. There is even a template for DOS, so you can display directories and manipulate files without typing DOS commands, even from within application programs. Figure 5 illustrates a small portion of the DOS template showing common file operations.

The templates are also very handy for Windows-based applications that already have mouse support. Figure 6, for instances, shows a portion of the Windows template designed for use with the Write word processing program. It allows you to perform functions with a single mouse-click that might take several using Write's own menus and mouse buttons. The WIZ comes with its own DOS and WINDOW's drivers. The templates and application drivers are sold separately.

Most templates also have a special macro

area. This includes several blank boxes for which you can program your own functions. The WIZ driver allows you to create keyboard macros assigned to those buttons, so you create and run a custom series of keystrokes with a single click.

The WIZ can also be configured for relative and absolute positioned. With relative positioning the screen position is determined by the distance you move the puck. You can lift up the puck and move it elsewhere without changing the screen position. This is the usual way most mice operate.

Absolute positioning, on the other hand, duplicates the X-Y coordinates on the pad with those of the screen. To point to a specific part of the screen, position the pointer on the corresponding position on the tablet. When the puck crosshairs are pointing at the center of the pad, the cursor will be at the center of the screen.

The system gets power directly from your system. If you have a dedicated mouse port, the WIZ tablet connects directly to the port, and the puck connects to the tablet. If you don't have a mouse port, you connect the system via the keyboard and serial ports. The arrangement is shown in Figure 7. The tablet plugs into the serial port and attaches to the keyboard interface using a Y-connector. The puck connects directly into the tablet. (If your keyboard connects directly into your system, as with the older Compaq portable computers, you need to order a separate power supply.)

The WIZ also has an optional pen. The pen has a button on the side and a tip that can be depressed against a hard surface. These act as mouse buttons and can be configured using the WIZ drivers.

For more information on the WIZ, contact:

CalComp, Inc.
2411 W. La Palma Avenue
Anaheim, CA 928-1
800-458-5888

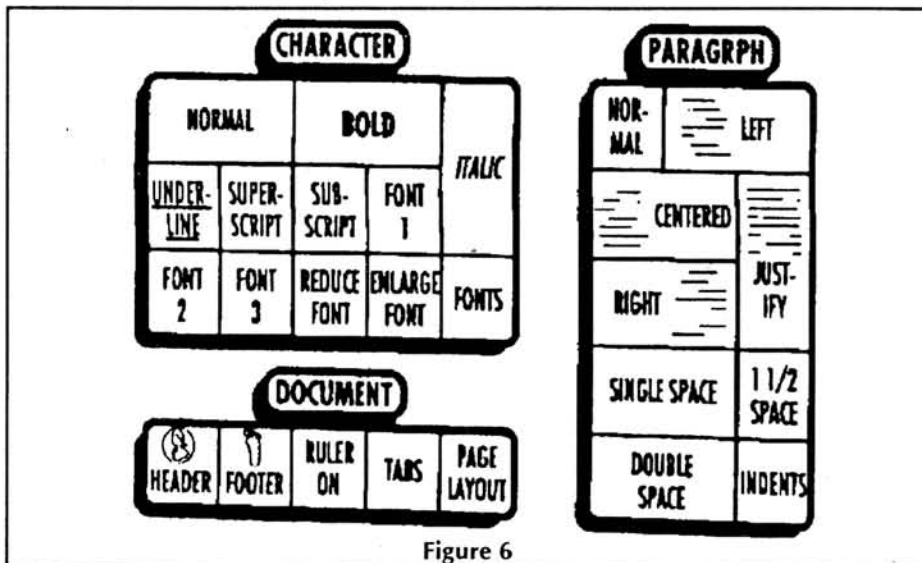


Figure 6

Light Pens

Instead of simulating the face of the screen on a pad or your desktop, light pens allow you to work directly on the screen. When you point to the screen, the light pen senses the exact moment the monitor's scanning line passes under its photo-detector. Because of the speed of the vertical sweep rate, the pen appears to react instantly, signaling the driver software to perform some action based on that coordinate position.

For example, suppose you point to a menu item that is between character positions 5 and 10 on the screen's 5th line. The pen transmits a signal to the computer when it senses the scanning line. The computer calculates where the scan line was at that time, and determines which menu option was at that position.

Pen-Based Systems

Entirely different from traditional light pens are the pen-based systems that are currently being developed, and which are the topic in many trade journals. Microsoft is working on a Pen Windows system, while Go Corporation is developing a DOS alternative called Pen Point.

Both of these systems are being designed to recognize some degree of hand-written input. At one level, pen-based systems can recognize gestures and symbols. A gesture might be drawing a circle around a group of cells in a spreadsheet to select the block; a symbol might include a check

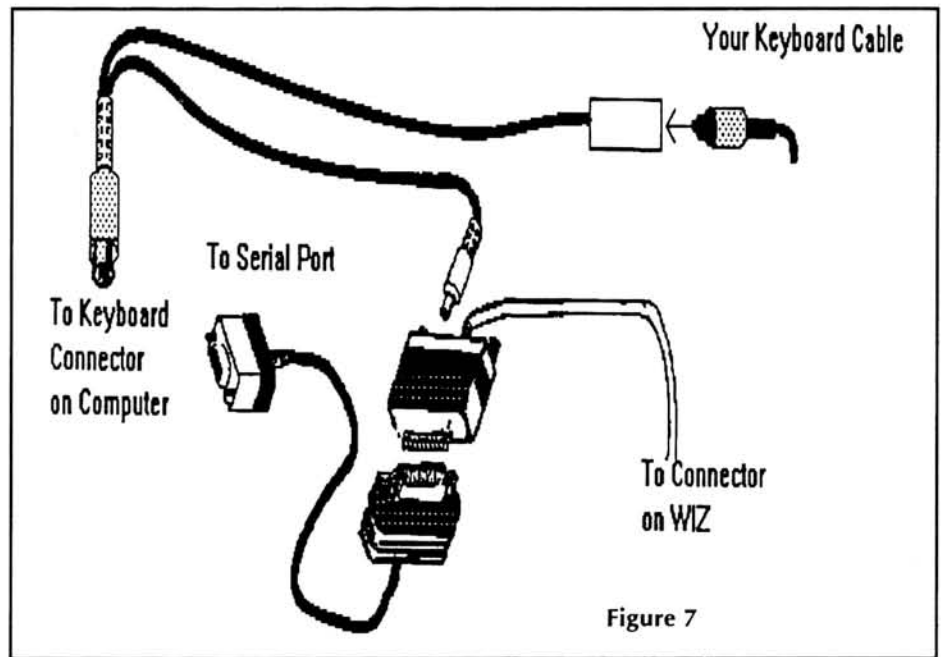
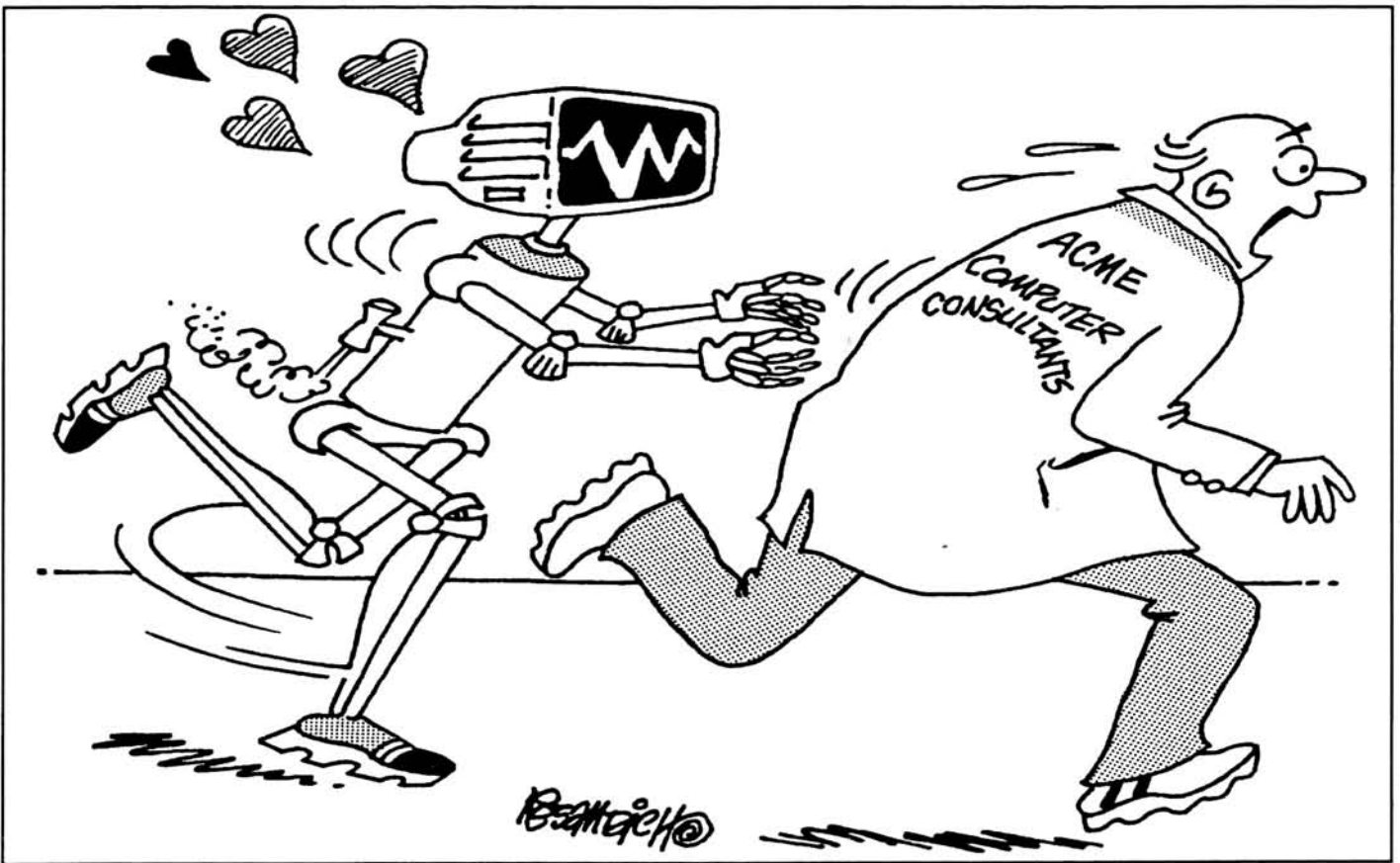


Figure 7

mark or cross-out drawn in a selection box. Imagine deleting a block of text from a document by just crossing it out with a stroke of the pen!

At a higher level, pen-based systems will recognize block and cursive writing, as well as punctuation characters and numbers. Instead of typing on a laptop keyboard to take notes at a meeting, a pen user would write quietly as if using a tablet and pencil.

Pen-based hardware and software are now in the prototype stage, with manufacturers promising release sometime in 1991. No matter what the final form, however, pen based systems will never replace the traditional keyboard for word processing and other data entry functions. Their main use will be for notebook-type systems, quick data entry and selection tasks, or as part of an overall graphical user interface.



Minimum Effort Hard Drive Backup

Ron Siebers
9334 Wentlock Road
Woodbury, MN 55125-3420

Backing up hard drive files on your IBM or compatible can be an endless chore, something you often fail to do until it's too late. The hard facts are that hard drives do fail, new or old. If you don't have some backup plan, you will eventually get caught. Think of the evolution you have gone through as you constantly change your AUTOEXEC.BAT and CONFIG.SYS files to accommodate each new program you add to your hard drive. Could you reconstruct all those lines from memory? Doubtful. Since I hate doing things a second time and have a poor memory to boot (no pun), I have developed a shortcut way to back up all my root directory and key program files on one floppy disk. Interested? Read on.

Home computer users generally cannot afford nor justify the purchase of backup hardware such as tape cartridge systems or removable hard drives. The affordable backup method is usually floppies and perhaps a backup software package such as Fastback Plus or Norton Backup. Using most back up programs, it takes time figuring out what needs to be backed up or you back up everything including a lot of files you have already backed up. The problem with using back up programs is human nature. We get lazy after a while and fail to do the backups. Yes, I said WE. I've been guilty too. Sure as computers are going to become even faster, Murphy's Law will kick in and your hard drive will fail the day before you were going to back up your hard drive.

I have therefore developed a system of backing up key hard drive files on a single floppy and data or text files on other floppies. Using my system, you will always have at least 2 copies of every data and configuration file. Backing up a hard drive really consists of two separate tasks. The first is backing up data or text files you create with programs. The second is backing up configuration files for both the computer and its programs.

Computer & Program Configuration File Backup

Configuration files consist of those for your computer (AUTOEXEC.BAT, CONFIG.SYS and any BATCH files and special drivers you use) and those for your programs. You can reinstall programs from original floppy disks, but most have to be configured for your computer hardware and preferences. That can take as much time as copying the program files. One secret of my backup method is that each software program such as WordPerfect, Microsoft Word, Wordstar, Lotus 1-2-3, Harvard Graphics, Lotus Freelance etc. has either a setup (FILENAME.SET) or a configure (FILENAME.CFG) file that holds the setup information you established to personalize the program to your needs. These files are usually small ASCII text files of 2K

or less. By copying only these files to a floppy disk, you can back up the configuration of all of your programs with little effort or disk space. What about those dozens or sometimes hundreds of other program files? They are all on your original program disks, waiting to be copied onto a new hard drive if ever necessary. There is no need to waste time and floppies making more copies of these files.

Another secret is knowing which program files to back up. I have listed in the following table the files to back up for commonly-used programs. These are the programs I am familiar with. Other programs have similar files. You can usually tell them by the more recent date than the other program files. The date is changed to

Files to Back Up From Popular DOS Programs		
Program	Filename.ext	Function
WordPerfect	WP{WP}.set WP{WP}US.SUP *.WPM *.PRS	Setup Supplemental Dictionary All macros Configured Printer Files (not *.ALL files)
Microsoft Word	MW.INI *.GLY *.CMP *.STY	Setup Normal & special glossaries Normal, user & file dictionaries Styles & style glossary
Wordstar Prof.	WS.EXE (Use a separate disk)	Setup in Main program file
Lotus 1-2-3	123.SET 123.CNF 123.DCF & LOTUS.LCF (ver 3.0)	Setup file Configure file Configure Files
Allways (for 1-2-3)	ALLWAYS.CNF	Configure File
Harvard Graphics	HG.CFG	Configure file
Freelance Plus	FL.DIR GCIFL.CFG GCISD.EXE DV77777.DRV	Setup file Configured Files (All Similar-named Device Drivers)
Procomm Plus	PCPLUS.DIR PCPLUS.PRM	Directory file Setup file
Norton Commander	NC.MNU NC.INI TREEINFO.NCD	Menu file Initial Settings Hard drive tree info

Figure 1

the current computer date each time you update the program setup or defaults. All of these files may fit on a 360K floppy and normally will fit on the newer 720K 3.5" disks. The one exception is the older Wordstar program versions which update the WS.EXE file. This is a pretty large file.

You will note that I list a number of WordPerfect and MS Word files to back up. That is because these and most word processors also have several dictionaries including one you can add words to during spell checking. You may also create many style sheets while using the program. You may also have a subdirectory full of macro files you will want to backup periodically. Some printer drivers contain port, baud rate and font information that if backed up can save time reinstalling the program.

Data and Text Backup

That takes care of your program and computer configuration backup, but I have other suggestions concerning backing up data files. I always caution against saving data only on hard drives because you will lose it when the hard drive fails. If you have a fast hard drive, you hate not using it for data storage because it is so much faster to load from and save to than floppies. Also, programs that spool printing to the data drive, like WordPerfect, take longer to print from a floppy than from a hard drive.

A compromise is called for. The simplest way to assure you won't lose data and text files is to save them to both hard drive and floppy at the time of creation. This only takes a few extra seconds for normal file

sizes and therefore doesn't become a time consuming chore you end up postponing. It simply takes a bit of discipline to save each file twice at the time of creation or update and you can never lose a data file from crashing your hard drive. Backing up is spread out over your computer use and it doesn't become a 30 minute chore you tend to avoid. This way I always have a backup on floppy, but I can load files faster from the hard drive whenever I use them again. Many programs like WordPerfect and Lotus 1-2-3 make it easy to save a file twice. You first save it to the default drive\directory on your hard drive without exiting and then rename the path to the floppy drive and save again. Then you exit the program. I suggest using at least one floppy for each program you use. Major projects may each need their own disk.

Another part of your backup plan should be to have a bootable disk for use in case of hard drive problems. I also make this my configuration backup disk. Normally, this must be a disk type for your A: floppy drive, which is the only one the computer checks for a bootable disk before booting from the hard drive (IBM standard). The only exception is the few clone brands (such as Zenith) that have provisions to boot from other drives. If your boot floppy is a high density drive, certainly use a high density disk. It will give you more room for backup files.

Notice that I previously said that I back up key files. What are key files? They are certainly the AUTOEXEC.BAT and CONFIG.SYS files including any alternate ver-

sions. Having backup copies of these files should always be a top priority. The best way to assure this is to always copy them to your backup disk after making any changes. You should always copy them twice and rename one pair .HD since you will have to alter the statements to drive A: instead of C: in order to boot from the floppy.

Other key files you should back up include any special device drivers called in CONFIG.SYS or AUTOEXEC.BAT. This would include any screen saver you may use. Create the same directories on your floppy that you have on your hard drive and copy these files. Then your path statement need only be changed from C: to A: for the floppy versions of your commands. Next, if you have created any batch files, copy them into the same-named directory on your floppy using the copy *.bat command. If you use a menu program such as PC Tools, Norton Commander, Wonder-Plus, Dtree, Xtree or whatever, save the configuration and menu files for those programs to avoid having to create them over again. If you have room left, make a DOS directory and copy the most necessary DOS files (Format.com, Chkdsk.com, Ansi.sys, Vdisk.sys, Emm.sys etc.).

Restoration

You won't get caught by a hard drive failure if you follow this simple plan. You will have all of your configured program information on one floppy that you can boot from even while your hard drive is unusable. Once you have restored or replaced your hard drive, you can easily restore all your files. You won't have the handy automatic directory creation of backup software, but that is a small inconvenience. My hard drive procedure consists of three steps after achieving a formatted new hard drive.

1. Create the directories you had before.
2. Copy all the DOS files and menu program files to the hard drive from your original program floppies.
3. Copy AUTOEXEC.BAT, CONFIG.SYS, all your batch files, all your device drivers and your menu program files (if any) from your back up disk and try a hard drive boot.
4. Copy all your program files from original program floppies.
5. Copy all your program configuration and set up files from your back up disk into the appropriate directories.
6. Copy all your data and text files into the appropriate directories.

This should put you right back where you were before the hard drive failed. You will not need to reconfigure your programs nor recreate a custom program menu. It will all be as before. The real beauty is, you can do all this with minimum effort and without purchasing any special hardware or software. ✽

Technical Problems?

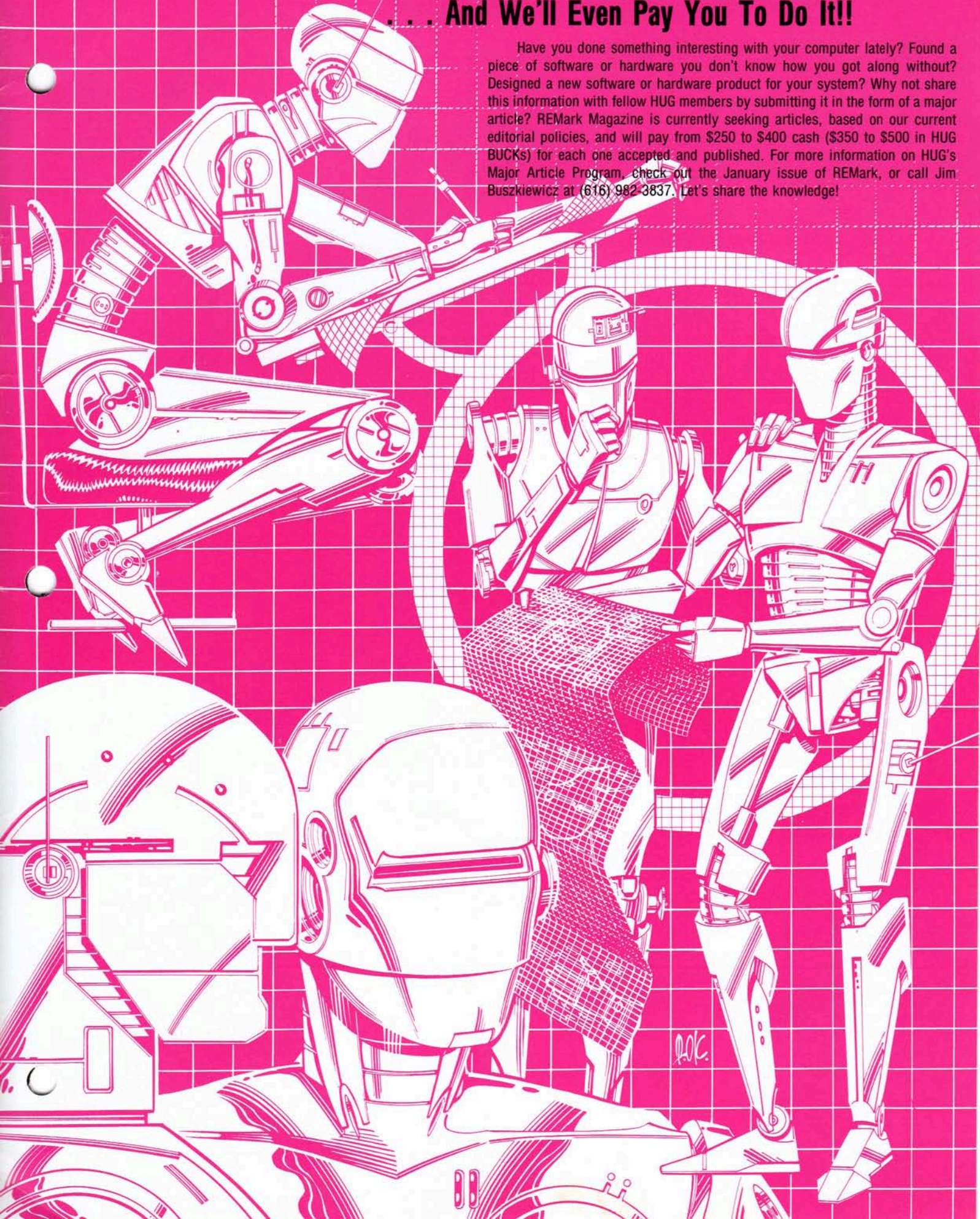
ZLink-COM1 Bulletin Board
message system is the answer!

Dial (616) 982-3956
with your modem today!

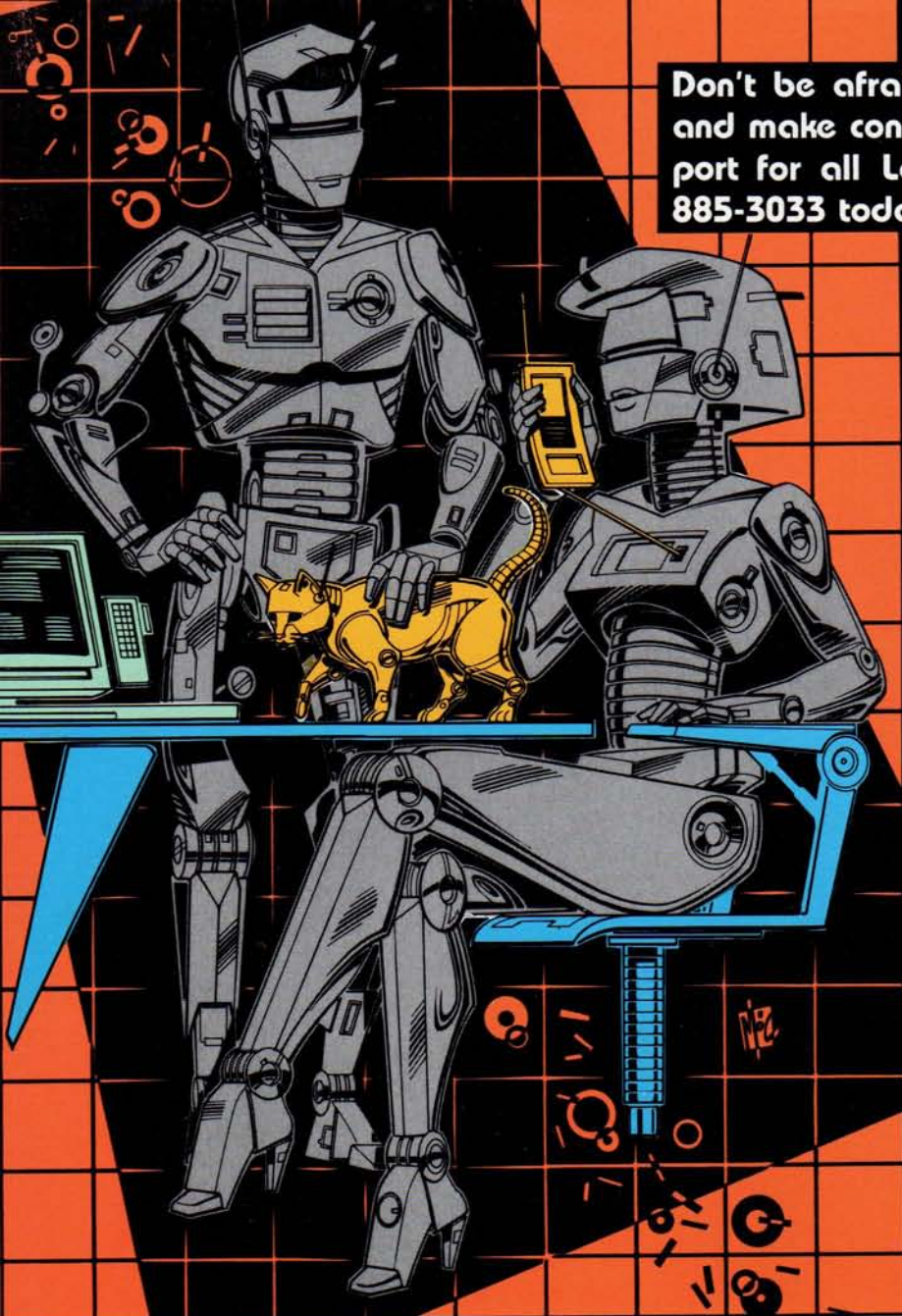
Zenith Data Systems

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

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



Don't be afraid to communicate! Get HUGMCP and make contact the easy way. Now with support for all Laptops, order HUG Part number 885-3033 today.



```
HUGMCP Commands

F1 - Prints This List, Your Storage Buffer Size, And How Many
    Bytes Are Presently In The Storage Buffer.
F2 - Allows Sending A Defined Message, Or Character Sequence.
    These Messages Are Entered Using The (F6) Setup Command.
F3 - Toggles The Storage Buffer On and Off. When The Buffer
    Is On, The (Buf) On The 25th Line Will Be High-Lighted.
F4 - Allows Saving Data To Disk From The Storage Buffer, Or
    Directly From The Mouse By Way Of XMODEM Protocol.
F5 - Allows Sending Data From Disk, Using Either XON-XOFF,
    Which Optionally Can Be Ignored, Or XMODEM Protocol.
F6 - Enters The Setup Mode So This Software Can Be Configured.
F7 - Clears Out Any Data That May Be In The Storage Buffer.
F8 - Send Data In Storage Buffer To Printer.
F9 - Exits Back To MS-DOS.

Storage Buffer = 524288 Bytes
Storage Buffer Usage = 0 Bytes

Select Message (0-0), (F1) To List, Anything Else To Abort --) _
F1:Msg F2:Msg F3:Buf F4:Save F5:Send F6:Clear F7:Clr F8:Print F9:Exit COM
```

```
HUGMCP Configuration Menu

1 This Function Allow The Baud Rate To Be Changed. Depending Upon Baud
    Rate You're Using, Normally F1 Would Be Valid After 300, 1200, 3000,
    12000, 24000, 48000, 96000, 192000, 384000, 768000, 1536000, 3072000.
2 This Function Allow You To Change The Word Parity. Normally, you
    should use NONE, EVEN, OR ODD. It's Acceptable To Put Some Values
    That It Also Necessary For XMODEM Protocol To Work Properly.
3 This Function Allow The Changing Of The Word Length. Normally The
    length Would Be Set To 8 Data Bits. This Value Is Acceptable To Put
    Some Values That It Also Necessary For XMODEM Protocol To Work Properly.
4 This Function Allow You To Enter Messages Which Can Be Automatically
    Sent With The F1 Key. The 0-0, F1-Computer Messages Can Be Used.
    Attention To It Special, It Should Contain Some Composites To Answer
    Questions, It's Special, It's Also Possible To Put Injection On Busy
    Waiting, Be Sure When This Program Is First Executed By Selecting The
    Enter Option During Setup.

Type (F6) F6: For More Help, Anything Else To Configure
F1:Msg F2:Msg F3:Buf F4:Save F5:Send F6:Clear F7:Clr F8:Print F9:Exit COM
```

```
HUGMCP Configuration Menu:

0 -> Modify Baud Rate
1 -> Modify Parity Type
2 -> Modify Word Length
3 -> Modify Or Add Auto-Messages
4 -> Miscellaneous Functions
5 -> Change Screen Color Assignments
6 -> Display Current Configuration
7 -> Make Changes Permanent

Select 0-6, (F1) For Help, Anything Else To Quit --) _

Baud Rate: 19200
Parity: NONE
Word Length: 8
Baudix: FULL
Response To Keyboard Disable: NO
Storage Buffer Data Parity Bit: SET TO ZERO
Send Modem Initialization Text: NO
Delete Charging: NO/NO
Modem Port Set To: COM1

F1:Msg F2:Msg F3:Buf F4:Save F5:Send F6:Clear F7:Clr F8:Print F9:Exit COM
```

ZENITH
data systems



Groupe Bull

BULK RATE
U.S. Postage
PAID
Zenith Users' Group

POSTMASTER: If undeliverable, please do not return.

\$2.50
P/N 885-2134